# RAMP-TSガイド

# Rapid Application Modernization Processガイド (ターミナルサーバー)

- RAMPを開始する方法
- 最新情報
- ライセンス要件
- インストールおよび構成
- RAMPの開始
- コンセプト
- 刷新の問題
- RAMP Windowsのチュートリアル
- スクリプト
- 画面の拡張
- 画面ラッパー
- プログラミング・テクニック
- 複数言語対応のRAMPアプリケーション
- トラブルシューティング
- よくある質問
- 付録

ご意見・ご感想は、LANSAサポートlansasupport@lansa.com.auまでお願いします。

免責事項:このガイドの情報が正確であるようにあらゆる努力をしてい ますが、LANSAはいかなる場合においても、このガイドの使用に起因す るいかなる損害に対しても責任を負いません。LANSAは明示または暗黙 のいかなる保証もしません。 エディション日付:2013年5月9日、EPC130100 © 2013 LANSA

## RAMPを開始する方法

RAMPを開始するには、次の5つのステップを実行することをお勧めします。

1.入門セクション「RAMPとは?」を参照する

2. 以下を参照し、5250アプリケーションの特性が刷新によってどのよう に変更されるかを理解する

刷新前の5250アプリケーション RAMPを使用した後の5250アプリケーション

3. 以下を参照し、刷新処理がどのように実行されるかを理解する

ステージ1:刷新フレームワークの作成

ステージ2:5250アプリケーションの増分的な刷新

ステージ3: IBM iプラットフォーム依存関係の排除

**4.** RAMPソフトウェアをインストールおよび設定する(「インストールおよび<br />
よび構成」を参照)

**5.** 重要な「RAMP Windowsのチュートリアル」または「RAMP Webの チュートリアル」を完了する

以上のステップを完了すれば、RAMPを使用した独自の刷新プロジェク トを計画および実行できるようになっているはずです。

#### RAMPとは?

RAMPは、iSeries (または) i5ベースの5250アプリケーションを迅速に刷新 するための段階的プロセスです。これはVisual LANSAフレームワークの 拡張です。



RAMPステージ1では、完了したときの刷新されたアプリケーションの外 観についてプロトタイプを作成します。

RAMPステージ2では、フレームワークで既存の5250アプリケーションの 表示を再確認し、増分的な刷新を行います。

RAMPステージ3はオプションですが、既存のアプリケーションが持って いる特定のiSeriesプラットフォーム依存性を削除するためにアプリケー ションの再構築を行います。

最終的な結果を理解するには、「刷新後のアプリケーション」を参照してください。

RAMPプロセスは、刷新されたアプリケーションの迅速で増分的な配布 をサポートするように設計されています。これは、1回のビッグ・バンで 結果が得られるまで何年間も作業しなければならないアプローチではあ りません。以前、ビッグ・バン・プロジェクトに関係していたことがあ る場合は、そのプロジェクトが引き起こす落とし穴がおわかりでしょ う。このRAMPアプローチでは、独自のペースで刷新を行うことができ るため、賢明な方法でリスクを管理できます。「主な利点」を参照して ください。

#### RAMPステージ1



RAMPステージ1では、プロトタイプ化された 刷新フレームワークを作成します。 このステップの目的は、「何か」ではなく 「何が可能か」についての見通しのよいビ ジョンを作成することです。つまり、単なる 現在のアプリケーションの反映ではなく将来 のビジョンとしてプロトタイプを作成しま す。

このプロトタイプ作成フレームワークは、以 下の理由で重要です。

- これは、5250画面パネルを再利用する場所と方法を定義します。
- これを使用して、開発者やエンド・ユー ザーなど、すべての関係者は得られる内容 や刷新プロジェクトの完了のために果たす べき内容を理解および検証できます。
- これは無駄なことではなく、実際の実行 可能なアプリケーション・フレームワーク へと発展します。

インスタント・プロトタイプ・ウィザードを 使用してプロトタイプを作成します。この ツールは、WindowsとWebブラウザー・アプ リケーション・デザインとの間の溝を埋める ため、5250のバックグラウンドを有する人々 には特に役立ちます。

大規模なアプリケーションの場合でも、刷新 フレームワークの配布にはわずか数日しかか かりません。プロトタイプ作成プロセスは非 常に迅速であるため、関係者による比較と検 証のために複数のプロトタイプを作成するこ とができます。

このステージの最後に、実際の実行可能なア プリケーション・フレームワークが配布され ます。これにより、得られる内容と実行する 必要がある内容をすべての関係者が確認でき るばかりでなく、プロジェクトの残りの部分 を簡単に定義および管理したり、さまざまな 成果物をプロジェクト参加者に割り当てたり することもできます。

アプリケーション・フレームワークは、プロ トタイプ部分を徐々にスナップアウトし、実 際のアプリケーション部分(再アニメーション 化された5250画面または新しいファンクショ ン)をスナップインすることで実際のアプリ ケーションへと発展します。

## RAMPステージ2



RAMPステージ2では、5250アプリケーション をフレームワークにスナップインして増分的 に刷新します。

## RAMPステージ2A

このステップでは、入力画面(一般にはメ ニュー)をフレームワーク内にスナップインし て、アプリケーションの使用頻度の低いセグ メント全体を迅速に刷新します。これは、ア プリケーションの使用頻度の低いすべてのセ グメントで行う必要があります。 この方法の主な利点は、5250画面にWindows ユーザー・インターフェースを導入し、非常 に少ない時間でフレームワークからこれらの

画面にアクセスできるようになることです。 ステージ2Bのようにアプリケーションの単一 の5250画面を個別に刷新する必要はないこと を理解することは非常に重要です。詳細につ いては、「RAMP Stage 2A: アプリケーショ ン・セグメント全体の迅速な刷新」を参照し てください。

#### RAMPステージ2B

このステップでは、特定のアプリケーショ ン・エリアと画面を個別に刷新します。この 手法は、アプリケーションの根幹部分で使用 します。

5250アプリケーションのDestination画面へのナ ビゲーションを記録し、これらをフレーム ワークにスナップインします。次に、エン ド・ユーザーがデータに効率的にアクセスで きるようにするフィルターを作成します。 拡張ナビゲーションとデータ・アクセスは、 大きな利便性を提供し、刷新されたアプリ ケーションの核となります。詳細について は、「RAMPステージ2B:頻繁に使用する画 面の個別の刷新」を参照してください。

#### RAMPステージ2C

このステップでは、ビデオ、音声、電子メー ル、機能強化されたデスクトップ統合などの 新しい機能や拡張機能を追加して、既存の 5250アプリケーションを選択的に拡張できま す。

追加する内容の量は、利用できる時間の量に 依存しています。詳細については、「RAMPス テージ2C:価値の付加」を参照してくださ い。

## RAMPステージ3



11

RAMPステージ3 (オプション)では、Windows サーバーやLinuxサーバーなど、iSeries以外の サーバー・プラットフォームを使用して実行で きるアプリケーションを作成できます。 ISVの場合、恐らく既存の顧客や新規顧客は自

身に提供される新しいオプションを歓迎しま す。

アプリケーションを多様な第三世代(3GL)言語 で生成し、.NETやWebSphereなどのさまざまな サーバー・プラットフォームで最大の実行効率 を実現できます。

再構築プロセスでは、次のような任意の技術を 使用できます。

• SOA (Service Orientated Architecture)アプ ローチ

 MVC (Mode-View Controller)アプローチ これらのアプローチの選択は任意です。
 このステージは、5250の世界の制限からユー ザーを完全に解放します。
 詳細については、「ステージ3:IBM iプラット フォーム依存関係の排除」を参照してくださ

## 刷新後のアプリケーション

これは、RAMPで刷新された5250人事アプリケーションです。これは、 外観が5250アプリケーションとは異なりますが、次の特徴があります。



例えば、社員の[明細]タブから[資格]タブに移動するには、タブをクリックし、資格5250画面を表示します。5250アプリケーションでF12キーを2回押し、従業員番号を入力し、Enterキーを押す必要があるという画面の 背後にある事実は、ユーザーにはまったくわかりません。

[ビデオ]タブと[Eメール]タブに注意してください。ビデオや電子メール などの機能は、5250アプリケーションで通常利用できませんが、刷新後 のアプリケーションに簡単に追加して、機能と操作性を拡張することが できます。 主な利点

1つのツールと1 つのスキル・ セットを使用す る	RAMPはLANSAの一部です。LANSAの大きな 強みの1つは、開発者が1つのツールと1つのス キル・セットを使用して業務上のすべてのITの 問題を解決できることにあります。1つのスキ ル・セットを持つ1人の開発者は、Windows リッチ・クライアント・アプリケーションから Webブラウザー・アプリケーションまでのソ リューションを作成して、XMLベースのド キュメント統合ソリューションを作成できま す。
フレームワーク は刷新の経路を 定義し、推進す る	RAMPは、アプリケーションの刷新にフレーム ワーク主導の手法を独自に使用します。これは 他に類がありません。次に、フレームワークは 刷新プロセス全体を定義および管理します。こ れは、刷新プロジェクトのさまざまな関係者間 の期待とコミュニケーションを管理する主要な 推進手段です。
非侵入性 - 既存 の5250アプリ ケーションに対 する変更なし	RAMPを使用するために既存の5250アプリケー ションを変更する必要はありません。
<i>DDS</i> なしの場合 でも、すべての 5250アプリケー ションをサポー ト	RAMPでは、Query、DFU、IBM i コマンドなど のDDSのないアプリケーションやオペレーティ ング・システム・コンポーネントであっても 5250ベースのアプリケーションを刷新できま す。
急場しのぎの 「回避的な」ソ リューションや 「問題を無視す る」ソリュー ションではない	RAMPは即時的な短期ソリューションを(ス テージ2で)提供し、次に(ステージ3で)そこから プラットフォーム非依存性へと進む明確な経路 を提供します。RAMPを使用することは、刷新 のための短期的で戦術的なソリューションの作 成と同時に、プラットフォーム非依存性に対す る長期的な戦略の実装というニーズを無視す る、急場しのぎの「回避的な」または「応急処

	置的な」画面スクラッピング・ソリューション ではありません。
配布時間を短縮 する増分的手法	RAMPはビッグ・バン的手法ではありません。 刷新後のアプリケーションの迅速な増分的配布 に対応するように設計されています。これによ り、リスクと市場投入までの時間が最小限度に 抑えられます。
ハードウェアや <i>O/S</i> アップグ レードが不要	RAMPはLANSAでサポートされる任意のプ ラットフォームで動作します。現在、LANSA にはIBM i V5R1以上が必要です。ユーザー (ユーザーがISVの場合、その顧客)は、RAMP で刷新されたアプリケーションをサポートする ためにハードウェアをアップグレードする必要 はありません。
<i>IBM i</i> ソリュー ションを短い時 間で配布	RAMPステージ2全体の焦点は、刷新された iSeriesアプリケーションを可能な限り短時間で 配布することです。アプリケーションを刷新し て可能な限り短時間で元の場所に戻すことは非 常に重要であることを私たちは理解していま す。

#### 5250アプリケーションの変化の過程

このセクションで示しているサンプル・アプリケーションは、3000を超 える5250画面を含む5250 ERP (Enterprise Resource Planning)アプリケー ションの一部です。

これは、LANSA 5250アプリケーションです。ただし、RAMPはLANSA アプリケーションに限定されません。RPGやCOBOLなどの任意の言語で 記述されたiSeries 5250アプリケーションで使用できます。RAMPは、 DFU、Query、IBM i コマンド5250画面など、DDSが存在しないアプリ ケーションでも使用できます。

刷新プロセスの間に5250アプリケーションに対する変更は行われません。

刷新前の5250アプリケーション

RAMPを使用した後の5250アプリケーション

刷新前の5250アプリケーション

従来のナビゲーション技術を操作に使用する5250アプリケーションやグ リーン・スクリーン・アプリケーションについて、少し考えてみましょ う。ユーザーがサインオンし、メニューが表示されます。次に、操作す るアイテムを特定することが必要になるまで、オプションを選択し、複 数のメニューを経由して移動します。



## RAMPを使用した後の5250アプリケーション RAMPを使用した後のWindowsリッチ・クライアント・アプリケーショ ンと同じ5250 ERPアプリケーションを以下に示します。

Manual Inventory Objects				<u>ר</u>	フィルター(	こより、検索基準に一致する
File Edit View Actions Tools Help	telp (Framework) ユーザーは、フレームワークのフィルターを 在庫アイテムのリストが作成されている			のリストが作成されています。		
😮 New 👻 🂥 Sign Off	Detais 使用	、、在庫番号に基づい	て在庫アイテムを			
検索します。 エンド・ユーザーは、操作するアイテムを					ザーは、操作するアイテムを	
On Tool Bar	Find by Key	End hu Other	Item	Description	IJ2	ストから選択します。
Inventory (Classic)		THU DY OUTS	A 100BALLS	A 100BALLS	Infiniti Golf Balls	
Items	by Item Code		■ A 100CAP	A 100CAP - 1	nfiniti Golf Cap	L
Inventory Objects	Item Number	/	€ A1000, 808	A 100CAPBOE	Infiniti Golf Cap with B Infiniti Hand Course	Bob Logo
E Titems	A 100		A 100COVER	A100COVER	fintit 9" Staff Bag	
Transactions			A 100PLAQUE	A 100PLAQUE	- Infiniti Golf Plaque	
Transfer Orders		Sear	ch E A100SHIRT-L	A 100SHIRT-L	- Infiniti Golf Shirt Large	
Lots					Clear List	
Locations	Item : Edit (A 100BA)	LS-A 100BALLS - Infiniti Golf B	alls)			•
🖉 Plants	🛄 Details 🥖 Edit	👺 Inventory Balances 🛛 🗯	Inventory Planning 🛛 🔁	Multilingual Captions	Notes	
all in the second seco	Screen Mode	CHANGE A 100BALLS				Next Prompt
	Description	Infiniti Golf Balls				
	Extra Descr	Extra Description test				
	Short Descr				7	ナップインされた5250画面は、
52507=1775	Item Class	CLASS Item	Type PUR	Item Group	o Code 在庫:	アイテムの詳細を保持するために
は、アプリケーションの	Item Division	Cntry	y of Origin		10000	使用されます。
異なる部分間のナビゲー	Tax Class					Longer of 7 B
ションに使用するアプリ	Manufactured?	0 (0/1) Non-:	Stock Item?	(0/1)		
ケーション・ツリーに	Group rech code	Diane				
置き換えられました。	Supplier ID					
	Rebates Flag	(0/1) Item	Status 1	Omit from	Web	
	Ship-From Plant		-		_	
	Unit of Measure	EA Dft P	urchase UM SL	Dft Sales L	JM DZ	
	Pricing U/M	DZ Itm C	Catalog No			
	Res. Time Fence					
	Univ Prod Code					
			(	Messages Ready	LANSA USA ENG	C_LPC 12/15/05 16:10

## 刷新されたアプリケーションのデータにアクセスする

フレームワークのフィルターは、エンド・ユーザーにアプリケーション のデータへの効率的で柔軟性の高いアクセスを提供するため、刷新され たアプリケーションに価値を付加する最も強力な方法の1つです。



## 刷新されたアプリケーションでのナビゲーション

RAMPアプリケーションでは、5250画面でWindowsアプリケーションが 実行されるのではなく、Windowsインターフェースによって5250画面が 実行されます。

RAMPが5250画面のスクラッピング・テクノロジの従来の形式とは異なり、非常に強力な機能を備えているのは、Windowsアプリケーションが 実行しているか、5250アプリケーションが実行しているかの違いによる ものです。

Inventory Objects								
File Edit View Actions Tools Help	(Framework) (Admi	nistration )						
Inventory Objects         File Edit View Actions Tools Help         New ▼ Sign Off         Items         On Tool Bar         Inventory (Classic)         Items         Items         Items         Inventory Objects         Inventory Objects         Inventory Objects         Inventory Objects         Items         Itens         Warehouses	(Framework) (Admi ユーザー クリックす 下部にある 関連する詳細 Detais / Edit Screen Mode Item Number Description Exits Descr	nstration) が在庫アイテム ると、ウィンドウロ 5250画面が変オ 情報が表示され ● Inventory Balanc CHANGE NIDOCOVER Infinit Head Covers Infinit Head Covers Infinit Head Covers	を の りし、 ます。 Enventory Plan	008ALLS 00CAP CAPPOB 00CAP 00CAPUE 00CAQUE 00CAQUE 00CAQUE 00CAQUE	Description A 1008ALLS - Infinit Golf A 100CAP-DB - Infinit Golf A 100CAP-DB - Infinit Golf A 100CAP-DB - Infinit Golf A 100B - Infinit Golf A 100B - Infinit Golf A 100SHIRT-L - Infinit Go C C C C C C C C C C C C C C C C C C C	Balls ap of Copers of Govers If Bag If Plaque If Shirt Large lear List	Next Prompt	
	Short Descr Item Class Item Division Tax Class Manufactured? Group Tech Code Buyer Suppler ID Rebates Flag Ship-From Plant Unit of Measure Pricing U/M Res. Time Fence Univ Prod Code	CLASS 0 (0/1) 13 (0/1) EA EA	Item Type Cntry of Origin Non-Stock Item? Planner Item Status Dft Purchase UM Item Catalog No	PUR 0 (0/1) 1 EA INFCAP	Item Group Code Omit from Web Dft Sales UM	ACCESS		
				Messages	Ready LANSA	USA ENG C_LPC	12/15/05 16:11	
								_

Inventory Objects				Chiefe Franker Franker
File Edit View Actions Tools Help (Framew	ork) (Administration)			
😮 New 👻 🎽 Sign Off 🛛 🔲 Deta	sis 🥃 Calculator			
Iter				
選択した在庫アイ 表示したい場合は、[ ノートを表示する52 刷新されていないアプ 2回使用し、メニュー アイテム番号を再度)	(テムに関連するノートを ノート]タブをクリックすると、 250画面が表示されます。 リケーションでは、F12キーを からアイテムを選択して、 入力する必要がありました。	Item 6 A 1008ALLS 8 A 100CAP 6 A 100CAP806 6 A 100COVER 8 A 100C0VER 8 A 100C8 9 A 100C8 9 A 100C8 9 A 100C8 9 A 100C8 9 A 100C8 9 A 100C9 9 A 100	Description A1008ALLS - Infinit Golf Balls A100CAP - Infinit Golf Cap A100CAPBOB - Infinit Golf Cap with Bob Logo A100CCVER - Infinit Golf Cap with Bob Logo A100CVER - Infinit Golf Plaque A100CHAQLE - Infinit Golf Plaque A100CHAQLE - Infinit Golf Planturace Clear List gual Caption	
B S S S S S S S S S S S S S S S S S S S	Aumber A100COVER ton Code age Code ENG : : 2-Change 3-Dates 4-Delete 10-Attac	. Infiniti Head Covers		OK Print Add Note Add Msg
Act	Code Note Id Description			
	DESC INFINITI HEADCOVERS			
	Playing in style. These logo he	adcovers will keep your		
	INFINITI equipment looking its	s best.		
	Available: X, 1, 3, 5			
	DEE			
	GHI		<u>II</u>	
V	VEB DESC INFINITI HEADCOVERS			
v	VEB Playing in style. These logo he	adcovers will keep your		
V	VEB INFINITI equipment looking its	s best.		
V	VEB Available: X, 1, 3, 5			
V	VEB ABC			
		Messag	es Ready DANSA USA ENG C_LPC	12/15/05 16:11
A CONTRACTOR OF A CONTRACTOR O				

1 Inventory Objects	
File Edit View Actions Tools Help (Framework) (Admi	nistration )
😳 New 🔻 🂥 Sign Off 🛛 Details	Calculator
Items	
	Para Paradation
Un	Item Description Colfeet Colfeet
Inventory	ADDORLD ADDORLD ADDORLD ADDORLD ADDORLD ADDOR
■      Inventory Windowsアプリケーションの場合	全日様に、 A 100CAPBOB A100CAPBOB - Infiniti Golf Cap with Bob Logo
	H A 100 C L BORD
使用して既存のアイテムをコピ	E A 100 E Edt L Trans Gold Shet Large
第しい在庫アイテムを作成で	きます。 Search Gradow Charles Clear List
	Inventory Banning
🥔 La	Inthit Head Cover)     Mitting and Capitage
	ances 🐤 Inventory Planning 🐨 Heatingean Capitonis 💽 Notes
WA CHAR	
a lor other	Vertrage V
Item Number	A100COVER Infiniti Head
Attention Code	Position  Add Note
Language Code	ENG
Action: 2=Change	3=Dates 4=Delete 10=Attach P/A
P/A Men/	
Act Code Note Id	Description
DESC	INFINITI HEADCOVERS
	Playing in style. These logo headcovers will keep your
	INFDNITI equipment looking its best.
	Avaiable: X, 1, 3, 5
	ABC
	DEF
	GHI
WEB DESC	INFINITI HEADCOVERS
WEB	Playing in style. These logo headcovers will keep your
WEB	IVFIDITI equipment looking its best.
WEB	Available: X, 1, 3, 5
WEB	ABC
	Manager Davids LANSA USH Exc. C.D.C. 151/15/05 156-11 (2)

RAMPedアプリケーションは外観が優れているだけでなく、使いやす く、取り組みが容易で、生産的であり、新しいユーザーでも簡単に学ぶ ことができます。

5250インターフェースでは実現できない、電子メール、音声、ビデオ、 Webブラウザ・インターフェース、デスクトップ統合機能など、新しい 機能を追加することで、刷新されたアプリケーションを容易に拡張でき ます。

## 刷新プロセスの概要

ステージ1:刷新フレームワークの作成 ステージ2:5250アプリケーションの増分的な刷新 ステージ3:IBM iプラットフォーム依存関係の排除

## ステージ1:刷新フレームワークの作成

RAMPを使用するには、最初に、完了したときの刷新されたアプリケー ションがどうなるかを示す実行可能なプロトタイプを必ず作成します。 このプロトタイプは非常に重要です。これは、開発者やエンド・ユー ザーなどのすべての関係者が、結果として大きな投資を行う前に、実現 する内容や実行する必要がある内容を理解することに役立つためです。 幸い、RAMPを使用すると、アプリケーションのプロトタイプ作成を非 常に簡単かつ迅速に行うことができます。



必要なもの

必要なもの

プロトタイプ作成には、経営者、ユーザー、およびIT担当者の効果的な コミュニケーションが必要です。

経営者とユーザーをプロジェクトの推進者とする伝統的な概念はここで も強力に機能します。プロジェクト推進者がいない場合、ITグループは プロジェクトの実際の価値を特定したり、これらを追加するうえで常に 苦悩することになります。

プロジェクト推進者に質問して、追加される最も重要な実際の価値の上 位5件を特定して設計し、これらを完全に実装することで、プロジェクト に対する責務が完全に明確になり、ラジオ・ボタン、ドロップ・ダウ ン、ツリーなどのありふれた価値のアイテムに没頭することなく、グ ループとして実際の価値を明確に考える利点が強化されることもありま す。

この段階では、「価値を付加するもの」のトピックを参照する必要があります。

## プロトタイプの作成

プロトタイプ作成プロセスは、非常に迅速であるため、比較および協議 できる複数のプロトタイプを生成できます。



Visual LANSAフレームワークは、業務用アプリケーションを定義および 実行するためのフレームワークです。

メイン・ウィンドウは、MS-Outlookまたは「ダッシュボード」スタイル で配置されます。詳細については、『Visual LANSA フレームワーク ガ イド』の「重要な概念」を参照してください。

設計者ばかりでなくエンド・ユーザーも、これらのさまざまなフォーム 領域の場所および配置方法を個人的な環境設定に従って、いくつかの方 法で変更できます。『Visual LANSA フレームワーク ガイド』の「ウィ ンドウ・レイアウトの調整」を参照してください。

🕈 お気に入り		
ファイル(E) 編集(E) 表示(V) ヘルプ(H) ウィントウ(W)	(フレームワーク)(管理)	
	(新規作成)	▶ (₀)検索
お気に入り	<ul> <li>(アウリケーション)</li> <li>(アウリケーション)</li> <li>(フマンド)</li> <li>(メニュー)</li> <li>(デ・ザ・インコードテーフ・ル)</li> <li>(コード・ウィザード)</li> <li>(コード・ウィザード)</li> <li>(インスタント フロトタイピック・アシスタント)</li> <li>(RAMP ツール)</li> <li>(RAMP ツール)</li> <li>(RAMP ツール)</li> <li>(RAMP ツール)</li> <li>(保存)</li> <li>(保存)</li> <li>(名前を付けて保存)</li> <li>(保存して再起動)</li> <li>(保存してあどう)</li> <li>(Webアフリケーションとして実行)</li> <li>(Web コンソール)</li> </ul>	ブロトタイプ作成を開始する には、「インスタントプロト タイピングアシスタント] メニュー・オプションを 選択します。
🗉 🖽 📔 👘 👘	(14-2)	• 0/07/02 9:27 🥥

メインビジネスオフジェクト: Customers,Products,Orders,Code Tables	
前回の値を回復	
Customers	[インスタントプロトタイピングアシスタント]メニューでは、まず設計者は
a Orders	アプリケーション(この場合は、ERPアプリケーション)の一部となる
Code Tables	ビジネス・オブジェクトを指定します。
	ビジネス・オブジェクトは、エンド・ユーザーが操作するものです (何らかの特殊な00またはIT用語ではありません)。 ここでは、願客、製品、および受注を指します。
	次へ >> <b>キャンセル</b>







次に、[終了]ボタンをクリックすると、プロトタイプ・アプリケーションが生成されます。インスタント プロトタイピング アシスタントを閉じると、Visual LANSAフレームワークのメイン・ウィンドウが再度表示されます。

# プロトタイプの実行と洗練化

#### プロトタイプERPアプリケーションが作成され、フレームワークに追加 されました。













プロトタイプ・アプリケーションが適切な場合は、そのさまざまなパー ツを実際のアプリケーションとして実装し、徐々にプロトタイプ・パー ツをスナップアウトして、実際のアプリケーション・パーツをスナップ インすることができます。これにより、プロトタイプは時間の経過とと もに実際のアプリケーションへと進化します。



アプリケーションのプロトタイプを作成する方法の詳細については、 チュートリアル「RAMP-TS001:刷新されたアプリケーションの基本的 なプロトタイプの作成」を参照してください。
#### ステージ2:5250アプリケーションの増分的な刷新

RAMPは、増分的で発展的なアプローチを使用して、次世代のアプリ ケーションを作成することにより、リスクと市場投入時間の問題を管理 する実際的な方法を提供するように設計されています。つまり、RAMP アーキテクチャはビッグ・バン的なアプローチではないため、開発者は アプリケーションの複数の小規模なバージョンを出荷して、リスクや市 場投入時間を最小限に抑えることができます。



5250画面の命名

画面の追跡と分類

RAMPステージ2A:アプリケーション・セグメント全体の迅速な刷新 RAMPステージ2B:頻繁に使用する画面の個別の刷新 RAMPステージ2C:価値の付加

## 5250画面に名前を付ける

5250画面をフレームワークにスナップインするための最初の手順は、 5250画面に名前を付けることです。

🐈 お気に入り			
ファイル(E) 編集(E) 表示(⊻) ヘルプ(	±) ウィンドウ(₩)	(フレームワーク)(管理)	
③作成 🗇 🖨 印刷 🐴	DSAB	(新規作成)	▶ "勿検索
お気に入り x ERP Prototype 1 HR テ*モアフ*リケーション Human Relation Wy Application 方気に入り * ③ フ*ロヴラミング テウニック * ④ 管理		<ul> <li>(フロハティ)</li> <li>(アフツケーション)</li> <li>(コマント<sup>*</sup>)</li> <li>(メニュー)</li> <li>(テッサインコート・デーフジル)</li> <li>(コート・ウィザート<sup>*</sup>)</li> <li>(インスタント フロトタイピックケ アシスタント)</li> <li>(RAMP ツール)</li> </ul>	5250画面の命名を 開始するには、 RAMPツールを起動します。
		<ul> <li>(仮想ワリップポード)</li> <li>(マージ ツール)</li> <li>(保存)</li> <li>(名前を付けて保存)</li> <li>(保存して再起動)</li> <li>(保存して終了)</li> <li>(Webアプツケーションとして実行)</li> <li>(Web コンソール)</li> <li>(ヘルプ)</li> </ul>	
≡ ⊞	<i>わ</i> を-ジ	(\\-,z)	0/07/02 13:02







RAMP	ツール - デウォルト セッション(RAMP-TS)		
7 ◀ ▶	RAMP-TS 5250 セッション   詳細		
名.	下のメータージをクリークし、可能なアクションを見て下さい 入フォームを命名 i50SMainMenu 入フームワークに定義されていません。 ーーをAU_AL て UNMでTIONとL てーのフォー	名前情報を保存したら、 ボタンをクリックして画面の 名前を付ける領域を閉じます。	>
<u>~8.</u> 	検証       再スタート       ジ       ナビヴィーション スグリフドの自動         Keymaps       AutoGUI       Screens         A.QDUI132.USRRCD       名前:         150SMainMenu       (ID = 12)         第名:       ジ         デジライム:       ン         開給       珠了       名前         150SMainMenu       (ID = 12)         第名:       ジ         アン・AL:       シ         150SMainMenu       (ID = 12)         第名:       ジ         アン・AL:       シ         150SMainMenu       (ID = 12)         第名:       ジ         アン・AL*:       Lock Screen         名前       行<汚	Display       Help         Image: Session       Display       Help         Image: Session       Image: Session       Image: Session         Image:	Tracking Info MainLogin ↓ Enter I5OSMainMenu



 画面に名前を付ける方法の詳細については、チュートリアル「RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新」および
 「RAMP-TS004:従業員画面の命名と分類」を参照してください。

### 画面の追跡と分類

刷新されたアプリケーションで再使用する画面やフィールドに名前を付けたら、これらの分類とこれらの間のナビゲーションの追跡に進むことができます。

	? ツール - デウォルト セッション(RAMP-TS)	
7*.∢ ►	RAMP-TS 5250 セッション 詳細	
	下のメータヒージをクリークし、可能なアクションを見て下さい フォームを命名 MainLogin として定義 Junction このフォームに名前付きのアイテムがありません このスクリーンへだ゙ゲートできるスクリーン・・・ このスクリーンがだ゙ゲートできるスクリーン・・・	ここで、フレームワークにスナップインされる画面にRAMPで到達 する方法の説明に進みます。 これを実行するには、さまざまな5250画面に移動し、刷新された アプリケーション内でどのように使用するかをフレームワークに 指示するために、表示された画面を分類する必要があります。
	く 検証 スナッフジョット (622×334) 再スタート	5250サインオン画面はこのルールの唯一の例外で、Junction 画面としてあらかじめ分類されています。
	Session Display Help User: Password: Reconnect: Cancel Login Advanced	- Auto - V C C Tracking Info MainLogin
	aXesTS Signon- Terminal closed	



🗌 RA	MP ウール - デウォルト セッション(RAMP-TS)	
	RAMP-TS 5250 セッション 副羊綿田	
	下のメーシヒージをクリークし、可能なアクションを見て下さい フォームを命名 i50SMainMenu として定義 Junction このフォームの名前つきのアイテム: utxtSelectionOrCmd (input) このスクリーンへナビゲートできるスクリーン・・・ MainLogin - Login Form このスクリーンがナビゲートできるスクリーン・・・ Milleria Login Form このスクリーンがナビゲートできるスクリーン・・・ Milleria Login Form	× <
	< Session Display Help 🔎 - Auto - 💌 🔍 🥘	Tracking Info
	MAXN       SYSTEM I メイソメニュー         システム: AS1002         次の1つを選択してください。         1. ユーザー・ウスカ         2. オフィス・クスカ         3. 沢用システム・ウスカ         3. 沢用システム・ウスカ         3. 沢用システム・ウスカ         3. 沢用システム・ウスカ         5. プログラミング         5. プログラミング         6. 潤傷         9. メニューの云布         10. 滑星後島 オブション         11. SYSTEM I ACCESS のクスク         90. サインオフ         温枳長目またはコマンド         「副本、FULL RY For Partition(dem)         ア3- 約月 知中・の放光         (2) CONVATOT ISM CORP. 1980, 2007.         ※ ※ ②	MainLogin ↓ Enter 150SMainMen u











RAMPステージ2A:アプリケーション・セグメント全体の迅 速な刷新

プロトタイプ作成ステージの間に、たまにしか使用されないアプリケー ションのセグメントを特定し、これらのセグメントのエントリー・ポイ ント画面へのナビゲーションに名前を付けて、コレオグラフ化しまし た。ここでは、これらをフレームワークにスナップインします。



#### 方法

方法

5250エントリー・ポイント画面に名前を付けて、これらの画面とのナビ ゲーションを記録すると、これらはフレームワークにスナップインされ ます。



チュートリアル「RAMP-TS002:アプリケーション・セグメント全体の 迅速な刷新」を参照してください。

#### RAMPステージ2B:頻繁に使用する画面の個別の刷新 プロトタイプ作成ステージの間に、よく使用されるアプリケーションの セグメントを特定し、これらのセグメントのDestination画面へのナビ ゲーションに名前を付けて、コレオグラフ化しました。ここでは、これ らをフレームワークにスナップインし、フィルターを作成して、データ へのフレキシブルで効率的なアクセスを提供します。



方法

方法

以下のチュートリアルを参照してください。

- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更

## RAMPステージ2C:価値の付加

このステップでは、ビデオ、音声、電子メール、機能強化されたデスク トップ統合など、新しい機能や拡張された機能を追加して、既存の5250 アプリケーションを選択的に拡張します。



新しいVisual LANSAコンポーネントを追加して、最新のIT技術の適用に よりビジネス・プロセスを向上するまったく新しい方法を提供します。 価値を付加するもの 方法 価値を付加するもの

刷新されたアプリケーションに真のビジネス価値を付加しなければ、プロジェクトは成功しない可能性があります。

ビジネスの成功や費用の節約をもたらすものは価値を付加します。これ らは、ビジネス・プロセスや生産性の向上、訓練費用の削減からスタッ フの満足まで多岐にわたります。同様に、エンド・ユーザーがジョブを 迅速に、適切かつ賢明に、高い個人的な満足度で実行できるものは価値 を付加します。

ツリーを使用して、5250メニュー・ナビゲーションを複数のラジオ・ボ タンやドロップ・ダウンに置き換えるなど、ありふれた価値のアイテム を導入しても、エンド・ユーザーにとっては負の価値にしかならないこ とがあります。

真の価値を既存の5250アプリケーションに付加するには、以下のような いくつかの方法があります。

洗練された強力 なフィルターの 使用によるデー タへのアクセス	フィルターは既存の5250アプリケーションで は実行できないことを簡単に実行可能で、一 般的なエンド・ユーザー・ビジネス・プロセ スに正確に合わせて調整できるため、価値を 付加します。
	例えば、基本的な製品照会の5250画面を使用 し、製品マスター・ファイルや発注詳細ファ イルに対して強力なフィルターを追加した場 合、本日の発注、在庫切れによる発注のすべ ての製品の検索、名前による検索、カテゴリ による検索などを行えるようにすることで、 エンド・ユーザーの日常業務に多くの価値を 付加することができます。
	また、エンド・ユーザーが独自の頻繁に使用 されるカスタム検索を定義できるフィルター (毎週月曜日に前の週に売れたすべての製品を 表示するなど)や一般的に使用するリストを動 的に作成するフィルター(すべての製品が有害 と示されるなど)も大きな真の価値を付加しま す。 5250アプリケーションを単に模倣するフィル ターは、多くの価値を付加しません

情報の集約	5250アプリケーションでは、倉庫に勤務して いる社員と事務所に勤務している社員が複数 のソースから情報を抽出し、それを管理レ ポート内に集約したり、異種の非統合アプリ ケーション内への入力に使用することが必要 になることがあります。 これらのタスクを1、2回のクリックで単一の フォームに集約できる場合は、経営者やユー ザーに対して真の価値を付加します。
デスクトップ・ アプリケーショ ンとの統合	ユーザーと顧客のコミュニケーションを促進 する電子メールやレポート作成および解析タ スクを促進するMS-Excelなどの機能を統合す ることで、価値を付加します。
繰り返しおよび キーの再設定の 削減	多くの5250アプリケーションでは、基盤の 5250アプリケーションが統合されていないた め、ユーザーによる情報のキーの再設定がま だ必要です。 これを削減または解消できる場合、経営者や エンド・ユーザーに価値を付加します。ま た、テンプレートのユーザー定義可能なリス トからの5250画面の洗練されたプロンプトや 事前入力も価値を付加する適切な方法です。

これまで説明した内容は、最も効果的なビジネス・プロセスの再構築の 根底にある簡素な例です。これは、すでに実現されていることであり、 これらを新しい改良された方法で再使用してください。

これらすべての点は、「ステージ1:刷新フレームワークの作成」で考慮 する必要があります。 方法

以下のチュートリアルを参照してください。

- RAMP-TS010:特別なフィールド・ハンドラーの使用
- RAMP-TS011:付属のドキュメント・コマンド・ハンドラーのス ナップイン
- RAMP-TS012: 付属のNotesコマンド・ハンドラーのスナップイン
- RAMP-TS014:HRデモ・アプリケーションへのRAMP画面のスナッ プイン
- RAMP-TSAD01:ボタンを使ったアプリケーションへの価値の付加
- RAMP-TSAD03:特別なフィールド受け渡し

### ステージ3:IBM iプラットフォーム依存関係の排除

RAMPステージ3は、iSeries以外のサーバー・プラットフォームをサポー トする必要がない場合のオプションのステージです。このステージの最 終的な目標は、すべての5250 Destination画面をプラットフォームに依存 しないVisual LANSAコンポーネントに置き換えることです。つまり、こ れには既存の5250アプリケーションの再構築が必要になります。

dia Products		
<u>File E</u> dit <u>V</u> iew <u>H</u> elp <u>W</u> indows (Fr	amework) (Administration)	RAMPステージ3では、すべての5250自由は
	SPAMELO	Visual LANSAコンホーネントに直さ換えられ、
		これによりIBM「ノノリトノオームの私行則係か
Products		HINTCAUGE 9 6
X		アプリケーションでは、LANSA、Visual LANSA
🖃 💕 ERP Prototype 1	By Delivery Date By	フレームワーク、およびWindowsデスクトップの
Code Tables	By Product Name By	すべての機能を活用します。
Customers	Duad unt Mana	
		Search IN ATI GOLF BALLS
E 🔶 Favorites	1	INFINITI GOLF CAP WITH BALL LOGO
🗉 藵 HR Demo Application	🔽 Clear List	INFINITI HEADCOVERS
🗄 👰 Programming Techniques		INFINITY STAFF DAG
🗄 💁 Administration		x
🗄 💕 RAMP Examples	Product : Details INFINI	ITI GOLF CAP WITH BALL LOGO
	📃 🗖 Details 🛛 💊 Notes 🛛 📥 R	Print 💷 Sales
	Product Code	A007 Save
	Product C ISND	366 S Delete
	Category	AC/65
	Product Division	INFINITI
	Supplier	SP 45
	Univ Product Code	
	Extra Description	LOGO PER ORDER
	Dricing LIM	0
	Pricing OM	·
E 🗄 🔒	Messages Ready VL	FPGMLIB ENG VLFPGMLIB 29/03/09 13:30 🥥

これにより、以下のような大きな利点を得ることができます。

- まず、アプリケーション全体がRPG.COBOL、DDS、またはiSeries やi5サーバーを使用して実行されているものに依存しなくなります。 これは、WindowsサーバーやLinuxサーバーを使いたい顧客にアプリ ケーションを販売する必要がある場合は非常に重要になることがあり ます。
- LANSAでサポートされる技術やアプローチを使用して、再構築を 計画することができます(例:SOA、MVC)。

- ステージ3は、市場投入までの時間を短縮し、リスクを低減する、
   迅速な複数増分配布アプローチをサポートします。
- ステージ3は、LANSA、Visual LANSAフレームワーク、Windowsデ スクトップ、およびWebブラウザのすべての機能をアプリケーション に開放します。
- ステージ3により、RAMPはアプリケーションを刷新する長期的な 戦略的アプローチとなります。これは、急場しのぎの「回避的な」ソ リューションではなく、将来へと向かう明確に定義された方向性を提 供します。

前提スキル

RAMPを使用するには、LANSAとVisual LANSAフレームワークを使用し てアプリケーションを開発する方法について、いくつかの基本的な知識 を持っている必要があります。以下のチュートリアルのいくつかを確認 することもできます(以下のリンクは他のガイドを指しています)。

- フレームワークのチュートリアル
- Visual LANSAのチュートリアル System iサーバーでVisual LANSA エディターを使用してプログラムを作成、コンパイル、確認する方法 を理解する必要があります。

#### 該当する専門家

RAMPを使用してアプリケーションを刷新するには、刷新するビジネ ス・アプリケーションおよびそのアプリケーションを実行する業種につ いての詳しい知識を持った人を利用できるようにする必要があります。 またこの人は、刷新のアウトプットとして、ビジネスが達成すべき目的 についても理解している必要があります。

どのようなツールを使っても、該当の専門家に相談せずに、アプリケー ションの刷新を成功させることは不可能です。その理由は次のとおりで す。

- 既存の機能がどのように使用されているかを誰も知らない(どのように動作するかのみを理解していることとは対照的です。)
- 刷新したバージョンをどのように使用するかについてのビジョンを 描く人がいない(どのように動作すべきかとは対照的)。

警告と免責

賢明さとリスクのジレンマ

RAMP-TSを進めると、VLFおよびRAMP-TSサーバーがJavaScript、 HTML DOMなどで公開される内部処理モデルを使用していることにお気 付きになるでしょう。組み込みのコードを確認したり、モデルを調査す るすべての種類のツールを使用することで、これを実行できます。

これにより、次のような「賢明さとリスク」のジレンマが生じることが あります。

ユーザーは、フレームワークやRAMP-TSの内部処理モデルを解析、変 更、再使用、または変更する賢明な論理を記述できるとします。また、 これを行う正当な理由があり、非常に役に立つ強力な結果を得ることが できるかもしれません。

ただし、これを行うことにより、非常に明確なリスクにさらされます。 このリスクとは、フレームワークやRAMP-TSの将来のバージョンで内部 モデルの動作や内容が変更される可能性があることです。これらの変更 により、内部モデルの使用時にその実用性が損なわれたり、機能しなく なったり、損傷を受けることがあります。

製品サポートとメンテナンスの観点から、LANSAは公開済みのインター フェース(本ガイドまたはVLFガイドでドキュメント化されている内容) をサポートします。

フレームワークやRAMP-TSの内部処理モデルを解析、変更、再使用、または変更する場合、ユーザーはリスクを負っていること、ユーザーやその勤務先の組織がその費用を負担しなければならないことに注意する必要があります。

このような操作を行うことは何ら問題ありませんが、関連するリスクと その後の結果の責任者について明確に示す必要があります。

# 最新情報

RAMPの当バージョンではLANSA Version13をサポートしています。

以前の バージョンでの新しい機能を確認するには、次のセクションを参照してください。

RAMP EPC868の新機能 RAMP EPC843の新機能

## RAMP-TS EPC868の新機能

このセクションでは、RAMP-TSのEPC868バージョンの新機能を説明します。

RAMP-TSとRAMP-NLスクリプトで更に多くの情報にアクセスできるようになりました。

objCommand オブジェクトには、現在のコマンドに関する任意の引数の 詳細やそのコマンドが実行される理由も含まれるようになりました。 (つまり、コマンドの実行なのかコマンドが有効化されたかの区別が つくようになりました。)詳細はobjCommandの定義を参照してください。

新しいファンクション

- LOCK\_FRAMEWORK ファンクション
- UNLOCK\_FRAMEWORK ファンクション
- RESTART\_LAST\_NAVIGATION ファンクション

以前のRAMPバージョンでの新しい機能を確認するには、次のセクショ ンを参照してください。

RAMPのEPC 843バージョンの新機能

## RAMP-TS EPC843の新機能

このセクションでは、RAMP-TSのEPC843バージョンの新機能を概説します。

#### RAMP-TSのSSLサポート

SSL (Secure Socket Layer)のサ ポートがRAMP-TSに追加され ました。

[サーバーの詳細]タブの
 [RAMP-TS (ターミナル サーバー)]グ
 ループ・ボックスの[HTTPSを
 使用]オプションをオンにし
 て、SSLでRAMP-TSを起動し
 ます。「HTTPSを使用」を参照してください。
 注: VLF-WEBまたはVLF-NET

アプリケーションからRAMP-TSでSSLを使用するには、VLF WebサーバーもSSLで動作して いる必要があります。

仮想クリップボードに保管されている値が現在の実行を越 えて存続しないようにする

AVSAVEVALUEファンクショ ンの新しいパラメータが追加 されました。このパラメータ をFALSEに設定すると、VLF仮 想クリップボードに保管され ている値はRAMP-TSアプリ ケーションの以降の呼び出し で存続しないようにすること ができます。デフォルト値は TRUEです。

RAMP-TSコマンド・ハンドラーから 他のビジネス・オブジェクトへの切り 替え

新しいAVSWITCHファンクションを使用すると、現在の画面から別のビジネス・オブジェクトに切り替えて、指定したコマンドを実行できます。これは、VLF Switchメソッドの基本的な実装です。

SHAREDオブジェクトのプライベー ト・バージョン

RAMP-TS画面がプライベート定義フォ ルダに維持されている場合、SHARED オブジェクトのプライベート・バー ジョンを使用できるようになりまし た。[RAMP-TS(ターミナルサーバー)]グルー プ・ボックスの[共有打ジェクトを含む] オプションをオンにして、SHAREDオ ブジェクト・ファイルがプライベート 定義フォルダにあることをRAMP-TSに 示します。「共有打ジェクトを含む」を 参照してください。

注:プライベート定義フォルダが指定 されていない場合、デフォルトの場所 からSHAREDオブジェクト・ファイル がロードされます。

サブファイルをテーブルとし て認識するときのコントロー ル

新しいOVRSFLAUTOGUIファ ンクションを使用すると、 Axesシステム・フラグの[サブファ イルテーブルの認識]を画面ごとにオ ン、オフすることができま す。

# ライセンス要件

開発者簡易チェック・リスト RAMP開発者は、この簡易チェック・リストを使用してください。

チェッ ク済み	ライセンスの タイプ	許諾される 内容	確認方法
	VL-IDEダング ルまたはソフ トウェア キー・ライセ ンス	Visual LANSAアプ リケーショ ンの開発	Visual LANSA (VL)アプリケー ションを作成、コンパイル、実 行でき、そのアプリケーション をSystem iサーバーにチェックイ ンできるかどうかを確認しま す。
	RAMPコレオ グラファ・ラ イセンス	RAMP画面 のコレオグ ラフ	System iサーバーでLANSA REQUEST(LICENSE)を使用し て、有効な"AXX"または"Ann"ラ イセンスを検索します。
	Axes-TSライセ ンス	Axes-TSサー バー経由で の5250セッ ションの表 示	IBM i コマンドWRKLICINF PRDID('1AXES*')を使用します。 アスタリスクに注意してください。 インストールされている機能に よっては、結果の画面に次の機 能が表示されることがありま す。Base、Terminal Server、 Application Server、Spooler File Server、Intersession、Terminal Serverなど。
	LANSAスー パー・サー バー・ライセ ンス	5250以外の インター フェースを 使用した、 System i上の リモート・ データおよ	System iサーバーでLANSA REQUEST(LICENSE)を使用し て、有効な"LXX"または"Xnn"ラ イセンスを検索します。

	びプログラ	
	ムへのアク	
	セス	

ライセンスを取得する場合は、LANSAの製品ベンダーにお問い合わせください。

# インストールおよび構成

サーバー上でのインストール バックアップの方針 構成 多数の開発者が同じアプリケーションで作業する場合 サーバー上でのインストール

フレームワークのインストールの詳細については、『Visual LANSAフレームワークガイド』を参照してください。 必要な操作は次のとおりです。

- まず最初にSystem iサーバーにフレームワーク・ソフトウェアをイン ストールします。このステップだけを実行してください。LANSA for the Webをインストールしているわけではないので、『Visual LANSA フレームワーク ガイド』の他のステップを実行する必要はありません。
- 次にVisual LANSAワークステーションにフレームワークをインストールし設定します。
- そして、IBM iサーバー上でのRAMP-TS / aXes-TSのインストールと構成を行います。
- 最後にRAMP-TS/aXes-TSインストールの確認をします。
- バックアップの方針。

RAMP-NL (newlook)を使用したことがある場合は、「RAMP-TSまたは RAMP-NLとのフレームワークの関連付けについて」というよくある質問 も参照してみてください。 IBM iサーバー上でのRAMP-TS / aXes-TSのインストールと構成

RAMP-TSを使用する前に、aXesターミナル・サーバーをIBM iサーバー にインストールして構成する必要があります。

インストールの前に

インストールを開始する前に、『aXesQuickStart ガイド』の計画に関するセクションを参照してください。

- この製品がすでにインストールされている場合は、アップグレードの 方法について『aXesQuickStartガイド』を参照するか、ベンダーに問 い合わせてください(aXesがインストールされているかどうかを確認す るには、GO LICPGMコマンドを実行し、インストールされているラ イセンス・プログラムを表示して製品(1AXES01)を探します)。
- 2. aXesがインストールされていない場合は、www.axeslive.comでダウン ロードをクリックして、ダウンロードします。
- 3. 『aXesQuickStart ガイド』の手順に従って、aXesをインストールします。
#### RAMP-TS/aXes-TSインストールの確認

RAMP-TS/aXesをIBM iサーバーにインストールした後、インストールを 確認する必要があります。

以下の確認ステップが問題なく完了するまで、RAMP-TSの使用を開始し ないでください。

1. ブラウザー・ウィンドウを開き、以下の形式でURLを入力します。 ここで、hostとport numberには、aXes-WSサーバーのホストとポート を指定します。

http://<host>:<port number>

次のような画面が表示されるはずです。

axes デフォルトページ このデフォルトページはaXesアプリケーションサーバーによって表示されています。 aXesのホームページへ進むには「ここ」をクリックして下さい。 Contents © LANSA Group. All rights reserved.

2. aXesホーム・ページのリンクをクリックします。 次のように表示されるはずです。



 [aXesセッションの開始]の右側の[ターミナル・セッション]オプションをクリックします。通常のユーザー・プロファイルとパスワードを使用して IBM iサーバーにサインオンすることができる、別のウィンドウが表示されます。

axes	端末セッション
ユーザ <sup>〜</sup> ハ°スワート <sup>*</sup> 再接続	
取消 ロゲイン Advanced	

- 4. IBM iサーバーにサインオンしてアクセスできることを確認したら、 サインオフしてウィンドウを閉じます。
- 5. [aXesセッションの開始]の右側の[開発者セッション]オプションをクリックします。
- 6. Webログオン・ダイアログが表示されたら、ユーザー*dev*、パスワード*dev*を使用してサインオンします。

次のような画面が表示されるはずです。

axes	端末セッション
ユーサ <sup>ĸ</sup> ー   パスワード 再接続 「	

7. 自分のユーザー・プロファイルとパスワードを使用して、IBM iサー バーにログオンできることを確認します。 バックアップの方針

ライブラリAXES、IFSフォルダaXes、そのすべてのサブフォルダを日常 的なバックアップ・サイクルに含める必要があります。これを怠ると、 開発者の作業で重大な損失または完全な損失が発生することがありま す。

「RAMP-TS画面の識別はどこでどのように保持されますか?」を参照してください。

## 構成

プラウザのセキュリティ設定の確認 RAMPの構成 ブラウザのセキュ リティ設定の確認

RAMP-TSを利用するには、ブラウザのクロスドメイン・セキュリティ (異なるドメインのドキュメントにアクセスするドキュメントに関する セキュリティ)を解除する必要 があります。

Internet Explorer のクロス・ドメイン・セキュリティはVLFおよびAxesの URLと同じホスト名を指定することで解除できます。例えば、Hostabcを 10.10.2.181にしてみましょう。

リバース・プロキ シの構成にかかわらずhostabcを参照していること、信 頼済 みのサイトに追加されているhostabcであることから、下記のURLは 利用できます。

http://hostabc:81/images/privatefolder/EX1\_Test\_ENG\_BASE.HTM? Developer=Y+TSIPAddress=hostabc+TSPortNumber=8080

リバース・プロキ シの構成にかかわらず10.10.2.181を参照していること、信頼済みのサイトに追加されている10.10.2.181であることから、下記の URLも利用できます。

http:// 10.10.2.181:81/images/privatefolder/EX1\_Test\_ENG\_BASE.HTM? Developer=Y+TSIPAddress=10.10.2.181+TSPortNumber=8080

下記のURLは利用できません。

http://10.10.2.181:81/images/privatefolder/EX1\_Test\_ENG\_BASE.HTM? Developer=Y+TSIPAddress=hostabc+TSPortNumber=8080

http://hostabc:81/images/privatefolder/EX1\_Test\_ENG\_BASE.HTM? Developer=Y+TSIPAddress=10.10.2.181+TSPortNumber=8080

他のブラウザについてはリバース・プロキシを 設定することによって解 除できます。

「Chrome、Safari、Firefoxのリバース・プロキシ設定」を参照してください。

#### **Chrome、 Safari、Firefox**のリバース・プロキシ設定

RAMP-TSを使用するすべてのPCでこのステップを行ってください。 ドメインはホスト:ポートのコンビネーションで参照されます。例え ば、VLFがポート番号81でホスト MyHost を利用するとします。この場 合、VLFのドメインは MyHost:81 となります。また、RAMP-TS(aXes)が ポート番号 8080でホストMyHost を使用する場合、RAMP-TSのドメイン は MyHost:8080 となります。つまり、VLFとRAMP-TSはそれぞれ異なる ドメインにアクセスすることになります。

ですから、Web サーバーでリバース・プロキシ機能を使用して、クロス ドメイン・セキュリティを解除する必要が あります。上記例のホスト名 で、IBM i Adminインスタンスのリバースプロキシ設定は次のようになり ます:

プロキシー 🕝

SL プロキ	=シー SSL プロキシー拡	張 FRCAリバース・つ	カキシー・キャ	ッシュ		
般設定	フォワード・プロキシー	リバース・プロキシー	プロキシー・テ	チェーニング		
バース・プ リモート	ロキシーの機能: 使用可能 ・・サーバーへのプロキシー系	≝ <b>▼</b> 3 ≣≭: 3				
	要求タイプ	ローカル仮想バス	リモート・サ	ナーバー URL		
例	クライアント要求	/mirror/foo	http://www.m	yserver.com/		
例	リダイレクトされた要求	/mirror/foo	http://www.m	yserver.com/		
0	クライアント要求	/ts/	http://MyHos	st:8080 <mark>/</mark> ts/		
0	クライアント要求	/agi/	http://MyHos	st:8080/agi/		
0	クライアント要求	/axests/	http://MyHos	st:8080/axests/		
追加	)					
Via ∾	,ダーに対するプロキシー・ア	クション:	既存の Via ヘッ	ダー行の保存		-
発信接続	続バッファー・サイズ:		)	지구ト 👻	または	• 0
リモート	・エラー情報のオーバーライ	۴:	使用不可 👻 🕼			
ホスト・	ヘッダーの保存:		使用不可 👻 🕼			
UIIL	・サービー・タイルマウト・		20	f-th		

この例の MyHost:8080の部分をユーザーのホスト情報に変えて、リバー スプロキシを設定します。リバー スプロキシが設定できたら、次の手順 を追ってください。 1.& nbsp; Webサーバーを再起動します。

2.& nbsp; ブラウザのキャシュをクリアします。

3.& nbsp; Fiddlerをスタートします。

4.& nbsp; まずMyHost:8080/ts/ts2/index.htmlに相当するページ、 そして次にMyHost:81/ts/ts2/index.htmlに相当するページを試してくだ さい。

MyHost:8080はRAMP-TS(aXes)のドメインです。

MyHost:8080/ts/ts2/index.htmlを入力すると、すぐに以下のようなページが表示されるはずです。

🅦 セッション 表示 Tools ヘルフ <sup>o</sup>	Auto - 🚽 🖉
axes	端末セッション
2- <del>1)</del> *-	
ለ°ጃワ−ト°	
■ 再接続	
取消ログ	
	Version 2.10.007 - Copyright © 2009-2011 LANSA Group. All Rights Reserved.
S 2 2 2	

この画面が表示されない場合は、aXesが正しくインストールされていないか、または構成に問題がある可能性があります。

MyHost:81はVLFのドメインです。VLFドメインを使って同じページを 試してください。同じページが表示されたら、Webサーバーの構成の変 更が正しく機能していることが確認できます。 RAMPの構成

以下のステップを実行して、フレームワークでRAMPを構成します。 RAMP-NLを使用したことがある場合は、以下のセクションをお読みくだ さい。

フレームワークが全体的なビジュアル・スタイル・テーマを持っている ことを確認する

RAMP-TSサーバーの詳細の指定

スーパー・サーバー・セッションのセットアップ

オプションでフレームワーク・ユーザーとセキュリティをセットアップ する RAMP-NLを使用したことがある場合は、このセクションをお 読みください

RAMP-NLをすでに使用しているフレームワークを開始した場合、そのフレームワークは引き続きRAMP-NLを使用します。このため、RAMP-TSを開始するには、*RAMP-NL*の詳細が関連付けられていないフレームワークから開始する必要があります。

つまり、RAMP-TSを開始する最も簡単な方法は以下のとおりです。

- 最新の組み込みデモンストレーション・システム(lastshipped.xml ファイル)を使用して、VLFを起動します。これにより、デフォルト の構成を使用しています。
- 1つのサーバーのみが定義されていることを確認し、それ を"LANSA iSeries + RAMP TS"タイプに設定する必要もあります。
- 既存のRAMP-NLサーバー定義を削除します。
- 既存のvf\_sy001\_nodes.xmlファイルの名前が変更されているか、削除されていることを確認します。
- 一意のセットのXMLファイル名を使用していることを確認し、フレームワークの詳細を保管します([フレームワーク] -> [プロパティ] -> [フレームワーク詳細]タブ)。
- [名前を付けて保存]オプションを使用して、このフレームワークを 別の名前で保管します。

RAMP TSを使用するように正常に構成した場合、RAMPツールを開くと 以下のように表示されます。



詳細については、「RAMP-TSまたはRAMP-NLとのフレームワークの関連付けについて」を参照してください。

### フレームワークが全体的なビジュアル・スタイル・テーマを 持っていることを確認する

RAMP-TSを使用するときは、フレームワークの全体的なテーマを使用す る必要があります。RAMP-TSアプリケーションでは、全体的なテーマな しを使用することはできません。

<b>ニ</b> フレームワーク					
Web フレームワークの詳細   開発者設定 - Wind 識別子   ユーザー プロパティ   ビジュアル スタイル	lows IS   開発者設定 - IBM i Apache ユーザー管理設定   イソス? アイコン   スタートアップ   使用可能コマンド   フレームワークの詳細   コマンド	妙刈刈り の表示	レーションシップ概要 デザインのエクスオ	夏  ポート  製品情報	
-ビジュアル スタイル	省略値のベース スタイル - YF_VS101 (JPN)	v	省略值	ſ	
使用不可	使用不可の省略値 - VF_VS106 (JPN)	¥	省略值		
濃色背景(小さいフォント)	濃色背景の省略値(小さいフォント) - VF_VS107 (JPN)	w.	省略值		
濃色背景(大きいフォント)	♥indows XP X外仙 濃色背景(大きいフォント) - VF_VS104X	¥	省略值		
ステータスパー フィールト	ステータスバー フィールドの省略値 - VF_VS111 (JPN)	¥	省略值		
URL	URLの省略値 - VF_VS105 (JPN)	×	省略值		
全体のテーマー					
全体のテーマ	2007 春				
▼ エンドユーザーはテーマを変更できます					
(					
				開	:3

#### RAMP-TSサーバーの詳細の指定

- 1. 設計者としてフレームワークを起動します。
- 2. フレームワークの[管理]メニューで[サーバー…]オプションを選択し ます。
- 3. MY/AS400という名前のサーバーを選択するか、[新規作成]ボタンを クリックして新しいサーバーを作成します。
- 4. [識別子]タブで、サーバーを説明するキャプションを入力します。
- 5. [サーバーの詳細]タブで、サーバーの名前を入力し、[サーバー タイプ]に [LANSA for System i + RAMP-TS]を選択します。
- 6. 画面の[RAMP-TS (ターミナル サーバー)]セクションで、RAMP-TS/aXes-TS サーバーのホストとポート番号を次のように入力します。

RAMP-TS (ターミナル サーバー)		
配布サーバーとして保存	<b>V</b>	
IP 7ドレス	<host></host>	
ポート番号	<port></port>	
実行モード・ロードパス	/ts/skins/	👝 HTTPS を使用
RAMP" – ルモート በ – ト ንግ	/ts/dev/	📄 HTTPS を使用
プライベート定義フォルダ		□ 共有わジェ外を含む
RAMP-TS 🌮	ールのインストールと構成をテスト	

- 7. フレームワークを保存して再起動します。
- 8. 設計者としてフレームワークをもう一度起動します。[管理]メ ニューの[サーバー…]オプションをもう一度使用します。
- 9. 定義したサーバーを選択し、[サーバーの詳細]タブに切り替えま す。
- 10. [RAMP-TS ツールのインストールと構成をテスト]ボタンをクリックします。
- 11. 表示されたWebサインオン・ダイアログで、ユーザーdev、パス ワードdevを使用してサインオンします。
- 次のような画面が表示されるはずです。

K Session Display Help	🖉 - Auto - 💌 🔍 🧒	Tracking Info
		Marringm
Usen		
Password:		
Reconnect:		
Cancel Login		
	1 🖻 👅 🚬	

この確認テストが問題なく完了するまで、RAMP-TSの使用を開始 しないでください。 スーパー・サーバー・セッションのセットアップ

RAMPを使用するとき、スーパー・サーバー・セッションをSystem i サーバーにサイン・オンさせる必要があります。サイン・オン・オプ ションを指定する方法は次のとおりです。

- フレームワークを起動します。
- フレームワークメニューを表示し、プロパティオプションを選択します。
- フレームワークプロパティで、ユーザー管理設定タブを選択します。
- サインオンの中のフレームワークを使用するためにリモートサー バーにログオンしますオプションを選択します。
- ダイアログを閉じ、フレームワークを保存します。

Web フレームワークの詳細 開発者設定 - Windows IIS 開発者設定 - IBM i Apache ユーザー管理設定 インスタンスリストリレーションシップ 概要

一権限設定 ── 「 つ い ームワー ウユーサ い ーと認証を使用する	
DBMSテーフルVFPPF06/7にユーザー明細を保管	
ユーザー 設定のXML ファイル:	Vf_Sy001_Users.xml
ューザーの組込インターフェースポイントCEP)をインボート	
-サインオン	
ログオンする必要があります	全て(MS-Windows と ブラウザ)
○ フレームワークを使用するためにローカルにログオンします ✓ ユーザーはパスワードを変更できる	□ ログオン再試行の最大回数 3 □グオン再試行の限度を超えた場合 ○ オロケンざを使用してつっせっく報告します
<ul> <li>フレームワークを使用するためにリモート サーバーにログオ</li> <li>ジ リモート サーバーに接続不可の場合、オフラインで作</li> </ul>	● 39ビージ 20 定 HO し 2 - 9 - 1 (報告 0 ま 9 1)します ○ 7レームワークの致命的エラー

オプションでフレームワーク・ユーザーとセキュリティをセッ トアップする

オプションでフレームワーク・ユーザーとセキュリティを使用できま す。

 フレームワーク・プロパティを表示します。ユーザー管理設定タブ でフレームワークユーザーと認証を使用するオプションを選択しま す。また、オプションDBMSテーブルのユーザー明細の保管も選択し ます。フレームワークを保存して再起動します。

Web フレームワークの詳細 開発者設定 - Windows IIS 開発者設定 - IBM i Apache ユーザー管理設定 インスタンスリストリレーションシップ概要

<ul> <li>権限設定</li> <li>▼ 「レームワークユーザーと認証を使用する」</li> <li>▼ DBMSテーフルVFPPF06/7にユーザー明細を保管</li> </ul>		
ユーザのアクセス権	SYSTEM	
ューザーの組込インターフェースホイント(IIP)をインホート		

- 管理メニューでユーザーオプションを選択します。
- ユーザー・プロファイルの詳細とその権限を指定します。詳細については、F1を押してコンテキスト依存ヘルプを使用します。

多数の開発者が同じアプリケーションで作業する場合

大規模なアプリケーションを刷新する場合、複数の開発者が作業を共有 する必要が生じます。

複数のフレームワーク・バージョンの扱い

スクリプトの命名規則

複数のフレームワーク・バージョンの扱い 「開発アーキテクチャ」を参照してください。

#### スクリプトの命名規則

RAMPスクリプトには、FORMSCRIPT\_137などの名前が割り当てられて います。名前は目的を反映し、数値接尾辞により現行フレームワーク内 で一意にされますが、実際のプログラム上の目的はありません。

複数の開発者が、各自の作業を後日統合する予定で独立したフレーム ワークで作業している場合、複製スクリプト名が複数存在する可能性が あります。この状況がRAMPの技術的問題を引き起こさないとしても、 一意のスクリプトを識別しようとする開発者は混乱するかもしれませ ん。

開発者はスクリプト・エリアでスクリプトの名前を変更できます。生成 されたスクリプト名に、可能であればスクリプトが関連づけられている 5250画面またはアプリケーションに関連する短い接頭辞をつける方法を お勧めします。

「xxxxxxx は親のないスクリプトなので削除してください」も参照して ください。

## RAMPの開始

ここでは、LANSA、およびアプリケーションの刷新に必要なLANSA内 部の機能を起動する方法について簡単に説明します。

LANSAの起動 フレームワークの起動 RAMPの起動 インスタント プロトタイピング アシスタントの起動 コード ウィザードの起動

### LANSAの起動

LANSAを起動するには:

- スタートメニューを使用してプログラムフォルダーを表示します。
- LANSAを選択します。
- LANSA 開発環境オプションを選択します。



LANSA開発環境が表示されます。

### フレームワークの起動

LANSA開発環境からフレームワークを起動します。

- ツールメニューを表示します。
- VL フレームワーク 設計者モードオプションを選択します。

ツール	⑦ ウイントや (W) ヘルフ (H)					
	構成( <u>C</u> )					
	配布 ツール					
	ロシウル モデラー					
ĸ	ウィザ^ート*					
	VL フレームワーク - 設計者モート*(D)					
	VL フレームワーウ - 開発者モード(⊻)					
	VL フレームワーク - 管理者モード( <u>A</u> )					
	VL フレームワーク - ユーザ <sup>、</sup> ーモート <sup>、</sup> ( <u>U</u> )					
*	Visual LANSA エラーロク <sup>®</sup>					
1	IBM i エラー ログ					
-	Web 実行時エラーログ					
	CS Builder/クライアント					
	Integrator スタジオ					
0	LANSA インポペートΦ					
7	テキスト 検索					

#### RAMPの起動

フレームワークのウィンドウからRAMPを起動します。

- フレームワークメニューを表示します。
- RAMP ツール...オプションを選択します。

🍓 HR デモアフツケーション		
ファイル(E) 編集(E) 表示(⊻) ツール(	① ヘルプ(H) ウィンドウ(W) <mark>(フレームワーク)</mark> (管理)	
③作成 白 凸 团 /	🔊 🖂 📮 💭 🔷 (新規作成)	•
HR デモアプリケーション	( ንግስ/ንዮፈ )	
x Detail Test	(779/7-5/12)	
	Hum (テ*サ*インコードテーブル…)	
田	(コードウィザード)	
🗉 🚳 管理	HR アプリケーショ (インスタント プロートタイビング ア	»بروريد)
	システム ドキュメ (RAMP ツール)	
	ご意見・ご質問 (仮想別ップボード)	ジ ・
	最新情報 (マージッール…)	アの行
	法的情報(保存)	53人
	<ul> <li>(名前を付けて保存)</li> <li>カレンダー</li> <li>(保存して重起動)</li> </ul>	भेज ठ. •
	明海レック(保存して終了)	
	く Webアフジケーションとして実	(Ţ ) •
	(Web ユンソール )	→
	(NJ2*)	× •
	< (h/-z)	> >
E 🗄 🗐	メッセージ ローカル JPN V20PGML	IB 10/07/01 11:22

RAMPツール・ウィンドウが表示されます。

#### インスタントプロトタイピングアシスタントの起動

チュートリアル「RAMP-TS001: 刷新されたアプリケーションの基本的な プロトタイプの作成」では、インスタント プロトタイピング アシスタン トの使用方法について説明しています。

インスタント プロトタイピング アシスタントを使用すると、アプリケー ションのプロトタイプを簡単に作成したり、既存のプロトタイプを変更 したりできます。

インスタント プロトタイピング アシスタントを起動するには、フレーム ワークメニューのインスタント プロトタイピング アシスタント…オプ ションを使用します。

🍕 HR デモアフツケーション			
ファイル(E) 編集(E) 表示(V) ツール(	[) ∿⊮ን°(⊞) ሳለንኑ"ን(₩)	(フレームワーク) (管理)	
	S 🗆 🖪 🖽 O	(新規作成)	•
HR デモアプリケーション		( ንኪパティ )	
🛪 🦨 Datel Test		(アフリケーション) (フフリド…)	·
		())	
	Hum	(デザインコードテーブル)	
Ⅲ 😭 お気に入り Ⅲ 🙆 プロヴラミング テクニック		(コードウィザード)	
🗉 🍕 管理	HR アプリケーショ	(インスタント フロトタイピッケ アシスタント…)	
	システムドキュメ	(RAMP ツール)	
	ご意見・ご質問	(仮想クリップボード)	•
	最新情報	(マージッツール …)	アの行 するたい
	法的情報	(保存) (名前を付けて保存)	5る人 封する;
	カレンダー	(保存して再起動)	•
	関連リンク	(保存して終了)	_
		(Webアフリケーションとして実行)	•
		(Web コンソール)	<u>·</u>
		(\\p))	* v
	<	(N/-2)	> >
	パクセージ	ローガル JPN V20PGMLIB (10/07/01 1	1:27

または、ナビゲーション・ペインのポップアップメニューからアプリ ケーションの作成または新しいビジネスオブジェクトを選択します。

👹 Human Relat	ion
7ァイル(E) 編集(E)	表示(2) クール(1) ヘルプ(1) ウィントウ(1) (1
(i) finit	18 D 🛛 A S 🗖 🗖 🔍 🔿
Human Relati	on
■ 50 DEM HR ■ 51 HR テ*モアブ H Human Rel ■ 51 Hy Applic ■ 55 A55(こ入り) ■ 52 PT ■ 55 PT	× ation sation が デカニック
	(アフリケーションの作成。)
	<ul> <li>(ビジネスオフジェクトの作成)</li> <li>(フロハラィ)</li> <li>(削除)</li> <li>位置 ▶</li> </ul>
6	聞く Human Relation 新しいウルトウ ウルトウ(W)

表示されるメッセージで、はいをクリックします。

インスタント	プロトタイビングアシスタントを使用しますか? 🔀
?	インスタント フロトタイビング アシスタントを使用してアフツケーションを作成 - "はい"をクリック 手動でアフツケーションを作成 - "いいえ"をクリック アクションをキャンセル - "キャンセル"をクリック はい(文) いいえ(N) キャンセル

インスタント プロトタイピング アシスタントが表示されます。

🔲 ብንスቃንኑ ፓኪኑቃብድን	<u>ሳ፣ ፖንአቱን</u> ኑ	×
ビジネスオフジェクトの定	→ アリションの付加 → オフジェクトをゲルーフに指定 →  →	^
"ビジネス・オブ:	ジェクト "でどのアクションをとるのか?	
Step 2. 全てのアクシ	ヨンの名前を以下に入力:(カンマで区例る)	
Windowsはオブジェクト- - (例 操作をするときにオ アクションはITの言葉で)	→アクションのアブローチでデザインされています。 オブジェクト(対象)を選択した後に何をするのかを選択します) はなくエンドユーザーが理解できる言葉で表現されなければなりません。	III
オブジェクトは先に指定し MS-Powerpointのオブジ - "クリップボードにこのす アクションは単純な短い頃 - (例、 <i>コピー、新規、編</i> 録 ビジネス・オブジェクトに - (例、 <i>明細、履歴、料金</i> - (例、 <i>明細、履歴、料金</i> - (例、 <i>明細、履歴、料金</i>	っているので、極めて簡潔な言葉で"アクション"を表現することができます。 ジェクトを選択して、マウスで右クリックした場合簡潔な"コピー"というメニューが表示されます。 テキストボックスをコピーする"とは表記されません。 動詞で表記します。 <i>集、印刷、承認、送信、返信、更物</i> ) 関連して短い名詞も使用できます。 そ、 <i>請求、添付、スクジュール、運船、警察、経費</i> ) ジェクトロは以下のアクションが考えられます:	<
<		
アクション:	詳細 , New , 注意 ,	
Step 3. 以下のアクショ	ヨンのリストから、右のビジキス・オブジェクトのリストに必要なものをドラックドロップ	~
複数のビジネス・オブジェク	クトに同様の操作を繰り返します。	~
雙-詳細 <mark>覺-</mark> New 覚-注意	●       # employee       (既存)         ●       詳細       アりション: 詳細 e         ●       New       アりション: New em         ●       注意       アりション: 注意 e	•
	<< 戻る 次へ >> キャンセル	

#### コート・ウィザートの起動

チュートリアル「RAMP-TS003:社員のデータ・フィルターの作成」では、プログラム・コーディング・アシスタントの使用方法について説明します。

コード ウィザードを使用すると、フレームワーク・フィルターとRAMP 画面のコードを簡単に作成できます。

起動するには、フレームワークメニューのコード ウィザードオプション を使用します。

💰 HR デモアフฃケーション			
ファイル(E) 編集(E) 表示(⊻) ツール(	ጋ ላዞን 🖽 ታለንዮን 😡 🕻	フレームワーク)(管理)	
③作成 台口 图 △	S 🗆 🗖 🖉 C	(新規作成)	- • I
HR デモアプリケーション		( ንግ እንን )	
× ■ ● ● ● ■ HR ■ ● ● ■ HR デモアフッリケーション ● リソース ● 和徹 Human Palation	AC Hum	(アフツケーション) (コマンド) (メニュー) (デッザインコードテーフツル)	, 
■ Wy Application ■ ☆ お気に入り ■ ☆ プログラミング テクニック	HR アプリケーショ	(コート・ウィザート・・・・) (インスタント プロトタイビング アシスタント・・・)	
🗈 🚳 管理	システム ドキュメ	(RAMP ツール…)	
	ご意見・ご質問	(仮想クリップボード)	
	最新情報	(マージ ツール )	アの行 「るた」
	法的情報	(保存) (各前友付付て保存)	うる人 きする:
	カレンダー	(保存して再起動)	0
	関連リンク	〈保存して終了〉	
		( Webアフジケーションとして実行 ) ( Web コンソール )	) )
	<	(ヘルフ <sup>*</sup> ) (トレース )	> >
🖻 📰 🔳 🛛 🔸	セージ 準備 ローカ	₩ JPN V20PGMLIB 10/07/01 1	4:36

# コンセプト

RAMPの使用に必要なステップ フレームワーク・ウィンドウ RAMPウィンドウ 画面のタイプ OBJECT-ACTIONユーザー・インターフェース

## RAMPの使用に必要なステップ

以下のステップを完了する必要があります。

ステップ	説明	使 用 頻 度 の 低 い ア プ リ ケ ー シ ョ ン 刷 新 の の 、 の の の 、 の 、 の の 、 の 、 の の 、 の 、	アプリ ケーショ ンの刷新
1. アプリケー ションのプロ トタイプを作 成します。	プロトタイプ 日来なションのす相がションでででした。 とで門るいした。 とび門のは ののすれがす。		
2. 既存のアプ リケーション で5250画面を 識別します。		エント リー・ポ 5250回 だししま す。	刷必るの面なル別するあて画要ー識
3.5250エント リー・ポイン ト画面を記 し、その画面 をフレーム ワークにス ナップインし ます。	フレームワー クは、画面に アクセスし、 表示する方法 を認識する必 要がありま す。		

4. 必要なフ レームワー ク・フィル ターを作成し ます。	強力なフィル ターの使用 は、新しい方 法で5250画面 を再利用する ときの基礎と なります。	必要あり ません。	
5.必要に応じ て、Windows 機能を利用し て新しい機能 を追加しま す。	例えば、e- mail、ビデ オ、グラフ作 成用の高度な 画面を追加で きます。		
6. 配布しま す。	アプリケー ションを配布 します。		

# フレームワーク・ウィンドウ

Demo Application						
<u>File Edit View Actions Tools Hel</u>	p (Framework) (Administrat	ion )				
💥 Sign Off 🛛 😳 New 🍃	Email 🔄 Print 🖂	🗂 Details 🔹 💌 🤆	Transfer 🚺 Calculator			
Employees						
On Tool Bar	🚺 by Skill 🔍 Other	Number	Name	Phone		
	by Name 🔗 by Locatio	n A3564	FREDDY BROWN	(02) 567-6758		
E-@ Demo Application	Specify a full or partial employed	ee name. 00090	VERONICA BROWN	344-2234454545 (02) 9609 4627		
Employees	Employee Surname	A1031	JOHN BLAKE	(02) 9668 9235		
Skills	Employee semane					
Departments	フィルター	インスタ	ンス・リスト			
Monthly Reports	フィルターを使用して、操	乍するオブこのリス	トには、フィルター基準に一致す	「るオブジェクト		
	ジェクトの選択条件を指定	します。 が表示さ	れます。作業するオブジェクトを	をこのリストから		
		選択しま	す。			
Web Sites	Clear the current list c	Search				
🗈 😳 Programming Techniques				<u> </u>		
	S Employee : Basic details (A	0070-VERONICA BROWN)				
	🐏 Basic details 🦳 🐉 Skills	🥐 Transfer 🛛 🔗 Email	📼 Video 🛛 🗟 All Details	Documents		
	Employee Number	A0070		Save		
ナビゲーション・ペイン	Employee Surname	BROWN		•		
ケーション間およびビジネス・	Employee Given Name(s)	VERONICA	コマント	ウトを操作するために		
オブジェクト間を移動します。	Street No and Name	12 Railway Street	y Street ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・			
	Suburb or Town	Baulkham Hills				
	State and Country	NSW Australia				
	Post / Zip Code	2153				
	Home Phone Number	(02) 9609 4627				
Messaces Ready Local ENG DCXUSER 10/10/06 15:42						

## RAMPウィンドウ



メッセージ・エリア 画面トラッキングエリア RAMP-TS 5250エミュレータ・セッション 画面とスクリプトのリスト 詳細エリア メッセージ・エリア



RAMPメッセージ・エリアには、ユーザーの現在位置と実行している処 理に関するメッセージが表示されます。

画面トラッキング・エリアで選択された画面に関するメッセージが表示 されます。また、このエリアを使用して画面の分類も行います。



メッセージ・エリアには、画面の追跡に使用できるボタンがあります。

検証	検証ボタンを使用して、現在の5250画面のレイアウト を調べて、オンライン・レポートを生成します。この レポートは、問題を分析したり、サブファイルで使用 されている行と列を確認するために使用します。
スナッ プ ショッ	スナップショットボタンを使用して、GUIモードまたは 5250モードで現在の5250フォームのスナップショット を作成し、ビットマップで保存します。イメージ:
F	<ul> <li>このイメージをRAD-PADプロトタイプのコマンド タブにドラッグ・アンド・ドロップして、他の開発 者またはエンド・ユーザーとのデザイン・セッショ ン中のコミュニケーションを強化できます。</li> <li>このイメージは、システムのマニュアルを作成す</li> </ul>

	るときに役立ちます。
	<ul> <li>このイメージは、接合画面、宛先画面、または特殊画面に関連付けられている5250画面を正確に覚えるときに役立ちます。</li> <li>ボタンに表示される(nnn x nnn)という数字は、保存され</li> </ul>
	るスナップショットのピクセル・サイズです。この数 字は、RAMP ツールウィンドウのレイアウトを変更す ると変化します。
再ス タート	再スタートボタンを使用して、追跡情報を消去して追 跡を再起動します。
[ナヒ <sup>゙</sup> ゲ <sup>ー</sup> ション スクリプト の自動 更新]	このオプションを使用して、ナビゲーション・スクリ プトの自動生成をオンまたはオフにします。

画面トラッキングエリア

RAMPウィン ドウ	メッセー ジ・エリ	RAMP-TS 5250エミュ	画面とスクリプ トのリスト	詳細エリア
	ア	レータ・セッ		
		ション		

トラッキング・エリアには、現在のRAMP-TSセッションで表示している 画面が表示されます。このセッションを終了すると、追跡情報はクリア されます。

トラッキングエリアで画面に使用される色によって、その画面のステー タスと割り当てた画面タイプが示されます。

Unknown Form	画面は識別されていません。
wrkjob	<ul> <li>画面は指定されていますが、RAMPで定義されていません。</li> <li>画面の目的に応じてアプリケーションの画面を定義する必要があります。</li> <li>Destination画面は、エンド・ユーザーが作業する画面です。</li> <li>Junction画面は、ナビゲーションの目的でのみ使用されます。</li> <li>Special画面は、メッセージなどに使用されます。</li> </ul>
i5OSMainMenu	画面はJunction画面です。
DisplayMessages	画面はSpecial画面です。
	画面はDestination画面です。



#### RAMP-TS 5250エミュレータ・セッション

RAMPウィン	メッセー	画面トラッキ	画面とスクリ	詳細エリ
ドウ	ジ・エリ	ングエリア	プトのリスト	ア
	ア			

RAMP-TSエミュレータ・セッションには、5250セッション画面が表示され、左側にアクション・タブが表示されます。[Screens]タブでは、名前を付ける必要がある画面およびフィールドの名前を指定します。



[Lock Screen]オプションを選択すると、別の画面を表示するリンクやボ タンをクリックしたときに現在の画面が表示されたままになります。 表示/非表示ボタンをクリックして、アクション・タブを非表示にするこ とができます。


製品ベンダーから要求された場合のみ、RAMP-TSセッション・ウィンド ウのメニュー([Session]、[Display]、[Help])を使用する必要があります。

## 画面とスクリプトのリスト

# 画面とスクリプトのリストには、フレームワークで定義されているすべての5250画面と関連するスクリプトが表示されます。

デフォルト セッション			
保存 🔻 新525	i0アフツケーション・	セッション 肖	山(除
		検索	次を検索 🗌 スクリフトト内
名前	り゙ループ	最終変更	‡+7°ション Δ
□ セッジョン			ร"วิสไปก ของวิลว
□ □ Junction (4)		201007	Login Form
i50SMainMenu		201007	
😠 PSLSYSMain		201007	
∃ FindEmployee		201007	
Destinatio		201007	
		201007	
DisplayEmp		201007	~

操作する画面とスクリプトを選択します。

リストの最上部にある検索フィールドを使用して画面とスクリプトを検 索できます。スクリプトの内容を検索する場合は、In スクリプト チェッ クボックスを選択します。

RAMPウィンドウでは、フレームワークへの変更を次の2つの方法で保存 できます。

全て保存	エンド・ユーザー・モードで実行するためのすべての スクリプトの生成、Webサーバーの詳細のアップロー ドを含む、完全フレームワーク保存が実行されます。 通常、エンド・ユーザー・モードでアプリケーション をテストする前に[全て保存]を実行します。
保存	処理内容を完全に回復できる部分フレームワーク保存 が実行されますが、実行時スクリプトの生成とサー バー詳細のアップロードは実行されません。

#### アプリケーションを配布したり、エンド・ユーザー・ モードで実行するには、完全フレームワーク保存を実 行する必要があります。

新5250アプリケーション セッション ボタンを使用して、画面とスクリプトを別の5250アプリケーション・セッションに分類できます(「画面とスクリプトの編成」を参照してください)。

#### 画面とスクリプトの編成

アプリケーションが大きくて複雑な場合は、アプリケーションの方針に 従って画面と関連するスクリプトを個別のグループに分類できます。分 類するには、個別の5250アプリケーション・セッションをRAMPウィン ドウで作成します。

数百の画面でアプリケーションを開発すると、アプリケーションのオブ ジェクト数が原因で管理がさらに複雑になります。また、アプリケー ションの初期起動時間は、アプリケーションのオブジェクト数に応じて 直線的に増加します。

この例では、Default Sessionに加えて、Personnel Applicationセッションが 作成されています。



新しいアプリケーション・セッションを作成するには、[新5250 Application Session]ボタンをクリックします。アプリケーション・セッ ションを削除するには、画面およびスクリプト・リストで[セッション] を選択し、[削除]をクリックします。

5250アプリケーション・セッションはお互いが完全に独立しているの で、お互いの存在を認識していません。つまり、あるセッションのスク リプトは別のアプリケーション・セッションのオブジェクトに移動でき ず、ログオン、ログオフなどの一般的なスクリプトおよびメッセージを 複製する必要があります。 個別のセッションが5250アプリケーション・セッションごとに開始され ます。

新しいグループを作成するには、RAMPウィンドウの新しい5250アプリ ケーション セッション ボタンをクリックします。そのキャプションは セッションの詳細エリアで編集できます。

常に1つの5250アプリケーション・セッションだけをアクティブにできま す。アプリケーション・セッションを変更するには、そのセッションの タブを表示します。定義したすべての画面と作成したすべてのスクリプ トが現在の5250アプリケーション・セッションに保存されます。

## 詳細エリア

セッションの詳細 Destination画面の詳細 スクリプト・エリア

## セッションの詳細

RAMPウィ メッセー ンドウ ジ・エリ ア	画面ト ラッキン グエリア	RAMP-TS 5250エ ミュレー タ・セッ ション	画面とス クリプト のリスト
------------------------------	---------------------	-----------------------------------------	----------------------

## セッションの詳細を使用して、5250アプリケーション・セッションのさ まざまな設定を指定します。

RAMP-TS 5250 セッション 詳新			
ーセッション - デウォルト セッション―			
キャプション デンフォルト セッ	9a)		
ユーザー オフ <sup>ッ</sup> シェクト タイフ°	RAMP_	DEFAULT_SESSION	名前の確認
RAMPの既定値のレイアウト	・サイズ		
高さ 412 幅	710	_ <u> </u>	左
Top マスクの高さ	Bo	ttom マスクの幅	
-RAMP スクリーン レイアウト スタ 回 国会レイアウト	XTU		
		0 /H 01/ /r	]
- スタロールハーー - 水平スタロールハッ	ーを表示		バーを表示
🔽 不明な52507ォームの表示	示時にフレームワ	ークをロックする	
📃 既存のコネクションのユーザ	-/パスワードを	再利用する	
セッションを常にユーザー オフシン	ェ小 名/タイプ:	を持ったサーバーヘリンクする	
RAMP-TS 最大ログオフ待機	時間(秒)	1	0
一特別なフィールド受け渡し―			
52507ィート・名	ファンクションキー	VL //ンドラー (クラス VF_AC017	オブジェウト) 🔼
1 DEPTMENT	F4	DF_PRM04	
2 SECTION	F4	DF_PRM04	
3 DATE_*	F4	DF_PRM07	
4	F4		
5	F4		
6	F4		×

[キャプ RAMP 5250アプリケーション・セッションのキャプション。 ション] [ユー フレームワーク ガイドの「ユーザーオブジェクトタイプ」を参照してく ザー・ ださい。

オブ ジェク ト名/タ イプ]	
高さ	フレームワークに表示されるときの5250画面のデフォルトの 高さ。
幅	フレームワークに表示されるときの5250画面のデフォルトの 幅。
F	RAMP画面タブの最上部と5250画面の間のデフォルトの間隔。 このオプションを使用して個々のRAMP画面での画面タイトル を非表示にできます。
左	フレームワークに表示されるときの5250画面のデフォルトの 左インデント。
Topマ スクの 高さ	5250画面の最上部を非表示にするときに使用できるマスクの デフォルトの高さ。 このオプションを使用して個々のRAMP画面で画面タイトルを 非表示にできます。
Bottom マスク の幅	5250画面の下部を非表示にするときに使用できるマスクのデフォルトの高さ。
[RAMP スクリーン レイアウト スタイル]	<ul> <li>[RAMP スクリ-ンレイアウト スタイル]を[フローレイアウト]に設定すると、RAMP 画面は表示に使用可能なスペースに収まるように自動的にサ イズ設定されます。</li> <li>[フロー]を使用した場合、</li> <li>画面の特定の位置およびサイズはサポートされていませ ん。</li> <li>画面エリアの上部および下部のマスキングを使用して画 面のコンテンツを非表示にすることはできません。</li> <li>ファンクション・キーの青色のバーを使用したり、表示 したりすることはできません。</li> <li>[水平スクロールドーを表示]と[垂直スクロールドーを表示]オプション</li> </ul>

	は、明確な理由により使用できません。 固定の場合、RAMP画面は表示に使用可能なスペースに収まる ようにサイズ設定されません。
[スク ロー ル・ バー]	[水平スクロールドーを表示]オプションをオンにした場合、VLF.WIN アプリケーションでは、固定サイズの5250画面が表示エリア に収まらないときに水平スクロール・バーが表示されます。 VLF-WEB/NETアプリケーションは、[水平スクロールドーを表示]オ プションがオンになっている場合と同様に常に動作します。 [垂直スクロールドーを表示]オプションをオンにした場合、VLF.WIN アプリケーションでは、固定サイズの5250画面が表示エリア に収まらないときに垂直スクロール・バーが表示されます。 VLF-WEB/NETアプリケーションは、[垂直スクロールドーを表示]オ プションがオンになっている場合と同様に常に動作します。
不5250フム示フムクロすのの、5250フムホフムクロす	このオプションは、不明な5250画面が表示されたときにフ レームワークにロックを適用します。 ロックが適用されると、ユーザーは、定義済みの5250画面に 移動するまでフレームワーク内を移動できません。 このようなロックが適用された場合は、ユーザーはフレーム ワークを終了できます。 通常このオプションは、不明な5250画面または予期しない 5250画面をトラップするために使用します。 すべての5250画面がRAMPに対して定義されている必要があ る、厳密に定義され、管理されたセッションでは、このオプ ションをオンに設定します。管理されないセッションでは、 このオプションを必ずオフに設定します。
既コシのザパワを用 存ネョユースー再す のクンー/ ド利る	このオプションを使用して、この5250アプリケーション・ セッションがサーバーに接続する必要があるときに、前回の 正常なサーバー接続を確立するときに使用したユーザー・プ ロファイルとパスワードを再利用するかどうかを示します。 このオプションによって、開始する必要がある新しい5250ア プリケーション・セッションごとに何度もユーザー・プロ ファイルとパスワードの入力を要求されないようにすること ができます。通常、ユーザーは、最初に確立するアプリケー ション・セッションに対してだけ入力を要求されます。 このオプションは、個別のユーザー・プロファイル・オプ

	ションまたはスーパー・サーバー接続の値によって自動的に オーバーライドされます。 フレームワークは、ユーザーがフレームワークを終了すると きまでサーバー接続の確立に使用した前回のユーザー・プロ ファイルとパスワードを記憶していますが、終了すると、そ の詳細は破棄されます。
セシをユザブク / プ持サバリすッョ常ーージトタをっーーンるンに オェ名イ た にク	ユーザーが5250アプリケーション・セッションを接続する場合は通常、接続するサーバーを選択するように指示されます。 このオプションを使用して、ユーザーがこのサーバー接続に関して選択できないようにすることができます。 このオプションを使用すると、無条件で5250アプリケーション・セッションはサーバーに関連付けられます。 このオプションを使用するには、最初に固有のユーザー・オブジェクト名/タイプをサーバーに割り当てます。 割り当てるにはフレームワークの管理メニューのサーバーオプションを使用します。 例えば、このサーバーにはユーザー・オブジェクト名/タイプ SERVER_2が割り当てられています。
	パーパー・パーの詳細 #8917 アイン         「「」」」、「「」」、「」、「」、「」、「」、「」、「」、」、」、」、」、」、

	に関連付けるサーバーを選択できません。 このオプションをセッションから削除するには、このオプ ションをデフォルトのブランクに戻します。
[RAMP- TS 最大 ロゲ わ待 機時間 (秒)]	RAMP-TSセッションをログオフ(サインオフ)する必要がある 場合、フレームワークが非同期時のログオフ操作の完了を待 機する必要がある最大時間をこの値に秒数で指定します。デ フォルト値は10秒です。0から120の範囲の整数値を指定しま す。この値の単位は秒数です。
特別な フィー ルド受 け渡し	フィールド用の高度なプロンプト機能。 プロンプトを表示するフィールドの名前、使用するファンク ション・キー、プロンプタとして使用するVisual LANSA フォームを指定します。 詳細は「高度なプロンプト」を参照してください。

## Destination画面の詳細

RAMPウィ ンドウ	メッセー ジ・エリ ア	画面ト ラッキン グエリア	RAMP-TS 5250エ ミュレー タ・セッ	画面とス クリプト のリスト
			ション	

Destination画面を「画面とスクリプトのリスト」で選択すると、 Destination画面の詳細が表示されます。

SLTABMain			
キャプジョン			
ケ <sup>*</sup> ルーフ <sup>*</sup>			
-RAMPの既定値のレイアウト サイズ			
高さ 412 幅 710 上	左		
Top マスクの高さ Bottom マスクの幅			
RAMP スタリーン レイアウト スタイル			
	○ 固定レイアウト	🔘 ንローレイアウト	J
ターケット By			
libUSMainMenu libUSMainMenu ReviewMaintPrint ReviewMaintPrint	int		
保存 份	存		
-ファンクション キー使用可能	_関連するコマント・ハントラー		
4~ Caption 利用可省利用可 FKEY ≤ ▲	<u></u>	わずずェ外内	^
Enter Enter 🗹 🗹 1	HR Details	Employees	
F1 F1 2	王 (の)の分離 Details Wrapper	Employees Employees	
		Telephone Search	
F4 F4 5	▼ 言羊糸田	Active Jobs	
F5 F5 🗌 🗍 6	<		>
F6 F6 🗌 7		再表示	
F7 F7 . 8			
	セッションID *AUTO		

これらのDestination画面の詳細を指定できます。

キャプ ション	
ク゛ルーフ <sup>。</sup>	この画面のグループ名を入力します。省 略可能です。 このオプションを使用して、関連する画 面に同じグループ名を入力できます。こ れにより、その画面を「画面とスクリプ

	トのリスト」で一緒にソートできます。 画面とスクリプトの基本的な構成につい ては、「画面とスクリプトの編成」を参 照してください。
RAMPの規定値のレイアウト サイス	この画面に対して「セッションの詳細」 で設定されたデフォルトのレイアウト寸 法を永久にオーバーライドする場合は、 これらのプロパティを使用します。
RAMP አሳሀ–ን レイアウト አቃイル	<ul> <li>[RAMP スクリーン レイアウト スタイル] を[フローレイアウト]に 設定すると、RAMP画面は表示に使用可能 なスペースに収まるように自動的にサイ ズ設定されます。</li> <li>[フローレイアウト]を使用した場合、</li> <li>画面の特定の位置およびサイズはサ ポートされていません。</li> <li>画面エリアの上部および下部のマス キングを使用して画面のコンテンツを 非表示にすることはできません。</li> <li>ファンクション・キーの青色のバー を使用したり、表示したりすることは できません。</li> <li>[水平スクロールバーを表示]と[垂直スクロー ルバーを表示]オプションは、明確な理由 により使用できません。</li> <li>[固定]の場合、RAMP画面は表示に使用可 能なスペースに収まるようにサイズ設定 されません。</li> <li>[セッション]とは、[セッション]のプロパ ティから継承された値です。</li> </ul>
<i>ፃ−</i> ታ <sup>*</sup> ット	このリストには、この画面から移動でき る画面が表示されます。 リストの最初の画面はexit junctionです。 つまり、この画面がデフォルトで移動で きる画面です。vOverrideExitJunctionプロ

	<mark>パティ</mark> を使用して、スクリプトのexit junctionをオーバーライドすることができ ます。
<b>9-ታ</b>	このリストには、この画面に移動できる 画面が表示されます。
ファンクション キー使用可能	<ul> <li>これは、5250画面で利用可能なすべてのファンクション・キーのリストです。</li> <li>このリストを使用して、5250画面のファンクション・キーを使用可能または使用不可にできます。また、対応するファンクション・キーと同じ機能を持つRAMP画面のプッシュボタンの実行時表示を有効または無効にすることもできます。</li> <li>使用可能にできるのは、5250画面にすでに存在するファンクション・キーだけです。</li> <li>例えば、5250画面にファンクション・キーだけです。</li> <li>例えば、5250画面にファンクション・キーだけです。</li> <li>例えば、5250画面にファンクション・キーだけです。</li> <li>のえば、5250画面にファンクション・キーだけです。</li> <li>アプリケーションに影響しません。</li> <li>ただし、宛先画面のボタン・スクリプトでF10キー用の独自のスクリプトを追加した場合は、RAMP画面でF10キーを使用可能にできます。</li> <li>ファンクション・キーを使用可能にできます。</li> <li>ファンクション・キーを使用可能にするには、利用可能なキー列のチェック・ボックスを選択します。</li> <li>デタンのキャプションは、キャプション利で変更できます。</li> <li>これらのファンクション・キーとボタン</li> </ul>

	使用して、実行時にオーバーライドする ことができます。
関連するコマンドハンドラー	RAMP画面が接続されるコマンドハンド ラータブ。 コマンドハンドラータブは、アプリケー ションのプロトタイプを作成するときに 作成されます。
ννε	<ul> <li>画面に対して開始するSystem i 5250セッション(ジョブ)を指定します。</li> <li>*AUTO:デフォルト値で、フレームワークが必要な5250セッションを自動的に管理することを示します。このタイプのセッションを管理セッションとやびます。このセッションと、ビジネスオブジェクト、およびインスタンスリストと完全に統合され、すべてのスクリプト機能を利用できます。</li> <li>SESSION_A-&gt; SESSION_Z:コマンドハンドラーまたはタブに対して開始する非管理セッションは、主にユーザーをログオンして、特定の開始地点に誘から、ユーザーは管理されないの行きます。セッションは管理セッションはに、1000000000000000000000000000000000000</li></ul>

れません(非管理のため)。 非管理セッションは、既存のアプリケー ションの大部分をフレームワークですば やく再利用できるので、便利です。 例えば、非管理セッションを"System Tables"というビジネスオブジェクトに関 連付けられたコマンドとしてのみ使用で きます。 ユーザーがフレームワークメ ニューの"System Tables"をクリックする と、ユーザーをログオンし、例えば50個 のシステムテーブルを管理する5250メ ニューにユーザーを誘導するフルスク リーン5250セッションが表示されます。 これにより、フレームワークでの識別や 登録が必要ない、管理されない方法で、 例えば数百の5250画面で構成され る"System Tables"機能全体にアクセスでき ます。ユーザーが"System Tables"タブから 移動して後でそのタブに戻った場合、現 在の5250セッション画面が、画面にタイ プに関係なく再表示されます。画面のナ ビゲート(つまり、画面の管理)は行われま せん。ほとんどの場合、ユーザーは未定 義または不明な5250画面から移動すると 考えられるからです。 つまり、ユーザーをログオンし、アプリ ケーションの定義済み開始地点にユー ザーを誘導し、5250アプリケーション・ エリア内でユーザーが自由に移動できる ようにする必要がない場合は、必ず \*AUTOを使用します。

スクリプト・エリア



詳細エリアには、「画面とスクリプトのリスト」で選択したスクリプト の詳細が表示されます。

スクリプトはアプリケーションを追跡する際に自動的に生成されますが、スクリプトの編集が必要になることがあります。

- FORMSCRIPT\_7-役割: Unknown operation using and FORMSCRIPT 7 キャプション NewEmployee - このフォームをどこからでも起動 ゲルーフ° 📋 🖺 关 笋 🥂 🖻 🔍 🕗 🎓 😽 🚱 🖗 🚱 🚱 🥝 🤪 🎬 🛷 スクリーン名検索 // This script was generated from skeleton file vf\_fpm030\_D.js, which is in your LANS // You may alter this file to change how new scripts are generated. \*\*NOTE\*\* You must // Script skeleton files are replaced with the standard shipped versions when reinste { /\* ------ \*/ /\* ============== vHandle ABRIVE ============= \*/ /\* ------\*/ /\* Handle arrival at this Destination \*/ /\* oPayload: The payload supplied by the event initiator \*/ /\* oPreviousForm: Reference to previous Form object \*/ vHandle ARRIVE: function(oPayload, oPreviousForm) var bReturn = true; var strCurrEmpno = GETVALUE("txtEmpno"); if ((this.uTrim(strCurrEmpno) == "") && (this.uTrim(oPayload.empno) != "") ) { /\* Insert was sucessful \*/ /\* Issue a message \*/ MESSAGE("Employee ", oPayload.empno ," created"); AVSIGNALEVENT("Add\_List\_Entry","BUSINESSOBJECT", oPayload.empno) AVCLOSEFORM(); } 3 < >

## スクリプト・ツールバーには、スクリプトでの作業時に役立つボタンが あります。左から次のようなボタンがあります。

	コピー
	貼り付け
×	切り取り
6	元に戻す
C	やり直し
	すべて選択
	検索
	置換
	フォント・サイズの拡大
	フォント・サイズの縮小
	行番号の表示/非表示
	行のコメント・アウト
	行のアンコメント・アウ ト
	行のインデント
0	行のインデント解除
<b>S</b>	コードの印刷
	変更のコミット

	スクリプトの確認
スクリーン名検索	[スクリーン名検索]を使用する と、画面名をすばやく見 つけることができます。 名前をクリックして画面 の名前をコードの現在の 挿入ポイントに貼り付け ることができます。

スクリプトのフォーマットと編集の方法については、「スクリプト・ ポップアップ・メニュー」を参照してください。

#### スクリーン名検索

RAMP-TSスクリプトを記述するときは、長い画面名を大文字と小文字を 区別して入力しなければならないことがよくあります。この処理を簡単 に行うには、スクリプト・エディター・フレームの上部に表示される [スクリーン名検索]を使用できます。

📄 🛍 📈 河 🥐 🖻 🔍 🥥 🏠 😓 💀 🚱 😳 🥥 🎲 🔛 🛷 200-024検索

これを使用するには、画面名を入力する必要があるポイントまでスクリ プトを入力します。例:

if (oPreviousForm.vName == I

画面名を挿入する場所にエディターの挿入ポイント・キャレットを表示 します。

上に移動し、[スクリーン名検索]内でクリックします。これを使用して、名前 のeorキャプションに表示される文字列(例:emp)を入力し、表示された リストから画面を選択して、使いたい画面の名前をすばやく見つけま す。

二重引用符で囲まれた画面名がスクリプト内に以下のように貼り付けら れます。

画面名を選択しないで[スクリーン名検索]をキャンセルするには、クリックしてスクリプトに戻ります。

if (oPreviousForm.vName == "DisplayEmployee"
{

画面のタイプ

アプリケーションの刷新では、5250アプリケーションの画面の分類から 始めます。

- Destination画面は、エンド・ユーザーが実際に作業する5250画面で す。この画面は、変更されずにVisual LANSAフレームワークにス ナップインされます。
- Junction画面は、ナビゲーションの目的でのみ使用されます。この 画面は、刷新されたアプリケーションでは表示されません。
- Special画面は、上記の2つのカテゴリーに当てはまらないメッセージなどの画面です。

## Destination画面

Destination画面とは、エンド・ユーザーがオブジェクトを操作する画面です。通常これは、メンテナンス画面です。

< Session Display Help	- Auto - 💌 🔍 🧔
INQUIRE38 資格情報の検索/保守 10/0	07/23 18:00:51
社員№	2125
取得日 資格 (DDMMYY) コード 資格記述 コメント	资格
98/03/25 ADMIN1 Administrath Part 1 Met requirement	D
98/05/05 FCD Economics Degree	D
94/06/05 ENG English Degree With distinction	D
96/09/15 INDUCT Company Induction	P +
F1= ヘルプ F3= 終了 F4=PROMPT F12= 取消し F14=MSG F21= 変 F22= 削除	更
S 🚱 🙆	▼ 🛛 🖾 上 津

しかし、メニューでもあります(「RAMP-TS002:アプリケーション・セ グメント全体の迅速な刷新」を参照してください)。



Destination画面は、RAMPアプリケーションで変更せずに再利用できます。



Junction画面

エンド・ユーザーは、Junction画面を使用してDestination画面に移動します。



ナビゲーションの目的でのみ使用されるこの5250画面は、RAMPアプリ ケーションではエンド・ユーザーに表示されません。

代表的なJunction画面には、次のものがあります。

- 5250メニュー
- 「Work With」タイプの5250画面(この画面は、単にデータ形式のメ ニューです)
- 受注番号、顧客番号、製品番号などのキーを入力して、詳細情報を 表示したり、アクションを実行するほとんどの5250画面

Special画面

Special画面は、ナビゲーション画面にもDestination画面にも当てはまら ないメッセージなどの画面です。



この画面は、5250画面フローのどの時点でも予期せず表示されることが あります。例:

- サイン・オン時に表示される5250表示メッセージ画面
- どの時点でも表示される可能性がある5250中断メッセージ画面
- ユーザー独自のアプリケーションの致命的なエラーメッセージ画面
- 5250再開対話式セッション画面

Special画面には通常、スクリプトが関連付けられています。このタイプ のスクリプトは通常、5250画面フローから画面を削除するためにキーを 送信するか、アクションを実行するので、削除スクリプトと呼ばれま す。 命名規則

RAMP-TSプロジェクトでは、以下の規則を強くお勧めします。

- 名前には、アルファベット(A->Z、a-z)と数値(0-9)のみを使用します。次に、名前はコード・ページで不変とします。
- 名前に埋め込みブランクを使用してはなりません。
- 名前の長さはおよそ30文字未満とします。長い名前は紛らわしく、
   スクリプトの作成やデバッグ時にエラーが発生しやすくなります。
- 通常、名前は大文字と小文字を区別します。
- 重複する名前を使用する、大文字・小文字ベースの命名規則を運用 してはなりません。例えば、画面のフィールド に"EmployeeNumber"を、その画面のサブファイルの同じフィールド に"EMPLOYEENUMBER"を使用するような命名規則では、何らかの 段階で混乱が生じることは明らかです。名前は、大文字と小文字に関 係なく一意である必要があります。

## 命名プロンプタ

RAMP-TSでは、プロンプタを5250画面の名前付きのフィールドに自動的に関連付けることができます。

これらのプロンプタを定義すると、プロンプタはStartDateなどの特定の 名前やDate\_\*(つまり、Date\_で始まる名前を持つフィールドはこのプロ ンプタに関連付けられる必要がある)などの一般的な名前を使用して、そ れ自体を画面のフィールドに関連付けることができます。

一般的な名前とともに自動プロンプト機能を使用すると、選択した命名 規則に影響を与える可能性があります。

## OBJECT-ACTIONユーザー・インターフェース

フレームワークを含むSystem iアプリケーションとWindowsアプリケー ションは、ユーザー対話用に同じ基本設計、OBJECT-ACTIONインター フェースを共有します。

このインターフェースでは、ユーザーは最初にオブジェクトを選択し、 次にそのオブジェクトに実行するアクションを選択します。反対に ACTION-OBJECTインターフェース(コマンド行アプリケーションなど)で は、最初にコマンドを指定し、次に対象のオブジェクトを指定します。 このように基本的な部分が類似しているので、System iアプリケーション はフレームワーク・モデルに適合します。

System iアプリケーションのナビゲーション画面は、フレームワークでは、操作するオブジェクトをすばやく見つけることができるフィルターやインスタンスリストなどのグラフィカル要素に置き換えられます。

典型的なWork with画面のオプションと関連する画面は、コマンドタブの 集合になります。

フレームワークでは、OBJECT-ACTIONモデルは、高度なGUI(Graphical User Interface: グラフィカル・ユーザー・インターフェース)で表されます。

基本モデルを共有するSystem iアプリケーションとフレーム ワーク・アプリケーション

ビジネスオブジェクト、フィルター、およびコマンドハンドラー(画面) のフレームワークの基本概念は、System iアプリケーションでは次のよう に表示されます。

(Please & () as hit for (0, ) on Consultant Abril Grides (0)	-	C#8			
લો દોદા તેણે લોકો માં કોસ	ana ale. Work with Coject Links (9	5000			
Type choices, press Ente			· · · · · · · · · · · · · · · · · · ·		
doject	<u>.</u>	74	L2-		
Object type Octail Otsplay option	+011 +322 +322	HELSER, +NLRER TRV, +NAME, +RASEC FRV, +USER, +ALL			
10 Sec.	en A - (24 x 80)	Magn		C (5 (2)	
the diff	yes Comunication Actions (Indo-		ビジネス・コ	オブジェクト・コマンド	
ビジネス・オブ	ジェクト・インスタ	ンス・リスト <sup>Objel</sup>	ET LINKS		1000
1	actory	/			
FOREX12 FARFYIN F244More keys Tup	options, press E	nter.			
2	Ec 3=Copy 4	Remove 5=Display	7+Rename 8=Displ	lay attributes	
(7)82-Immericantific tarlat	letha ge current d	irectory			
Opt	Objec. link	Type Attr	ibute Text		
-	bin	DIR			
-	dcximg	DIR			
	dex ism	019			
	deepak	STHF			
	dej	DIR			
_	dev	910	100	Property Line Mr.	1288
	decponilb a man	a se a se a se a se a se			-
	doug	Record Diject (ANK		- Electr	y Attributes
	Type on				
Y ar	meters or con depart	1 "Jane"	Faith Lines		1.47.00
5.24	wit Edeboon			figures report in on	000 N
£22	Display entir		· · · · · · · · · · · · · · · · · · ·	Busistiany exemptions	·
· ***	orsprag entri				13448
100				All Audited Title	22
-D lines	Sendon Automobile Started			faat offe	"主天常社"
CP INC	and a second sec	"リネーム"っていら	12 K=_	Read for another (MC) Read for anythics (MC)(MC)	按示周1生
		04-7 167k			コマンド・ハンドラー
	elettaid	PERFORMENT PERFORMENT	Differ to use this disaler	Frank Delar in continue	
	130.000			ra-cell fabric-pet fabricaries a	NUMB TOPOS
	- <b>1</b>		64-43*		46/46

次のような要素があります。

フィル ター	ここでは、Work withコマンドによって、表示するオ ブジェクトのリストをフィルターするためのオプショ ンが提供されます(多くの"Work with xxxx"インター フェースはメイン・ディスプレイの内部もフィルター できます)。
ビスブクイタススネオェ・スリトン	フィルターの検索基準にマッチするリンクのリスト。 これらのリンクがユーザーのビジネスオブジェクトに なります。
ビジネ ス・オ	それぞれのビジネスオブジェクトに対して実行できる 2=Edit、7=Rename、8=Displayなどのオプション。

ブジェ クト・ コマン ド	
コマン ド・ハ ンド ラー	コマンドの実行時に実行されるプログラム(例として 7=Renameと8=Display attributesが表示されています)。

## フレームワークでは、同じ概念が次のようにGUIで表示されます。

b by have i to by Location will by Sail Cl, Oth Select by Rull or Partial Latt. Name Select al Employees Select where Employee Salary a greater than Select Select by Department Gude	er	Name SROWN N ALAN BLOODS HISON SSMISON 1	K
	A 100 A	ME DY	ブジェクト・コマンド
Clear the current list of employees Employees . Electric (4300) - Robert SHCTHE() Basic details	Search Allow Sale Allow Sale Allow Generation Allow Generation Contraction (C. All Details	on Poki Kr R/f880	2 G
eloyee Grven Name(b) Robert eet Ne and Name Soft of Town te and Country M / Zo Code Robert Robert Soft Outrue Road, CEE WHY, Robert Robert Soft Outrue Road, CEE WHY, Robert Robert Soft Outrue Road, CEE WHY, Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Robert Rober		<mark>~全ての</mark> ~転送	明細"コマンド・ハンドラー コマンド・ハンドラー

刷新の問題

アプリケーション内で最も重要で複雑な5250プログラムが刷新の落とし 穴になる可能性がある アプリケーションはどのくらいの期間でRAMP化できるか?

再利用、修正、または書き換え

アプリケーション内で最も重要で複雑な5250プログラムが刷新 の落とし穴になる可能性がある

刷新の最も巨大で最も厄介な落とし穴は、アプリケーション内の最も重 要でかつ通常最も複雑な5250プログラムに存在します。そのプログラム は、ERPアプリケーションではオーダー・エントリーを処理し、保険ア プリケーションではポリシーマスター更新を行います。

すべての5250アプリケーションには、この巨大で厄介な5250プログラム が少なくとも1つは存在します。

「このプログラムを処理できれば、RAMPはあらゆる処理を実行できる」という単純な考えに基づいて、このような種類の5250プログラムをあらゆる刷新の概念実証作業に含めることは、説得力があり論理的です。

しかしその結果、このプログラムの特性を理解すること、およびこのプ ログラムのスクリプトを作成することに多くの時間を費す場合がありま す。これを処理するために過剰な時間を費やしたり、また刷新する必要 のある他の数百または数千もの重要な5250プログラムからすべての注意 がそらされることがなければ、それでも良いのですが…。このような場 合、落とし穴に陥る可能性があります。

- *ISV*サイトでは、「可能性のある顧客に刷新した製品を適切に披露 するために、新しいVisual LANSAコンポーネントに真っ先に変更す るプログラムは何か」を検討する必要があります。
- 社内開発サイトでは、「新しいVisual LANSAコンポーネントに変更した場合に、エンド・ユーザーの生産性と利用性を最も向上させることのできるプログラムは何か?」、また、「より優れたものに置き換えられたプログラムで、刷新プロジェクトに対してほとんどの経営者とエンド・ユーザーからサポートを集めるプログラムは何か?」を考える必要があります。

ほとんどの場合、この2つの回答は共に、最も巨大で厄介な5250プログラムになることでしょう。

なぜ、より優れたものに置き換えることを検討しないのか? これが本当の場合、次の質問は、「なぜ、このプログラムを再利用する ことにこれだけの時間と労力を費して、より優れたものに置き換えるこ とを開始しないのか?」となるはずです。

その理由は明らかに、置き換えに関する時間と費用をかけないようにす

るためです。

ただし、さまざまな市場、ビジネス、政治的な理由ですぐにでも置き換える必要があるという製品化の場合、それをただちに実行することを真 剣に検討して、再利用することに過度の時間を費やすことなく、また刷 新する必要のある他のすべての5250プログラムの損失が注目の的になら ないようにする必要があります。 アプリケーションはどのくらいの期間でRAMP化できるか?

重要:この答えは、既存の5250画面を再利用するRAMPステージ2 のみに関係します。5250画面をVisual LANSAコンポーネントに置 き換えるRAMPステージ3には関係しません。

採用する手法に依存します。

次のような4つのメニュー(または他の共通アクセス・ポイント)と36個の 他の画面で構成される単純な5250アプリケーションを考えます。



次の手法を採用することをお勧めします、

1. 最初に、高速ナビゲーションの刷新を実行します

この例では、A、B、C、Dの4つのメニュー(またはアクセス・ポイント) のみを識別および定義し、それらを完全な画面目的としてRAMPにス ナップインします。

これで、ナビゲーションが刷新されている5250アプリケーション全体 を、エンド・ユーザーに配布できます。

通常は、アプリケーション自身(少なくとも一部)を完全に刷新して、ア プリケーションの価値を高めることも行います。

このステージで「アプリケーションはどのくらいの期間でRAMP化できるか?」という質問に答えることは簡単です。1つのメニュー(または共通アクセスポイント)に付き15分です。したがって、この例では、1時間(4x15分)になります。

2.次に、選択したアプリケーションを刷新したり、刷新したア

プリケーションを付加します。 ここでアプリケーション領域A、B、C、Dを評価します。

- それぞれの使用頻度はどれくらいですか?
- 完全な刷新を行うことで、ユーザーの生産性は向上しますか?方法は?必要な処理は何ですか?
- 完全な刷新を行うことで、ユーザーの操作性は向上しますか?方法は?必要な処理は何ですか?
- 完全な刷新は、製品のデモンストレーションやマーケティングを支援しますか?方法は?必要な処理は何ですか?

以上の評価に基づいて次のような決定を下す場合があります。

- アプリケーション領域Aを刷新し、それを初期リリースとしてエンド・ユーザーに配布する。
- その後、アプリケーション領域Cの60%を刷新し、それを新バー ジョンとしてユーザーに配布する。
- アプリケーション領域Dの完全な刷新は行わない。ビジネス価値が 向上しないため。
- 最後にアプリケーション領域Bの25%を刷新し、最終バージョンを エンド・ユーザーに配布する。

「アプリケーションはどのくらいの期間でRAMP化できるか?」という質問の答えは、この手順で決定する刷新の方法により異なります。 質問には、完全な刷新を行う部分や、刷新に必要な仕事量とその順番な どが決まるまで答えることはできません。

キー・ポイント

- ナビゲーションは非常に短時間で刷新できる。
- アプリケーションの刷新には時間がかかるが、価値が大きく向上する。
- 刷新した5250アプリケーションは付加的に配布できる。1度にすべてを刷新する必要はない。
- RAMPで使用するためだけに、すべての5250アプリケーションの完 全な刷新を強制されない。
- 一部のアプリケーションについては完全な刷新を行わず、後で新し

い Visual LANSA コンポーネントに置き換えることができる。
### 再利用、修正、または書き換え

RAMPのコンテキストでは、再利用(5250画面)と修正(画面ラッパーを使用)と書き換え(新しいVLコンポーネント)に関する決定は複雑です。

次回、「1週間で書き換えできそうだ」と誰かが言った場合は、次の点を 考慮していることを確認する必要があります。

- 1. 既存のすべての機能とインターフェースを確認して理解するための 時間
- 2. 刷新されたUIを使用して再設計するための時間
- 3. コードを作成してユニット・テストを行う時間(このステップには前述のように実際に1週間かかる可能性あり)
- 4. 既存のすべての機能とインターフェースを(再)テストするための時 間

「それを1週間で書き換えることができます」と誰かが言ったとすると、 多くの場合、テストに2、3倍の時間が必要になることを念頭に置いてく ださい。2倍や3倍の係数を適用しても、実際の書き換えでは10倍以上に なることがあります(つまり、手順3はジョブ全体の10%未満になること があります)。

手順1と4の完了に必要な時間が根本的に削減されるため、チャム/ラッパー・ソリューションは役立ちます。

また、何らかのテンプレート/ウィザード/コード生成の使用を検討して いる場合は、手順3のみが迅速化されることを覚えておいてください。 つまり、手順3がジョブのわずか10%であり、それを2倍迅速に行って も、プロジェクト全体の50%ではなく5%しか改善されません。

# RAMP Windowsのチュートリアル

RAMP刷新プロジェクトは、次のように進める必要があります。反復リ リース・サイクルを使用して、時間やリソースが許す限り、繰り返しア プリケーションの機能を強化することができます。

RAMPス テージ1	↓プロトタイ プ	RAMP-TS001:刷新されたアプリケーショ ンの基本的なプロトタイプの作成 RAMP-TS005:デザインの検証
	↓標準の設定	プロジェクト標準を作成します。
	↓ セキュリ ティの決定	セキュリティ/権限ポリシーを設定します。
	↓画面の命名	RAMP-TS002:アプリケーション・セグメ ント全体の迅速な刷新 RAMP-TS004:従業員画面の命名と分類
	↓画面の分類	RAMP-TS002:アプリケーション・セグメ ント全体の迅速な刷新 RAMP-TS004:従業員画面の命名と分類
RAMPス テージ2A	↓迅速な刷新	RAMP-TS002:アプリケーション・セグメ ント全体の迅速な刷新
RAMPス テージ2B	↓個別の刷新	RAMP-TS006:基本的な照会画面のスナッ プイン RAMP-TS007:データ入力ファンクション のスナップイン RAMP-TS008:照会画面の更新モードへの 変更 RAMP-TS009:ナビゲーションの追跡 RAMP-TS015:画面バリエーションの概要 と処理
RAMPス テージ2C	↓ 価値の付加 参照:価値を 付加するもの	RAMP-TS003:社員のデータ・フィルター の作成 RAMP-TS010:特別なフィールド・ハンド

		ラーの使用
		RAMP-TS011 : 付属のドキュメント・コマ ンド・ハンドラーのスナップイン
		RAMP-TS012:付属のNotesコマンド・ハン ドラーのスナップイン
		RAMP-TS013:インスタンス・リストの データをExcelに送信する
		RAMP-TS014:HRデモ・アプリケーション へのRAMP画面のスナップイン
		RAMP-TSAD04:aXesを使用して画面を再 設計する
RAMPス テージ3	<b>↓ プラット</b> フォーム依存 関係の排除	IBM iプラットフォーム依存関係を排除しま す(オプション)。
	↓テスト	最終的なアプリケーションをテストしま す。
	↓配布	チェックリスト/計画シート => WIN-RAMP- TS
	← 繰り返し	ステージ2Aから、2B、2C、3まで、すべて のサイクルで、アプリケーションのコン ポーネントを次々と移動します。 ステージの組み合わせや比率は、プロジェ クトにかかる時間に影響を及ぼし、利用で きる時間やリソースの量によって決まるた め、非常に重要です。

「高度なチュートリアル」も参照してください。

## チュートリアルを使用する前に

### チュートリアルの対象者

このチュートリアルは、RAMP-TSの使用方法の学習を希望する初級から 上級までのLANSA開発者を対象としています。高度なVisual LANSAの知 識は必要はありません。Web開発用フレームワークを使用する場合、 LANSA for the Webトレーニングが必要です。

#### チュートリアルの使用方法

- チュートリアルは順序通りに修了することをお勧めします。作成するア プリケーションのスタイルに関連する実習を完了します。Windowsアプ リケーションのみを作成する場合、WEBEVENTとWAMに関する実習を スキップすることもできます。
- 複数の開発者がチュートリアルを使用できるように、すべてのLANSAオ ブジェクトにiiiで始まる名前を付けます。iiiには、名前のイニシャルな ど、任意の3つの文字が使えます。例えば、名前がJohn David Smithであ ればJDSの文字を使用します。iiiCOM01という名前のコンポーネントの 作成を要求された場合、JDSCOM01という名前のコンポーネントを作成 します。iiiは必ず固有の3つの文字で置き換えてください。
- Visual LANSA のライセンスが無い場合やトライアルバージョンを使用している場合、DEMを使用してiiiを置き換える必要があります。iiiCOM01という名前のコンポーネントの作成を要求された場合、DEMCOM01という名前のコンポーネントを作成します。

### 使用区画

RDMLXが有効な区画を使用する必要があります。

チュートリアル用にDEM区画の使用をお勧めします。DEMシステムには、チュートリアルで使用する人事システムのデモンストレーション・ファイルとすべての必要なファイルが含まれます。

WEBEVENTアプリケーションを作成する場合、ファンクションで RDMLXを有効にしてはなりません。

チュートリアルのインストール

チュートリアルを使用するには、Visual LANSAフレームワークとRAMP-TSがインストールされている必要があります。「インストールおよび構成」を参照してください。

チュートリアルでは、人事デモンストレーション・システムのファイル (区画の初期化によりインストール)が必要です。 トレーニングを使用できる開発者数

同時にトレーニングを使用する開発者数に制限はありません。ただし、 各開発者が固有のiiiIDで作業することが重要になります。

フィードバック

このチュートリアルに対する皆様のご意見・ご感想をお聞かせくださ い。LANSAオンライン・マニュアルおよびトレーニングの質の向上に活 用させていただきます。ご意見・ご感想はlansatraining@LANSA.com.au 宛に電子メールにてお送りください。 RAMP-TS001:刷新されたアプリケーションの基本的なプロト タイプの作成

刷新されたアプリケーションのプロトタイプを作成することは、RAMP を使用するときに実行する必要がある最初のステップです。完了した結 果の表示、振る舞い、感じがどのような見通しになるか、実際の刷新化 ステップの前に確かめることができます。

このプロセスは、設計を検証し、新しい、あるいは潜在的なビジネスの ニーズをすばやく検証する手段として機能します。

目的

- アプリケーションの刷新の成功におけるこのステップの重要さを理 解する
- フレームワーク・プロトタイプ作成ツールの使用方法を理解する



この目的を達成するには、以下のステップを完了してください。

- 刷新前のアプリケーション
- RAMP-TS001 ステップ1. アプリケーションのプロトタイプの作成
- RAMP-TS001 ステップ2:コード・テーブル・プロトタイプの変更
- RAMP-TS001 ステップ3:社員プロトタイプの検証
- RAMP-TS001 ステップ4:従業員情報へのエンド・ユーザー・アク セスのプロトタイプ作成
- RAMP-TS001 Step 5. フィルターの表示
- AMP-TS001 ステップ6. 基本的なプロトタイプの検証
- まとめ

刷新前のアプリケーション

このチュートリアルでは、Personnel Tables (コード・テーブル)と人事シ ステム・メンテナンス・アプリケーションの刷新を行います。

Personnel Tablesアプリケーションは、テーブル保守メニューとさまざまなテーブル・メンテナンス・ファンクションで構成されています。



Personnel Tablesアプリケーションは使用頻度が低いため、単にメイン・ メニューをフレームワーク内のスナップインして、刷新を行う予定で す。

人事システム・アプリケーションで刷新する部分は、社員の資格のメン テナンスと新しい社員の登録です。

Personnel System Main Menu	Enrol a New Employee
Enter number of function required or place cursor on same line. 1. Full Employee Listing 2. Enrol a New Employee 3. Browse/Naintain Employee and Skill Files 4. Employee Business Phone Number List 5. Perdorm General Employee Search 7. Telephone Number Search 8. Dept/Section/Employee Window Enquiry 9. Personnel Table Maintenance Menu 10. Display HELP Text 11. Exit from Application	Employee Number
Browse/Maintain Employee and Skill Files Employee Number <u>A1234</u> +	Browse/Maintain Employee and Skill Files         Employee Number       A1234         Employee Surname       JACK         Employee Given Name(s)       STEPHENS         Street No and Name       6 Melissa Place         Suburb or Town       Mest Pennant Hills         State Constraints       2125         Home Phone Number       (82) 9871 7773         Denactement Code       0
	Department Code       20       *         Section Code       ES +         Start Date (DDMMYY)       14/08/95 +         Termination Date (DDMMYY)       0/00/08 +         Date Skl Skill       Skill         Acquired Code       0/00/08 +         0/00/01 AC       Accountancy Degree       NOT BAD       D         0/00/01 ADENIN2       Administrath Part 2       VERY GODD       D         5/05/94 ENG       English Degree       With distinction       D
	F1=Help F3=Exit F4=Prompt F12=Cancel F14=Hsgs

人事システムは使用頻度が高いため、このアプリケーションの刷新に重 点的に取り組みます。

# RAMP-TS001 ステップ1. アプリケーションのプロトタイプの 作成

アプリケーションを刷新する最初のステップは、そのプロトタイプを作 成することです。プロトタイプを作成するには、アプリケーションで処 理するビジネス・オブジェクトを特定する必要があります。これを実行 するには、エンド・ユーザーがシステムが何を処理するかを説明すると きに使用する言葉を調べます。ほとんどの場合、この言葉は、アプリ ケーションメニューと画面タイトルに反映されます。

サンプル・アプリケーションでは、ユーザーがコード・テーブルと社員 を処理するので、Code TablesおよびEmployeesビジネス・オブジェクトを 作成します。

#### 1. LANSA開発環境にログオンします。



## 2. LANSAエディターで、設計者としてフレームワークを起動します。

ツール	<u>፲</u> ሳ / ኦዮጵያ 🖤 🗤 🖸			
	構成( <u>C</u> )			
	配布 ツール			
	ロシ"カル モテ"ラー			
ĸ	ウィザート"			
	VL フレームワーク - 設計者モートヾ( <u>D</u> )			
	VL フレームワーク - 開発者モード(⊻)			
	VL フレームワーク - 管理者モード( <u>A</u> )			
	VL フレームワーウ ー ユーザーモード(凹)			
*	Visual LANSA エラーログ			
1	IBM i エラー ログ			
	Web 実行時エラーログ			
	CS Builder/クライアント			
	Integrator スタジオ			
ø	LANSA インホ <sup>®</sup> ートΦ			
	テキスト 検索			

インスタント プロトタイピング アシスタントを起動して、新しいビジネス・オブジェクトを作成し、これらを含むアプリケーションを作成します。

(71	-47-5) (管理)	
	(新規作成)	•
	(プロノパティ)	
	(アフツケーション)	•
	( גאַעקב)	
	( ))	
	(デザインコートデーフル…)	
	(コード ウィザード )	
	( ብህスタント ጋግトタイピング アシスタント )	
	( RAMP ツール )	
	(仮想ソリップボード)	•
	(マージ ツール )	
	(保存)	
	(名前を付けて保存)	
	(保存して再起動)	
	(保存して終了)	
	(Webアプリケーションとして実行…)	•
	(Web コンソール )	•
	(^#7°)	•
	(トレース)	•

 Code TablesとEmployeesという名前の新しいビジネス・オブジェクト を作成します(名前をコンマで区切ります)。

メインビジネス オフジェクト:	Code Tables,Employees	
前回の値を回復		
🥔 Code Tables 🥔 Employees		
	· 太へ >>	キャンセル

5. [次へ]をクリックします。

- 6. [詳細]コマンドをビジネス・オブジェクトにドラッグして、Code Tablesに関連付けます。
- 7. [アクション]フィールドに入力し、新しい2つのコマンド[ドキュメント]と[イベント]を作成します(コンマで区切ります)。
- 8. [詳細]、[作成]、[ドキュメント]、[イベント]コマンドを[Employees] に関連付けます。



- 7. [次へ]をクリックします。
- 8. アプリケーションのリストに入力して、人事という新しいアプリ ケーションを作成します。

最後にCode TablesとEmployeesを人事アプリケーションに追加します。

9. Code TablesおよびEmployeesビジネス・オブジェクトをドラッグして、人事アプリケーションにドロップします。

63	フ゜ロク゛ラミンク゛ テクニック	アプリケーション(既存)
6	HR デモアプリケーション	アプリケーション(既存)
6	お気に入り	アプリケーション(既存)
6	管理	アプリケーション(既存)
6	DEM HR	アプリケーション(既存)
🖃 🙆	人事	アフ°リケーション
	👍 Employees	ヒッショネス オフッショェクト
	迈 Code Tables	ヒッショネス オフッショェクト

- 10. [次へ]をクリックします。
- 11. [終了]をクリックします。

人事アプリケーションがフレームワークで作成されました。



# RAMP-TS001 ステップ2:コード・テーブル・プロトタイプの 変更

このステップでは、Code Tablesビジネス・オブジェクト用に作成された デフォルトのフィルターを削除してそのプロパティを変更します。単に 既存の5250画面をフレームワークに添付してアプリケーション・セグメ ントを迅速に刷新している場合、フィルターは不要です。

- 1. [Code Tables]をダブルクリックしてそのプロパティを表示します。
- 2. [フィルター]タブを表示します。
- 3. デフォルトで作成されている[フィルターの作成]を選択し、[削除]をク リックして削除します。

👹 ビシネス オブショントのフロパティ	- Code Tables		
識別子 アイコン ビジュアル スタイル	7ィルター 7ィルター設定 使用可能コマ	ンド コマンドの表示 ユーザー プロバティ サブタイプ インスタ	C 📲 🕨
二フィルターの作成	識別子 アイコン フィルター登録の言	定	
	キャプション	フィルターの作成	(JPI
	EDH:		(JPI
	順序	2	
	内部識別子:	50F864E4D52B46BF9DC3D7C06F355056	
	2-ザー オブジェクト タイプ	50F864E4D52B46BF9DC3D7C06F355056	오
	Web の使用を許可		
	👽 Windows の使用を許可		
	最終変更	20100702-170815-V20PGMLIB	
	RAD-PAD 77114名	C:#PROGRA~1#LANSAV12#X_WIN95#X_LANSA#x_dem	¥execi
新規作成 削除			
		- PP	じる

フィルターがない(つまりインスタンス・リストがない)ため、[詳細]コマ ンドの定義を変更する必要もあります。

- 4. [使用可能コマンド]タブを表示します。
- 5. [詳細]コマンドを選択し、コマンド・タイプをビジネス・オブジェ クト・コマンドにします。

👹 ድንጎትス አንንታታትወን ከለንታና – Co	de Tables	
識別子 アイコン ビジュアル スタイル フィルター 使用可/不可の設定を切り替えるにコおリスト からドラッグ アンドドロッフ します	<ul> <li>フィルター設定 使用可能コマント* コマント*の;</li> <li>□ 詳細 (DETAILS)</li> </ul>	表示   ユーザー ブロバティ   サブタイブ   インス究 ◀ 🕨
選択可能コ AJAXEX1 AJAXEX2 AJAXEX3 ASSIST ASSIST ASSIST COMMAN CRUD e-mail Example 1 Example 2 Example 3 Example 4 Example 5 Example 6 Example 7 Example 8 Fax	コマント・タイクを選択して下さい         ・ビジネスオフジェクトコマント・         順序         コマント・オフション         常にアクティク・         省略値         当客地値         当客地値         当客地値         コマント・オフション         第にアクティク・         当略値         コマント・オフション         第二         第時値         コマント・オフション         第二         131         マ         Web の使用を許可         マ         米沙コ・アケック・メニューの表示         マ         キュウールハニー上に表示         他のコマント・ション・         アクセス制限         非表示コマント・として実行	ウィントゥのサイズ     幅     高さ       ウィントゥウ     幅     高さ       ウィントゥウ     ・     ・       ソンスタンス コマント。     ・     ・       ウィントゥウ     ・     ・     ・       ウィントゥウ     ・     ・     ・       ウィントゥウ     ・     ・     ・       ウィントゥウ     ・     ・     ・       シャラ     ・     ・     ・       文字引数 1:     ・     ・       数字引数 2:     ・     ・
Mritter		用じる

フレームワークを再起動するかどうかを確認するメッセージが表示され ます。

注意:	
◆ 行なわれたデザイン変更を有効にす るためにはフレームワークを保存、終 了、再実行する必要があります。	開じる 保存して再起動
■ このような変更を行なう場合に通知	

- 6. [このような変更を行う場合に通知]メッセージ・ボックスをオフにし、[閉じる]ボタンをクリックします。
- 7. プロパティ・ウィンドウを閉じます。
- 8. フレームワークを保存します。

# RAMP-TS001 ステップ3: 社員プロトタイプの検証

- このステップでは、Employeesビジネス・オブジェクトのプロトタイプ を検証します。
- 1. 人事アプリケーションの[Employees]をクリックして、プロトタイプ を表示します。



- 2. フィルターの[検索のサンプル]ボタンをクリックして、インスタン ス・リストにサンプル・データを入力します。
- インスタンス・リストのいずれかのサンプル社員をクリックして、
   それに関連付けられているコマンド・ハンドラーを表示します。



[作成]コマンドのコマンド・ハンドラーがその他のコマンド・ハンド ラーとともに表示されていないことがわかります。これは、デフォルト でビジネス・オブジェクト・コマンドとして設定されているためです。 4. インスタンス・リストのいずれかの社員を右クリックし、コンテキ スト・メニューを表示して[作成]コマンドを選択します。



社員の作成のコマンド・ハンドラーが表示されます。

- 一般に、インスタンス・リストとは関係のない別のウィンドウにビジネ ス・オブジェクト・コマンドを表示したいと考えます。このため、別の ウィンドウに表示されるように[作成]コマンドの定義を変更します。
- 5. Employeesビジネス・オブジェクトをダブルクリックし、そのプロパ ティを表示します。
- 6. [コマンド表示]タブで、[オブジェクト コマンド表示]を[別の標準 ウィンドウ]に変更します。

識別子 アイコン ビジュアル スタイル フィルター	7/11月一設定(使用可能コマント、コマントの表示):	ユーザー フロハ
コマント・タフ・のスタイル:	97°	
全てのコマンド タブの表示:	自動	¥.
コマント、タフ、の位置:	F	Ψ.
オフジェクトコマンドの表示	新しい標準ないがり	
インスタンスコマンドの表示	ウィンドウの一部を使用	w.
複数行のタブシート キャブション		
▽ 切り離し可能		

- 7. [Employees]プロパティを閉じます。
- 8. [作成]コマンドを選択します。これは別のウィンドウに表示されま す。

	Employees
	このパネルはビジネスオフジェクト Employeesのアクジョン(またはコマント) 作成 を扱います。
10801	この後の作業で、このコマンドハンドラーを実際のプログラムに取り替えることができます。
	このパネルは入力可能です。テキストを削除後、独自のテキストを追加できます(RAD-PADから絵 やイメージも追加できます)。
	ここで追加したテキストは保存され、デザインを具体化したり、機能を他者に説明したりするために 使用できます。
	詳細を表示 コート <sup>*</sup> ウィザート <sup>*</sup> RAD-PAD
	H:325 W:491 メッセージ レコートドサイス*

9. [Employees]ウィンドウを閉じます。

次のステップでは、最初のプロトタイプを開発してすべての関係者が理 解できるようにします。 RAMP-TS001 ステップ4:従業員情報へのエンド・ユーザー・ アクセスのプロトタイプ作成

このステップでは、フィルターを作成して従業員情報にアクセスできる 種々の方法のプロトタイプを作成します。 従業員情報をどのように取得したいのかエンド・ユーザーの意見を引き 出すために、定義されている論理ファイルに関係なく、エンド・ユー ザーに役に立つと思われるさまざまな方法で、データを検索するプロト タイプ・フィルターを提供する必要があります。 このステップは、エンド・ユーザーに価値を付加できるようにアプリ ケーションを刷新するうえで非常に重要です。

- 1. Employeesビジネス・オブジェクトをダブルクリックし、そのプロパ ティを表示します。
- 2. [フィルター]タブを表示します。
- 3. [新規作成]ボタンを使用してさらに2つのフィルターを作成します。
- 4. 最初のフィルターのキャプションを[名前]にします。
- 5. その他のフィルターのキャプションを[By Date of Birth]と[By Salary] にします。
- 6. [Employees]プロパティを閉じます。

社員のフィルターが表示されます。



### RAMP-TS001 ステップ5. フィルターの表示

このステップでは、最初のプロトタイプの開発を開始してすべての関係 者が理解できるようにします。フィルターでRADPADを使用し、フィル ターのおおよその外観を表示します。

- 1. [By Date of Birth]フィルターを表示します。
- 2. フィルターのすべてのテキストを選択します。

de Employees		×
ファイル(E) 編集(E) 表示(V) ヘルプ()	1) ウィントウ(₩) (フレームワーク)(管理)	
④作成	🖹 🖹 🛕 🔟 🗖 🕼 🕄 🗌 🎯 🖾 🍇 🕸 🗐 🛛 🚧 🕅 外分検索	
Employees		-
× ■ DEM HR ERP Prototype 1 HR デモアプツケーション Human Relation My Application ま気に入り Code Tables Employees M 0 7°bン゙ラミングテウニック 管理	名前       By Date of Birth       By Salary         マイルターのプロトタイプとして、アプリケーションがどのように見、 えるかを暗認してください。       Employee       記述         Employee       記述         この後の作業で、このフィルターを実際のフィルターに取り替える ことができます。       EMPLOYEE0001       Employee …         EMPLOYEE0003       Employee …         EMPLOYEE0004       Employee …         EMPLOYEE0005       Employee …         EMPLOYEE0006       Employee …         EMPLOYEE0007       Employee …         EMPLOYEE0007       Employee …         EMPLOYEE0007       Employee …         EMPLOYEE0007       Employee …         EMPLOYEE0008       Employee …         EMPLOYEE0009       Employee …         EMPLOYEE0009       Employee …         EMPLOYEE0009       Employee …         EMPLOYEE0009       Employee …	×
	🤞 Employee:詳編 (EMPLOYEE0005-Employee number 5)	×
	■ 詳細 ▲ イベント ドキュメント このパネルはビジネスオブジェクト Employeesのアクション(またはコマンド) 詳細を扱います。 この後の作業で、このコマンドハンドラーを実践のプログラムに取り替えることができます。 このパネルは入力可能です。テキストを削除後、独自のテキストを追加できます(RAD-PADから絵やイメージも追加できます)。 詳細を表示 コードウィザード RAD-PAD	<
	メッセージ 準備 ASID02_TS JPN V20PGMLIB 10/07/02 18:22 🥥	

- 3. テキストを削除します。
- 4. イメージ・パレットを使用して、テキスト挿入画像を入力し、フィ ルターのプロトタイプを作成します。例:

▲ 名前 → By Date of Birth → By Salary →
It can be used to, for example, identify all employees who'sbirthday is coming up.
Search:
O Birthday next week
O Date
O Date range 27JAN1964 1 27JAN64 1
コートドウィザート・  RAD-PAD 検索のサンフッル

- 5. 他の2つのフィルターのプロトタイプをすばやく作成します。
- 6. フレームワークを保存して再起動します。

# RAMP-TS001 ステップ6. 基本的なプロトタイプの検証

刷新されたアプリケーションのプロトタイプ作成の最初のステージが完 了しました。このプロトタイプは、刷新後のアプリケーションの基本的 な構造と機能を示しています。



実際の状況では、この時点でアプリケーションの関係者に基本的なプロ トタイプを示して、以下の点を確認する必要があります。

- 選択したビジネス・オブジェクトについて関係者が理解でき、受け 入れ可能であること。
- ビジネス・オブジェクトに関連付けたコマンドに、関係者が情報を 操作する方法が反映されていること。
- フィルターで提供される情報へのアクセスが実用的で適切であること。

実際のアプリケーションのプロトタイプを作成しているときは、この時

点でエンド・ユーザーにプロトタイプを試用してもらいます。システム が目に見える形になっていれば、ユーザーが実際にデータを入力してみ るのも容易でしょう。

フィードバックを収集した後、最初のプロトコルを調整します。

プロトタイプは、ステップRAMP-TS005:デザインの検証で洗練されます。

まとめ

重要なポイント

以上で、刷新されたアプリケーションの基本的なプロトタイプを完了し ました。プロトタイプを使うと次のようなことができます。

- 設計上の問題点がないかどうか検証する。
- エンド・ユーザーなどに試用してもらい、意見を集める。
- 要求仕様を満たしていない事項があればすぐに設計し直すことができる。
- 問題が見つかった場合に代替案を検討できる。

### 覚えるべき事柄

- フレームワークでアプリケーションとビジネス・オブジェクトを作 成する方法、およびコマンド・ハンドラーをビジネス・オブジェクト に関連付ける方法
- フレームワークのプロパティを設定する方法
- RADPADとイメージ・パレットを使用してプロトタイプを表示する 方法

RAMP-TS002:アプリケーション・セグメント全体の迅速な刷 新

アプリケーションの5250画面ごとにRAMPを使用する必要はありません。多くの場合、アプリケーション・セグメント全体が使用されることはあまりないため、これらを一般にフレームワーク内にきわめて短時間でスナップインすることができます。

この方法を使用する時期と方法の理解は、RAMPプロジェクトの完了に かかる時間に影響を与えます。

目的

- アプリケーションの使用頻度の低い部分を迅速に刷新する方法を示す
- アプリケーションをRAMP化する基本的なステップに習熟する
- RAMP-TSで使用される主要概念の一部を紹介する



この目的を達成するには、以下のステップを完了してください。

- 開始する前に
- RAMP-TS002 ステップ1:画面の命名
- RAMP-TS002 ステップ2:画面の分類とナビゲーションの追跡
- RAMP-TS002 ステップ3: [キャンセル]および[終了]ボタンの削除
- RAMP-TS002 ステップ4:フレームワークへのアプリケーションの スナップイン
- まとめ

始める前に

RAMPシステムをセットアップする必要があります。詳細については、 「インストールおよび構成」を参照してください。

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

• RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成 開始する前に

RAMP-TSの使用を開始する前に、いくつかの重要な事項を念頭に置く必要があります。

すべてで大文字と小文字が区別されることに注意してください。

RAMP-TSは、大文字と小文字が区別されるJavascriptを使用します。画面 やフィールドの命名、変数、オブジェクト、ファンクションの作成や呼 び出しのときは、大文字と小文字の使い分けを慎重に確認する必要があ ります。

問題が発生したときは、まず大文字と小文字の使い分けを確認する必要 があります。

#### RAMP-TS 5250セッションを使用する

RAMPツールの5250セッションは厳密に追跡および監視されます。これ は以下を意味します。

- RAMP-TS 5250セッションは、実行時の5250セッションよりも大幅 に低速で実行されることがあります。
- 5250の設計時の画面の応答があるまで待機する必要があります。応答よりも先に入力しないようにしてください。画面アクションは、 ゆっくりと正確に行います。

Axes開発者モードの5250セッションは、Axes開発作業のみを実行するように設計されています。つまり、Axes以外の開発作業(RPGプログラムの編集など)以外に、Axes開発者モードの5250セッションを使用するべきではありません。

エンド・ユーザー環境でAxes開発者モードの5250セッションを使用してはなりません。

#### 5250アプリケーションの動作を理解する

画面をRAMPで処理するときは、5250アプリケーションの動作方法を扱うスクリプトを生成します。

通常、5250アプリケーションは、(望ましくは)設計の規則に従い、一貫 した5250 UIを生成しようとしている開発者によって設計されるため、動 作にパターンがあります。このため、RAMPプロジェクトでは、

- アプリケーションのこれらの動作パターンを見つけて、スクリプト 内で適応させる必要があります。
- 最も避ける必要があるのは、例えば、200個のスクリプトを作成した後に意外な動作や予期しない動作が見つかることです。その理由は、元に戻ってすべてを変更しなければならないためです。
- 代表的な10個の5250画面をRAMP化した後など、これらの画面を非常に厳密にテストして、プロジェクトで意外なことや予期しないことが後になって見つかる可能性を最小限に抑える必要があります。
- 理解していることでプロジェクトが損なわれることはありません。
   最悪の場合、プロジェクトがさらに複雑になってしまうことがあります。これは理解していないために発生します。したがって、理解していないことを最小限に抑えるように努力する必要があります。

## RAMP-TS002 ステップ1:画面の命名

このステップでは、アプリケーションの画面に名前を付けます。ナビ ゲーション(該当する場合)に使用する画面やフィールドに名前を付ける ことは、RAMPでアプリケーションを刷新するときに常に実行する必要 がある最初のステップです(ナビゲーションを追跡してはいけません。ま た、同時に画面にも名前を付けます)。

- 1. フレームワークが実行されていない場合は、起動します。
- 2. フレームワーク・メニューの[RAMPツール…]オプションを選択し、 RAMPツールを起動します。

(71	<mark>←ムワーク)</mark> (管理)	
	(新規作成)	•
	(プロノパティ)	
	(アフ <sup>¶</sup> ケーション)	+
	(אַרָאָב )	
	() <u>,</u> )	
	(デザインコードテーフル…)	
	(コート <sup>®</sup> ウィザ <sup>。</sup> ート <sup>®</sup> … )	
	(	
	(RAMP ツール)	
	(仮想クリップボード)	×
	(マージ ツール )	
	(保存)	
	(名前を付けて保存)	
	(保存して再起動)	
	(保存して終了)	
	(Webアフツケーションとして実行…)	+
	(Web コンソール )	+
	(~µ7°)	+
	(トレース)	+

RAMPウィンドウが表示されます。

3. メッセージ・エリアで、RAMPツールが起動されていないことを示 すメッセージをクリックし、オプションを選択して起動します。

F	AMP-TS 5250 セ	がり 詳細			
	下のメージをクリー	<b>妙し、可能な</b>	アクションを見て下る	きしい	
	1 このうく)をか	わしーデー EDE0= スタート 525	표현대도도 51 57777 50	*#*開始してください	
		オンライン カ	讣を表示		
	<	Ш			>

4. RAMPセッションが開始されたら、サーバー・システムに接続しま す(デフォルトのRAMP-TS開発者認証のユーザーIDとパスワードは*dev* と*dev*ですが、システム値は変更されていることもあります)。

へ接続	? 🛛
	E C
のサーバー 警告: このサーバーは、ユ することを要求しています	にはユーザー名とパスワードが必要です。 ーザー名とパスワードを安全ではない方法で送信 (安全な接続を使わない基本的な認証)。
ユーザー名( <u>U</u> ): パスワード( <u>P</u> ):	
	OK キャンセル

[RAMP-TS 5250エミュレータ セッション]ウィンドウが表示されます。

🔲 RAMP ツール - デウォルト セッジ	אין (RAMP-TS)
テラオルト セッション	RAMP-TS 5250 セッション 詳細
保存 名前 ■ セッション ■ Junction (1) ■ Destination (2)	下のメッセージをクリックし、可能なアクションを見て下さい フォームを命名 MainLogin として定義 Junction このフォームに名前付きのアイテムがありません このスクリーンへナビゲートできるスクリーン・・・ このスクリーンがナビゲートできるスクリーン・・・
שם Destination (0) שם Special (1) שם גאשריא (2)	
	Keymaps       AutoGUI       Screens       Session       Display       Tracking Info         一致の候補       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●       ●
	画面の定義     保存 キャンセル 削除 自動定義     User:     Land     Sai:     Abi:     MainLogin     別名:     Compared     Added     Ad

トラッキング情報エリアで、サーバーのログイン画面にデフォルト でMainLoginという名前が付けられていることがわかります。これは、デ フォルトのRAMP名が付けられている唯一の画面です。

5. アクション・タブの表示/非表示ボタンをクリックしてタブを非表示 にします。

eens		Session Display Help
7.17		アクション・タブの表示/非表示ボタン
		User:
		Passwor
	eens <b>X17</b>	eens

### 6. サーバー・システムにログオンします。

表示される最初の画面は、次のプログラム・メッセージです。



画面トラッキング・エリアに、画面の名前が付けられていないことが示されます(Unknown Form)。

また、[RAMPツール]ウィンドウの左上に別のフローティング・ウィン ドウ(aXesデザイナー)が表示されていることがわかります。


[aXesデザイナー]ウィンドウは、チュートリアル「RAMP-TSAD04: aXesを使用して画面を再設計する」で使用します。ここでは、できる だけ小さいサイズに設定してください(以降のチュートリアルでもこの ように操作します)。



6. 画面に名前を付けるには、[RAMP-TS 5250]ウィンドウの左上隅にあ るアクション・タブの表示/非表示ボタンをクリックします。



7. 表示されたタブ・フォルダーの[Screens]タブを選択します。



8. 画面にDisplayMessagesという名前を付けます。

Keymaps AutoGUI Screens	
一致の候補	
ID 名前 7.17	
4 DisplayMessages 0 へ <新しい定頼>	
最有力候補:DisplayMessages (4)	
<b>画面の定義</b> 保存 キャンセル <b>削除</b> 自動定義	
署名: A.QDDSPEXT.HEADER.INFFMT	
文督: DisplayMessages (ID = 4)	
別名:	
<b>記述:</b>	
< ×	
サフ・ファイル:	
開始 称了 名前 「行/エント 見出し リー 行	
N.	Y

9. [保存]ボタンをクリックします。



トラッキング情報に画面の名前が表示されます。



- 10. [セッション]ウィンドウにフォーカスを移動し、Enterキーを押して メッセージを閉じます。i5/OS Main Menuが表示されます。トラッキン グ情報で画面に名前が付けられていないことが示されます。
- 11. メッセージ画面に名前を付けたときと同じ方法で、これ にi5OSMainMenuという名前を付けます。

また、入力されたコマンドを追跡できるようにコマンド・ライン・フィー ルドに名前を付ける必要もあります。

- 12. [画面]タブのフィールドのリストで下にスクロールします。
- 13. コマンド・ライン・フィールドを見つけます。

Keymaps AutoGUI	Screens	K Session Display Help	🔎 🕞 - Auto - 🗾 🔍 🤯
<u>=Cxe</u> -	<ul> <li>×</li> </ul>	MAIN SYSTEM I メインメニュ 次の 1 つを選択してください。	- システム : ASI002
<del>サ)*ファイル:</del> 開始	行/エント 見出し リー 行	1. ユーザー・タスク 2. オフィス・タスク 3. 汎用システム・タスク 4. ファイル,ライブラリー,およびフォルダー 5. プログラミング	
7 <b>~~)<sup>5</sup>: 名前</b>	Ti 7에 sfl	6. 通信 7. システムの定義または変更 8. 問題処理	
90.サインオフ	^	9. メニューの表示 10. 情報振動オプション 11. SYSTERT X KOSSS のクラク	
<ul> <li>選択項目またはコマ、</li> </ul>	19 1 ボ	90. サインオフ	
===>	20 2	選択 項目またはコマンド	
	20 7	===>	
	22 2	F3= 終了 F4=ホ*ワ]ホ*n F9=テu]n*の複写 F12=取 F23= 初期 xp**の設定	マり消し F13≕ 情報援助
F3= \$% f	22 13	(C) COPYRIGHT IBM CORP. 1980, 2007.	
	22 26		
F9=Tu]n, 0)锁字			_ [¥ ■上

14. フィールドにtxtSelectionOrCmdという名前を付けて[保存]をクリックします。

*ブイル:         絵 終了 名前 パーパント見出し リー 行         パーパント見出し         ・         ・       Lock Screen         名前 行 列 sfl         90.サインオフ       ・         道沢項目またはコマンド       20 2         ===>       20 7         ・       20 7         ・       22 2				~
<ul> <li>Lock Screen</li> <li>名前</li> <li>行列</li> <li>90. サインオフ</li> <li>19</li> <li>1</li> <li>選択項目またはコマンド</li> <li>20</li> <li>22</li> <li>22</li> <li>22</li> <li>22</li> <li>22</li> </ul>	7711.: 16 韩了 名前	行/ リー	エット 見出 行	L
Lock Screen     名前 行 列 sfl     90. サインオフ     19 1     選択項目またはコマンド     20 2     ===>     txtSelectionOrCmd 20 7     22 2				< >
名前     行列     Sfl       90.サインオフ     19     1       道扱項目またはコマンド     20     2       ===>     20     7       txtSelectionOrCmd     20     7       22     2	- <b></b> +*:		Loc	k Screen
90.サインオフ 19 1 遠訳項目またはコマンド 20 2 ===> txtSelectionOrCmd 20 7 22 2	名前	fī	列 Sfl	
19 1       選択項目またはコマンド       20 2       ===>       txtSelectionOrCmd     20 7       22 2				
選択項目またはコマンド 20 2 ===> txtSelectionOrCmd 20 7 22 2	90.サインオフ			^
20 2 ===> txtSelectionOrCmd 20 7 22 2	90.サインオフ	19	1	^
txtSelectionOrCmd 20 7	90. サインオフ	19	1	^
txtSelectionOrCmd         20         7           22         2         2	90. サインオフ - 選択項目またはコマンド -	19	1	
22 2	90. サインオフ 	19	1 2	^
	90. サインオフ 選択項目またはコマンド ===> txtSelectionOrCmc	19 20 20	1 2 7	

15. i5/O5 Main Menu画面のコマンド行に以下のコマンドを入力して、 テーブル保守メニューを表示します。

lansa run psltab partition(dem)

- 16. 画面にPSLTABMainという名前を付けて識別します。
- 17. 矢印ボタンをクリックしてタブ・フォルダーを非表示にします。 以上で画面の命名が終了しました。

RAMP-TS002 ステップ2:画面の分類とナビゲーションの追跡 このステップでは、アプリケーションの画面を分類し、ナビゲーション を追跡します。画面を分類するときは、そのためにナビゲーション・ ルートを記録するスクリプトが作成されます。

このステップの前に、次のFAQをお読みください。画面の分類で操作を 間違えました。分類を変更するにはどうしたらいいですか?

移動するときは、番号を入力し、Enterキーを押してメニュー・オプションを選択します(キーストロークでメニュー・オプションをクリックすると、トラッキングはその操作に追従できません)。

- 1. テーブル保守メニューを終了します。
- 2. サインオフします。
- 3. メッセージ・エリアの[再スタート]ボタンをクリックして、ナビ ゲーション・トラッキングを再起動します。

<		
検証	スナッフ <sup>®</sup> ショット(862×424)	再スタート

ログイン画面がJunctionとしてすでに分類されていることがメッセー ジ・エリアに表示されます。



- 4. システムにログオンします。
- 5. メッセージ画面をSpecialフォームとして分類します。システムに よっては、この処理に少し時間がかかることがあります。RAMPが画 面タイプを割り当てるまでお待ちください。



- 6. メッセージを閉じて、i5/OS Main MenuをJunctionとして分類しま す。
- 7. 次のコマンドを入力してテーブル保守メニューに移動します。

lansa run psltab partition(dem)

- 8. これをDestination画面として分類します。
- 9. Destination画面から移動するナビゲーションを追跡するためにス テップをさかのぼります。

F12キーを押す	i5/OS Main Menuに戻ります。
コマンド行に90を入 力する	サインオフします。

以上で、アプリケーションの画面の分類とナビゲーションのトラッキン グを完了しました。 RAMP-TS002 ステップ3:[キャンセル]および[終了]ボタンの 削除

このステップでは、テーブル保守メニュー画面から[キャンセル]および [終了]ボタンを削除して、ユーザーがこの画面からアプリケーションの 他の部分に移動できないようにします。

- 1. [デフォルト セッション]タブを表示します。
- 2. Destination画面のリストで[PSLTABMain]を選択します。

	■ 検索	次を検索	🗖 スクリフᅆ内	
名前	ל לי	゙ループ 最終変更		ŧゃプション
י לאָאָ∍) ום <mark>ום</mark> Junction (2)	8			デドフォルト セッジョン
🖼 MainLogin		20100705-	150050-V2	Login Form
⊡ ⊡ Destination	(1)	20100705-	150114-72	
		20100705-1	150216-V2	
<ul> <li>□ □ □ Special (1)</li> <li>□ = UNKNOWN_FOR</li> <li>□ □ ↓ λ/ λ/</li></ul>	M=	20100705-	150043-V2	特別なUNKNOWNフォーム
FORMSCRIPT_1	2	20100705-1	150043-V2	=UNKNOWN_FORM= - u
FORMSCRIPT_1	3	20100705-1	150050-V2	MainLogin - このフォ
FORMSCRIPT_1 FORMSCRIPT_1	4 5	20100705-1 20100705-1	50114-V2  50215-V2	i5OSMainMenu - こ PSLTABMain - このフ
FORMSCRIPT_1 FORMSCRIPT_1	4 5	20100705-1 20100705-1	150114-V2 150215-V2	i5OSMainMenu - こ PSLTABMain - この7.

PSLTABMain画面の詳細が右側に表示されます(すべての詳細を表示する には、RAMPウィンドウの再配置が必要になることがあります)。

3. PSLTABMainの[ファンクション キー利用可能]リストで、[終了]および[取消し] キーの選択を解除します。

-								
Ft7 742	_							
バルーフ°								
RAMP	の既定値の	レイアウト サイン	Ζ'					
高さ	412	幅	710	上			左	
Top 7	スクの高さ		В	ottom 7.	スクの帽	ī		
-RAMP	マンシーン レイア	'ウト スタイル-		() T	定レイア	ウト		
				h h2 l	-			
-9-7'9	1			-9-7.9h	By			2
	保存				1	呆存		
	保存	- 44			ſ	保存		
- <b>7</b> 7)99	保存 ョン キー使用す	丁能			1	¥存	ার্করবস্থ	   
-77V99	保存 aン キー使用 Caption	可能	E C	FKEY :		保存 関連 (コマン	:するコマン ト* (タブ*)	
-ファンクシ キー F3	保存 マンキー使用 Caption 終了	「能」  利用可能	利用可	FKEY =		保存 -関連 ロン 「販	iするコマン ト* (タブ) 読	
-7ァンウシ キー F3 F4	保存 aンキー使用 Caption 終了 F4	可能		FKEY = 4 5		保存 関連 コマン 即 の 日	でするコマン ト* (タフ*) 読 見成	
-ファンウシ キー F3 F4 F5	保存 aンキー使用 Caption 終了 F4 F5	可能 利用可能		FKEY = 4 5 6		保存 月辺連 「見速 「別速 「別速 「日 」	「するコマン 下 (タフ <sup>*</sup> ) 売 手成 細細	
-ファンクジ キー F3 F4 F5 F6	保存 aンキー使用 Caption 終了 F4 F5 F6	可能 利用可能 日 日 日 日 日		FKEY s 4 5 6 7		¥存 ■ ■ ■ ■ ■ ■ ■ ■	するコマン ド (タブ) 売 師 舗細	
-ファンウジ キー F3 F4 F5 F6 F7	保存 aンキー使用 Caption 終了 F4 F5 F6 F7	可能 利用可能 日 日 日 日 日 日 日 日 日 日 日 日 日		FKEY = 4 5 6 7 8		<b>米存</b> 「関連 コマン 「販 一 朝 朝 朝 新	するコマン <u> 下 ( 97 * )</u> ( 97 * ) (	
-ファンウジ キー F3 F4 F5 F6 F7 F8	保存 マンキー使用 Caption 終了 F4 F5 F6 F7 F8	T能 利用可能 日 日 日 日 日		FKEY = 4 5 6 7 8 9			するコマン ト*(97*) 売売 結細 編組 メント	
-ファングジ キー F3 F4 F5 F6 F7 F8 F9	保存 aン キー使用 Caption 終了 F4 F5 F6 F7 F8 F9			FKEY 5 4 5 6 7 8 9 10		保存 見速 コマン 一覧通 一覧 一覧 一覧 一覧 一覧 一覧 一覧 一覧 一覧 一覧	するコマン <u> ト*(97*)</u> 売 転 細 細 新 規 メント 、 、 、 、 、 、 、 、 、 、 、 、 、	
-ファングジ キー F3 F4 F5 F6 F7 F8 F9 F10	保存 マンキー使用 Caption 終了 F4 F5 F6 F7 F8 F9 F10			FKEY s 4 5 6 7 8 9 10 11		<b>保存</b> 開連 コマン 「販」 一 開連 一 開連 一 開連 一 開連 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の 一 の の の の の の の の の の の の の	でするコマン ト <sup>*</sup> (タフ <sup>*</sup> ) 売 E成 細 紙 組 紙 規 パ ント	
-ファンウシ キー F3 F4 F5 F6 F7 F8 F9 F10 F11	保存 aン キー使用 Caption 終了 F4 F5 F6 F7 F8 F3 F10 F11			FKEY s 4 5 6 7 8 9 10 11 12		<b>保存</b> 「関連 「取) 「別) 「別) 「別) 「別) 「別) 「別) 「別) 「別	するコマン ド (タフ <sup>*)</sup> ド売成細 細 税 規 パ キュメント ト ・ 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、	
-ファンウジ キー F3 F4 F5 F6 F7 F8 F9 F10 F11 F12	保存 aン キー使用 Caption 終了 F4 F5 F6 F7 F8 F9 F10 F11 取消し			FKEY s 4 5 6 7 8 9 10 11 12 13			するコマン ド (タフ <sup>*)</sup> ド売成 細 細 税 規 パ コント	

RAMP-TS002 ステップ4:フレームワークへのアプリケーションのスナップイン

このステップでは、アプリケーションをフレームワーク内にスナップインします。また、PSLTABMain画面で非管理セッションを開始することも指定します。

非管理セッションは、主にユーザーをログオンして、特定の開始地点に 誘導するために使用します。この地点から、ユーザーはフレームワーク で管理されることなく5250アプリケーション内を移動できます。

これにより、フレームワーク・メニューの[コード・テーブル]をクリッ クすると、5250セッションでユーザーがログオンし、メニュー画面が ユーザーに表示されます。これで、数百の5250画面で構成されている場 合もあるコード・テーブル機能全体に、画面の命名、分類、追跡なしで アクセスできるようになりました。ユーザーがコマンド・ハンドラーか ら移動し、その後戻った場合、画面のタイプに関係なく現在の5250画面 が再表示されます。

 PSLTABMain画面の詳細で、[関連するコマンドハンドラー]リストの 使用可能なコマンドのリストをスクロールして、Code Tablesオブジェ クトの[詳細]コマンドを見つけます(コマンドを検索するときは、列を ソートできることに注意してください)。コマンドを選択します。

ŀプション ル−プ -RAMPの 高さ Top マス -RAMP;	D既定値のに 412 りの高さ ス別ーン レイア	/アウト サイ 幅 ウト スタイル	710 710	上 tom マフ	スクの幅	左		
	💿 tv:	242			C	固定レイアウト	ר 🔘	ローレイアウト
カーケシット			_h-	-htsp	Bv			
-772/994	保存 ンキー使用同	丁能———			6	保存  _−関連するコマント <sup>*</sup>	ハンドラー	
-ファンクショ) キー	保存 ンキー使用同 Caption	J能 利用可能		(EY s		森存   ・関連するコマンド   ・マリント**   ・マリント**    ・マリント*	ハント・ラー	2-₩°- オフ° 🔻
ファンクショ) キー Enter	保存 ンキー使用す Caption Enter	「能」 「利用可能」 ■▼	利用可 FK	(EY ء		保存 - 関連するコマンド <sup>*</sup> - フマント <sup>*</sup> (タフ <sup>*</sup> ) ^ ■ E印刷	ハントラー へのリンク オフドジ・エクト内 Products	ユーザ <sup>ゃ</sup> ー オフ <sup>・</sup> ・・・ ヘ CF83B034
ファンクション キー Enter F1	保存 ンキー使用。 Caption Enter F1	J能 利用可能 ▼	¥IJ用■ FK	(EY د 1 2		R存 関連するコマンド コマンド (タブ)^ 印刷 回販売	ハントラー へのリンク オフドジ・エクト内 Products Products	2-♥°- オフ° ▲ CF83B034 CF83B034
ファンクション キー Enter F1 F2	保存 ンキー使用可 Caption Enter F1 F2	J能 利用可能 □ □	¥IJ用□ FK	(EY ∉ 1 2 3		R存 関連するコマント <sup>*</sup> コマント <sup>*</sup> (タフ <sup>*</sup> )/ 印刷 販売 作成	ハント・ラー へのリンク オフドジ・ェクト内 Products Products Products Products	2-サ°- オフ° ▲ CF83B034 CF83B034 CF83B034
ファンクショ) キー Enter F1 F2 F3	保存 ンキー使用回 Caption Enter F1 F2 終了	T能 利用可能 □ □	利用可 FK	(EY s 1 2 3 4		保存 - 関連するコマント <sup>*</sup> コマント <sup>*</sup> (タワ <sup>*</sup> )へ 日印刷 回販売 一作成 マ詳細	ハント・ラー へのリンク オフドジ・ェクト内 Products Products Products Code Tables	2-サ°- オフ° ∧ CF83B034 CF83B034 CF83B034 99820455
ファンウショ キー Enter F1 F2 F3 F4	保存 ンキー使用車 Caption Enter F1 F2 終了 F4	T能 ▼ ■ ■ ■		(EY s 1 2 3 4 5		採存           コマント*(タワ*)へ           コマント*(タワ*)へ           ロ印刷           販売           作成           ご詳細           詳細	ハント・ラー へのリンク オフドジ・エクト内 Products Products Products Code Tables Employees	2-∜°- オフ° ∧ CF83B034 CF83B034 CF83B034 99820A55 99017EA3
ファンクショ キー Enter F1 F2 F3 F4 F5	保存 ンキー使用 Caption Enter F1 F2 終了 F4 F5	J能 ▼ ■ ■ ■ ■ ■		(EY ≰ 1 2 3 4 5 6		採存       コマント*(タワ*)へ       コマント*(タワ*)へ       ロロ刷       町売       作成       ご詳細       詳細       新規	ハント・ラー のリンク オフドジ・ェクト内 Products Products Products Code Tables Employees Employees	2-サ°- オフ° ▲ CF83B034 CF83B034 CF83B034 99820A55 99017EA3 99017EA3 90017EA3
ファンクショ キー Enter F1 F2 F3 F4 F5 F6	保存 Caption Enter F1 F2 終了 F4 F5 F6			(EY ≰ 1 2 3 4 5 6 7		<ul> <li>         · 関連するコマンド         · コマンド         · (タブ)/         · 印刷         · 印刷         · 印成         · 「作成         · 「詳細         · 詳細         · 新規         · ドキュメント         · ゲント         · ゲント         · ゲント         · ケント         · ・         · ケント         · ・         · ・         · ・</li></ul>	ハント・ラー のリンク オフドジ・ェクト内 Products Products Products Code Tables Employees Employees Employees Employees	2-サ*- オフ* ▲ CF83B034 CF83B034 CF83B034 93820A55 93017EA3 93017EA3 93017EA3
ファンクショ キー Enter F1 F2 F3 F4 F5 F6 F7 F7	保存 Caption Enter F1 F2 終了 F4 F5 F6 F7 F7			(EY ≰ 1 2 3 4 5 6 7 8		<ul> <li>「関連するコマント"</li> <li>「知道」であっていた。</li> <li>「印刷」</li> <li>「販売</li> <li>「作成</li> <li>「詳細</li> <li>「詳細</li> <li>「新規</li> <li>ト"キュメント</li> <li>「ペ"ント</li> </ul>	ハンドラー へのリンク オフ <sup>・</sup> ジ <sup>・</sup> ェクト内 Products Products Products Code Tables Employees Employees Employees Employees	2~♥°~ 才7° ∧ CF83B034 CF83B034 CF83B034 99820A55 99017EA3 99017EA3 99017EA3
ファンクショ キー Enter F1 F2 F3 F4 F5 F6 F7 F8 F7	保存 2 キー使用 Caption Enter F1 F2 終了 F4 F5 F6 F7 F8 F7 F8	J能 利用可能 □ □ □		(EY ≰ 1 2 3 4 5 6 7 8 9		R存 - 関連するコマント <sup>*</sup> コマント <sup>*</sup> (タワ <sup>*</sup> )/ 印刷 販売 - 作成 マ 詳細 - 詳細 - 新規 - ト <sup>*</sup> キュメント - イ <sup>*</sup> ント	ハントラー へのリンク オフ <sup>・</sup> ジ <sup>・</sup> ェクト内 Products Products Products Code Tables Employees Employees Employees Employees Employees	2-♥°- オ7° ∧ CF83B034 CF83B034 CF83B034 99820A55 99017EA3 99017EA3 99017EA3
ファンクショ キー Enter F1 F2 F3 F4 F5 F6 F7 F8 F9 F10	保存 2 キー使用 Caption Enter F1 F2 終了 F4 F5 F6 F7 F8 F9 E10			(EY ≰ 1 2 3 4 5 6 7 8 9 10		R存 関連するコマント <sup>*</sup> (タワ <sup>*</sup> )へ 印刷 販売 作成 <b>マ詳細</b> 新規 ト <sup>*</sup> キュメント イ <sup>*</sup> ント	ハントラー 、のリンク オフドジ・ェクト内 Products Products Products Code Tables Employees Employees Employees Employees Employees Employees	2-サ°- オフ° ∧ CF83B034 CF83B034 CF83B034 98820A55 99017EA3 99017EA3 99017EA3

2. PSLTABMain画面の詳細で、セッションIDを*SESSION\_A*に設定しま す。

セッションID	SESSION_A

3. [保存]ボタンをクリックして、RAMP定義を保存し、[保存]を選択 します。

テフォルト セッション					
保存 🔹	新5250	)ፖጋ ግን ተንቋ	ワ セッション	削除	
保存		•	検索	次を持	<b>検索 🗌</b> ズ
全て保存		· · · ·	最終変更	‡¢]	7°9∍ນ

RAMPツールで作業するときは、一般に作業の[保存]を行います。エンド・ユーザー・モードでアプリケーションをテストする準備が整っている場合のみ、RAMP定義の[全て保存]を行います。

 フレームワークを表示し、人事アプリケーションを選択し、次に Code Tablesビジネス・オブジェクトを選択します。テーブル保守メ ニューが表示されます。



5. メニューのいずれかのオプションを選択し、コード・テーブルにア クセスします。

🐇 Code Tables						
ファイル(E) 編集(E) 表示(V) ヘルプ(	<u>ዘ</u> )	M) (71/−47	-り)(管理)			
③作成 台 画印刷 奇	PISIA	(A)		🖂 🍇 🗟 🗿 🛛 🚧	动榆壶	
			* ~ ~   *		//iz.m •••	
Code Tables						
X						x
🕀 🍇 DEM HR	📣 Code 1	Tables				1**
🕀 🌠 ERP Prototype 1	4.		=¥ -	コード・テーブル		
🕀 🐕 HR 🕫 €77°94~9a2			BT -	4 1 7 270	バーを表示	
					トドキュメンテーション	
		部門	課		1 1-777 747	
🖃 🚰 人事	SEL	コード	コード	課名称	トレース オン	
🐗 Code Tables		ADM	01	INTERNAL ADMIN	スクリーンの調査	
Employees		ADM	02	PURCHASING SECT		
田 (1) 717 717 79-97		ADM	03	ACCOUNTING SECT		
		ADM	04	SALES & MARKETI		
		ADM	05	MAINTENANCE		
		ADM	06	PERSONNEL SECTI		
		ADM	09	VEHICLE MAINTEN		
		AUD	01	ADMINISTRATION		
		AUD	02	PURCHASING		
		AUD	03	ACCOUNTING		
		FLT	01	ADMINISTRATION		
		FLT	02	PURCHASING		
		FLI	05	ETNANCIAL CONTR		
		GAC	01			
		GAC	02	PURCHASTING		
	データる	野換に置換	文字が使用され	た可能性があります。		
E 🖽 📕	×	ッセージ 3	創備 ASID02_TS	S JPN V20PGMLIB 1	0/07/05 15:43 🥥	

システムの設定方法によっては、画面の外観が少し異なる場合があります。

以上で、コード・テーブルのメンテナンス・アプリケーション・セグメ ントが刷新され、フレームワークで完全に機能するようになりました。 RAMP-TSの構成によっては、刷新後の画面が少し異なることがありま す。

#### まとめ

重要なポイント

- アプリケーションの5250画面ごとにRAMPを使用する必要はありません。多くの場合、アプリケーション・セグメント全体が使用されることはあまりないため、これらを一般にフレームワーク内にきわめて短時間でスナップインすることができます。
- アプリケーションの刷新している部分にかける時間とリソースの量 を慎重に検討する必要があります。
- 名前は任意の順序でいつでも付けることができます。1つのセッションでナビゲーションに使用するすべてのフォームとフィールドに名前を付ける必要はありません。ただし、画面やフィールドを分類してアプリケーションのナビゲーションを追跡する前に、すべての画面およびフィールドに名前を付ける必要があります。

覚えるべき事柄

- フレームワークでアプリケーションとビジネス・オブジェクトを作 成する方法、およびコマンド・ハンドラーをビジネス・オブジェクト に関連付ける方法
- 画面に名前を付ける方法
- 画面を分類し、ナビゲーションを追跡する方法
- 画面のボタンを有効または無効にする方法
- 画面をフレームワーク内にスナップインする方法

## RAMP-TS003:社員のデータ・フィルターの作成

このチュートリアルでは、Employeesビジネス・オブジェクトの機能的な 名前フィルターを作成します。コード ウィザードを使用してフィルター を作成します。

目的

- 刷新されたアプリケーションにフィルターが根本的に価値を付加す る方法を理解する
- アプリケーションで情報にアクセスする種々の方法を提供するフィルターをすばやく作成する方法について理解する

<mark>◇ Employees</mark> 77ℓル(E) 編集(E) 表示(V) ^     ③ 作成   〇   四則   □     Employees	w7*(1) ウィント*ウ(w)(フレームワーウ)(管理)	このチュートリアルでは、社員を名前で検索する フィルターを作成します。 フィルターは、刷新されているアプリケーションに 価値を追加する最も強力な方法の1つです。	
× ■ DEM HR ■ ERP Prototype 1 HR デモアブツウ〜ション Human Relation ■ お気に入り ■ 大事 Code Tables Employees ■ ブログ・ラミング、デクニック 管理	名前 By Date of Birth By Salary 社員姓 S マリストの消去	を かいまた この	
🖻 🗰 📴	メッセージ 準備 ASI002_TS	JPN V20PGMLIB 10/07/06 9:14	

この目的を達成するには、以下のステップを完了してください。

- RAMP-TS003 ステップ1. 実際の名前フィルターを作成する
- RAMP-TS003 ステップ2. 名前フィルターをスナップインとして組み 込む

- RAMP-TS003 ステップ3. フィルター・コード
- まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新

RAMP-TS003 ステップ1. 実際の名前フィルターを作成する このステップでは、従業員の姓によりPSLMSTファイルを検索する実際 のフィルターを作成します。コードウィザードの使い方についても学び ます。

1. [名前]フィルターの[コード ウィザード]ボタンをクリックします。



[コード ウィザード]ウィンドウが表示されます。ここでは、フィル ター、インスタンス・リスト、およびコマンド・ハンドラー内にプラ グインできる別のタイプのコンポーネントを作成できます。フレーム ワークを初めて使用するときは、コード ウィザードを使用することを 強くお勧めします。

多くの場合、最初は、実行可能なコンポーネントを生成するフィル ターを使用します(CRUDフィルター(Create/Read/Update/Delete)、ファ イルまたはビューを検索するフィルターなど)。作業の進行と共に、ス ケルトン・フィルターのみを使用したり、単に作成したいフィルター に類似するフィルターをコピーします。

- 英語以外のシステムを使用している場合は、[フレームワーク]、ツ リー表示の左上にあるフレームワークの順にクリックします。下側に コード生成オプションの設定オプションが現れるので、オプションを 選択し、プリファレンスを設定します。
- 3. 左上のリストで、[人事]アプリケーション、[Employees]、および[名 前]フィルターが選択されていることを確認します。
- 4. その下にある[標準MS Windows]をプラットフォームとして選択しま す。
- 5. 生成するコードのタイプとして、[ファイルおよびビューを検索するフィルター を作成]を選択します。



- 6. [次へ]ボタンをクリックします。
- 7. 次のページで、[このビジネスオブジェクトに一番近い物理ファイル]としてPSLMSTを指定します。

□ コート* ウィザ*ート*						
コード生成するオフジェ外を選択して下さい			7711	しおよびビューを検索するフィルタを作り	戎	
<ul> <li>□ (*) (* 3/ 3/ *) * 2h-&gt;Employees</li> <li>□ コマント* ハント* &gt;&gt;-&gt;新規</li> <li>□ コマント* ハント* &gt;&gt;新規</li> <li>□ コマント* ハント* &gt;&gt;*新規</li> <li>□ コマント* ハント* &gt;&gt;ト* キュメント</li> <li>▲ コマント* ハント* &gt;&gt;+</li> <li>キュメント</li> <li>▲ コマント* ハント* &gt;&gt;</li> <li>イント* &gt;&gt;</li> </ul>	この を ファイ どう	ひとジネスオフジェイ 皆定して下さい。 とジネスオフジェク ル: 「ュアル識別子(Visu	小の定義を自 Nicー番近い JalID1 と Visu	<ul> <li>         ・         ・         ・</li></ul>	わジ:	よりNこ近い物理ファイルの名前
] フィルター->By Date of Birth		フィールド名	Туре	記述	^	選択分の削除
ノ///ター->By Salary □ コント・ハット・ラー・>カレートワーカニーフレッア	1	EMPNO	ALPHA	社員Na		全て買加金
再表示	2	SURNAME	ALPHA	社員姓		± CHIMA
∼生成したいプラットフォームを選択	3	GIVENAME	ALPHA	社具名	_	
● MS Windows (Visual LANSA の使用)	4				~	
<ul> <li>Web (LANSA for the Web の使用)</li> <li>Web (WAM コンボーネントの使用)</li> <li>Web (AJAX スタイル コンボーネントの使用)</li> </ul>	このしま	が物理ファイルのフィー す	ルトを追加	キーの追加	全-	
生成するコードの種類を選択して下さい	20	かうム用識別子 (4	AKey1,2,3,4,5 -	と NKey1,2,3,4,5 の値を生成するた	め)_	
ファイルの全論理ビューで検索するフィルク		フィール・名	Туре	記述	^	選択分の削除
CRUD フイルタ ファイルおよびビューを検索するフィルタを作成	1	EMPNO	ALPHA	社員Na		全て削除
フィルターのソースのアクトラインを作成	2					
tavListManager.AddtoList の実行(ソースの一部分)	ů	-			~	
	このしま	物理ファイルのフィー す	ルドを追加	キーの追加	全"	CiBbo
	通	面の柳 (AColumn	くと NColum	nn<> の値を生成するため)		
		フィールド名	Туре	記述		選択分の削除 🗸
	<					
				<< 戻る	>	ケヘ >> キャンセル

コード ウィザードは、必要なビジュアル識別子とプログラム的識別子を 検出します。

- ビジュアル識別子は、ユーザーがビジネス・オブジェクトの一意の インスタンスを識別するために使用するフィールドです。
- プログラム的識別子は、プログラムがビジネス・オブジェクトの一 意のインスタンスを識別するために使用するフィールドです。通常、 これらの識別子はインスタンス・リストのデータを構成するファイル のプライマリ・キーとなります。
- 追加の列は、プロトタイプ作成フェーズの間に追加された、インス タンス・リスト内の追加の列を表します。

- 8. [次へ]ボタンをクリックします。
- 9. 次のページで、[フィルタリングと検索に使用するビュー]として PSLMST2を指定します。これは、SURNAMEおよびGIVENAME フィールドによるキー付きのPSLMSTファイルの論理ビューです。

作成するフィルターごとに適切な論理ファイルが必要になることに 注意してください。すべてのフィルターを実装する前に、データ・モ デルを検証し、すべての論理ファイルが存在することを確認します。 これにより、プロトタイプの実行のプロセスを迅速に行うことができ ます。

10. 検索操作に使用するビューのキーとして、[SURNAME]フィールド を選択します。



- 11. [次へ]ボタンをクリックします。
- 12. [インスタンスリストを更新するためのシグナルを受け取るRoutine] オプションを選択します。
- 13. [生成]ボタンをクリックします。

次のページ、[生成されたソース コード]にフィルターのソース・コードが表示 されます。ここで、このコードを含むコンポーネントを作成する必要が あります。

- 14. 実際のフィルターの名前として、iiiRMP01を指定し、説明として名 前フィルターを指定します。(iiiは自分のイニシャルです。Visual LANSAの非ライセンス版または試行版を使っている場合は、iiiの部分 をDEMとする必要があります)。
- 15. [生成]ボタンをクリックし、コンポーネントを作成します。



しばらくして、フィルター・コンポーネントがVisual LANSAエディター に表示されます。

16. コンポーネントをコンパイルします。

RAMP-TS003 ステップ2. 名前フィルターをスナップインとして組み込む

新しく作成したフィルターをコンパイルし、テストできる状態になった ので、これをフレームワークにスナップインする必要があります。

- 1. フレームワークで、コード ウィザードを閉じます。
- 2. Employeesビジネス・オブジェクトをダブルクリックし、そのプロパ ティを表示します。
- 3. [フィルター登録の設定]タブを表示します。
- 4. iiiRMP01をWindowsフィルター・ハンドラー・コンポーネントとして指定します。

🤞 ヒシキス オブシェクトのフロハラ	r - Employees	
識別子   アイコン   ビジュアル スタイル	フィルター フィルター設定 使用可能コマンド  コマンドの表示 ユーザー プロパティ サブタイプ  インスタンス リストの設定	1
二名前	識別子 アイコン フィルター登録の設定	
By Date of Birth By Salary	常にアクティフ 省略値 💌	
	-741/3- AVF5	
	● コンホーネント EOMRMP01	
	〇 サンプル - RAD-PAD _RADPAD_9C33AB6CFEE8485EAC43F8D936DA6FD6.HTM	
	-Web 7'50#	
	○ WEBEVENT ファンクション VFU0301 2~7泊セス: VF_PR003	
	本 物の位置 右 マー・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	
	レイアウトダイブのフォーム 見出れ。神とフ、ラウス、リストル。神(省略値) 💌	
	VFU0501	
	○ AJAX ページ	
	<ul> <li>・サンプル - RAD-PADRADPAD_9C33AB6CFEE8485EAC43F8D936DA6FD6.HTM</li> </ul>	
	りうス名 🔍 🔍	
	7tzu71	
新規作成 削除		
		:3

 Employeesビジネス・オブジェクトのプロパティを閉じて、名前フィ ルターを表示します。これにより、実際のフィルターが表示されま す。 6. [姓]フィールドに文字を入力し、[検索]ボタンをクリックして実際の フィルターがフレームワークにスナップインされ、使用可能であるこ とを確認します。

ファイル(E) 編集(E) 表示(V) ヘルプ(H) ウィンドウ(W) (フレームワーウ) (管理)         ③ 作成       二       白 印刷       二       M       二       6       M       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1       1 <t< th=""></t<>
③作成 🗇 酉印刷 范 🗋 🖄 Δ 🔟 🗖 📓 ڭ 💮 🚳 🚳 🖗 🖗 🖓 🚺 🕺 🕺 🥠 粉皮素
Employees
<ul> <li>○ A</li> <li>○ A</li></ul>

#### RAMP-TS003 ステップ3. フィルター・コード

ほとんどのフィルターはコード ウィザードを使用して簡単に作成するこ とができますが、フィルターのコードを記述する方法を理解している必 要があります。

- 1. Visual LANSAエディターに切り替えて、iiiRMP01の再使用可能な パーツを開きます。
- 2. 生成されたソース・コードを[ソース]タブで検証し、フィルターが コード化され、データがインスタンス・リストに追加される方法を確 認します。

更新が発生することがフレームワークに通知されます。

Invoke #avListManager.BeginListUpdate

次に、リストの既存のアイテムがクリアされます。

Invoke #avListManager.ClearList

次に、データが選択されます。Visual LANSA Fundamentalsチュートリア ルで学習したテクニックの1つを使用して、これを実行できます。例:

Select Fields(#XG\_Ident) From\_File(PSLMST2) With\_key(#XG\_Keys) Generic(\*yes) Nbr\_Keys(\*Compute)

次に、ビジュアル識別子が設定されます。

Change #UF\_VisID1 #EMPNO Change #UF VisID2 #SURNAME

次に、データがリストに追加されます。

Invoke #avListManager.AddtoList Visualid1(#UF\_VisID1) Visualid2(#UF\_VisID2) AKey1(#EMPNO)

VisualId1はインスタンスリストの列1に、VisualId2は列2に表示されま す。Akey1は従業員を一意的に識別するキーです(この場合、フィールド は英数字なので、Nkey1ではなくAkey1)。 最後に、インスタンス・リストの更新が完了したことがフレームワーク に通知されます。 Invoke #avListManager.EndListUpdate)

3. 次に、エディターの[詳細]タブをクリックし、コンポーネントのプロ パティを表示します。

PrizontalScroll	False
🖻 HorizontalScrollInc	1
PorizontalScrollPos	0
🛷 アウトライン	
<b>回</b> 詳細	
🖯 ሀቱንንጉሀ	

すべてのプロパティが表示されていることを確認する必要があります。

- 4. [オプション]メニューの[設定]オプションを選択します。
- 5. [詳細]をクリックし、[拡張機能の表示]オプションが選択されている ことを確認します。

📑 LANSA Ø	設定	
<ul> <li>→般</li> <li>ソースエディタ</li> <li>デ<sup>*</sup>ザイン</li> <li>デ<sup>*</sup>ザイン</li> <li>ズSL</li> <li>デ<sup>*</sup>バッグ</li> </ul>	<ul> <li>一般</li> <li>○ 詳細情報の表示</li> <li>○ ビントの表示</li> <li>○ ビントの表示</li> <li>○ オー順の表示</li> <li>○ パロ部ファイルの表示</li> <li>○ パロ部ファイルの表示</li> <li>○ マスター情報を保存</li> <li>○ マスター情報を保存</li> <li>○ アンスタント</li> <li>○ アシスタント</li> <li>○ 詳細</li> <li>○ 拡張機能の表示</li> <li>○ アウトライン</li> </ul>	OK( <u>O</u> ) キャンセル( <u>C</u> ) 適用 リセット 全てリセット エディターのリセット

6. コンポーネントの[Ancestor]プロパティが#VF\_AC007であることがわ

かります。すべてのフィルターは、一連の定義済みの動作を提供する このベース・クラスから継承します。

📜 LANSA ፲ディター -	EOMRMP01(再使用	月可能なパーツ
ファイル(E) 編集(E) 表示	〒── 打ジョン@) 検査	{( <u>R</u> ) Web デン
🔁 作成 👻 🤌	80 % 6	26
詳細		8
EOMRMP01		-
<mark>ንግ</mark> ባየንተ ፈላንት አንታት		
P Ancestor	#VF_AC007	^
P BusyUpdates	Wait	
P BusyUpdatesOfParent	False	
ComponentClassName	EOMRMP01	
ComponentDescription	By Name Filter	

7. エディターの[アウトライン]タブを開き、Ancestorコンポーネント VF\_AC007からどのコンポーネントを継承しているか確認します。



8. [avLISTMANAGER]コンポーネントを右クリックし、[機能]オプションを選択します。



9. コンポーネントのメソッドを展開して検証します。



10. iiiRMP01コンポーネントを閉じます。

これらのコンポーネントの構造化の方法については、フレームワークガイドの「フィルターおよびコマンド・ハンドラーの構造」を参照してください。

#### まとめ

重要なポイント

 フィルターは、刷新されたアプリケーションに価値を付加する主な 方法の1つです。

ヒントとテクニック

 デモンストレーション・アプリケーションに使われているフィル ターのソース・コードが、DF\_\*という名前のコンポーネントのリポ ジトリにあります。

これまでの学習内容

- 独自のフィルターを作成するために必要な作業
- フィルターをスナップインとしてフレームワークに組み込む方法
- コード ウィザードの使い方

# RAMP-TS004:従業員画面の命名と分類 このチュートリアルでは、Employeesビジネス・オブジェクトで使用され る人事システム画面に名前を付けて分類します。 目的

• 画面の命名と分類について詳しく学ぶ



この目的を達成するには、以下のステップを完了してください。

- RAMP-TS004 Step 1. 画面の命名
- RAMP-TS004 ステップ2. 画面の分類
- RAMP-TS004 ステップ3. ナビゲーションの追跡
- RAMP-TS004 ステップ4. Destination画面のスナップショットの作成

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成

## RAMP-TS004 ステップ1. 画面の命名

このステップでは、アプリケーションの画面と必要なフィールドに名前 を付けます。

- 1. フレームワークを起動します。
- 2. RAMPを起動します。
- 3. サーバーに接続します。
- 4. 以下のコマンドを入力し、PSLSYSアプリケーションを起動します。

lansa run pslsys partition(dem)

5. メイン・メニューにPSLSYSMain、オプション・フィールドに txtOptionという名前を付けます。

名前: PSLS	YSMair	1	(ID = 1	4)		^
別名:	2	_				
記述:					< >	
サフ・ファ 同社会	イル: 林了	名前	fi	/T7F	見出し	
5	19	SFL004SF	y- 1		fi 	~
7~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	**:			Г	Lock Scre	en
	名前		ក	列	Sfl	
	社員管	理メイン・メニュー	1	20		
П			3	2		
	番号を	選択するか、カー	ソルを達	択項	目へ移動し	
	txtOpt	on	5	2	٧	
	1.		5	5	Y	
<u> </u>	<u> </u>		5	9	۷	
	计昌结	報 の ÉDRI				

フィールドの前にあるチェック・ボックスは選択しないでください。こ のチェック・ボックスは、他の同様の画面間でフィールドを区別するた めに使用します。

以降のステップでは、変更を保存するかどうかを確認するメッセージが 表示されるたびに[OK]をクリックします。

Web ページからのメッセージ	×
定義の変更を保存しま	:すか?
OK キャンセル	

- 6. メニューのオプション2を選択して、社員情報の登録画面を表示しま す。
- 7. これに、NewEmployeeという名前を付けます。
- 8. 社員No.フィールドにtxtEmpnoという名前を付けて、[保存]をクリックします。

Keymaps AutoGUI Screens	K Session Display Help	🔎 - Auto - 💌 🔍 🧔
<u> </u>	社員情報の登録	
NewEmployee	社員院	
<u>Ma-</u>	社員姓	
記述:	住所1 住所2	
<u></u>	住所3	
ማን <sup>ተ</sup> ንታላጺ።	1) 2010年1月11日1日11日11日11日11日11日11日11日11日11日11日11日	
開始 林了 名前 行/エット見出し リー 行	部門電話番号	
A 1	課コード	
×	入社日付 (DDMMYY)	
7-AF: Lock Screen	XE JET ⊟ 1\] (DDMMYYY)	
名前 行列 Sfl		
社員情報の登録		
3 2		
社員Na	F1= ヘルフ F3= 終了 F4=PROMPT F12= 収泊し F14=MSG	
txtEmpno 3 43		
4 2		
社員姓		▲ ■ 上 書

- 9. F12キーを押して、社員管理メイン・メニューに戻ります。
- 10. メニューのオプション3を選択して、社員と資格の詳細を表示します。
- 11. 社員検索画面にFindEmployeeという名前を付けます。
- 12. 社員No.フィールドを見つけて、txtEmpnoという名前を付けます。



- 13. [保存]をクリックします。
- 14. A1234などの社員番号を入力し、Enterキーを押して資格情報の検索 / 保守 画面を表示します。
- 15. 画面にDisplayEmployeeという名前を付けます。
- 16. [保存]ボタンをクリックします。
- 17. F21キーを押して画面を更新モードで表示し、フィールドに名前を

# 付けることができるようにします。

18. 画面の[姓]フィールドにtxtSurnameという名前を付けます。

Keymaps AutoGUI Screens	🛠 Session Display Help 🔎 - Auto -	- 🔍 🍙
★単: Diaplay/Employee (10 - 47)	資格情報の検索/保守	
Ma:	社員No	
	社員名	
202: 	住所2	125
<u>ک</u> ۲.۲۰۰۲ :	自宅電話番号	
開始 執了 名前     行/エント見出し リー 行	課コード	
18 21 R@F00003 1	追慰日付 (DDMMYY)	
7~11: Lock Screen	取得日 没格 (MDMMAYY) コード 次格記述 コマント 次	12
名前 行 列 Sfl	98/03/25 ADMIN1 Administratn Part 1 Met requirement     98/05/03 ADMIN2 Administratn Part 2	
4 2	98/05/05 ECD Economics Degree 94/06/05 ENG English Degree With distinction	+ 0
社員姓 ····································	F1= ヘルプ F3= 終了 F4=FROMPT F12= 取消し F14=MSG	
txtSumame 4 43		
5 2		
社員名 5 37		🔀 🖬 L

## これで画面の命名が終了しました。



- 19. RAMP定義の部分的な保存を実行します。
- 20. F3を押してi5/OS Main Menuに戻ります。
#### RAMP-TS004 ステップ2. 画面の分類

このステップでは、画面をJunction画面、Special画面、またはDestination 画面として分類します。

 [北 ケ -ション スクリフ トの自動更新]チェック・ボックスの選択を解除し (次のステップでナビゲーションを追跡します。ここでは画面の分類の みを行います)、自動ナビゲーションに関するメッセージをしばらくの 間無視します。

11:ウーション スクリフトの自動更新

- 2. RAMP-TS 5250セッションでシステムにログオンします。
- 3. メッセージ画面が表示された場合は、画面を閉じます。
- 4. i5/OS Main Menuで、以下のコマンドを入力します。

lansa run pslsys partition(dem)

- 5. オプション2を選択して新しい社員を登録します。
- 6. 社員情報の登録画面が表示されたら、F12キーを押して、社員管理メ イン・メニューに戻ります。
- 7. オプション3を選択し、社員を参照します。
- 8. Find Employee画面で、従業員番号を入力します。例:A1234

資格情報の検索 / 保守画面が表示されます。

以上で、刷新しようとしている、分類の準備が整った人事システム・ア プリケーションのすべてのリストを取得しました(画面の分類は、RAMP-TS002 ステップ2:画面の分類とナビゲーションの追跡で説明しているよ うにナビゲーションを追跡している間に、またはここで説明している2つ の異なるステップに従って実行できます)。

通常、画面の分類は異なるタスクとする方が適切です。準備が整った ら、保有しているスプレッドシートや何らかのドキュメントを使用し て、JunctionからDestinationへの移動を入念に追跡することができます。 この方法で、その後のやり直しの量が最小になるはずです。

- 以上で、画面の分類を開始できます。
- 9. [Tracking Info]エリアのMainLogin画面をクリックします。これは Junctionとしてあらかじめ定義されています(自動的に分類されていま す)。
- 10. [Tracking Info]エリアを使用し、残りの画面を以下のように分類します。

PSLSYSMain	Junction
NewEmployee	Destination
FindEmployee	Junction
DisplayEmployee	Destination

以上で画面の分類が完了しました。



#### RAMP-TS004 ステップ3. ナビゲーションの追跡

このステップでは、アプリケーションのナビゲーションを追跡します。

- 1. キャンセルしてi5/OS Main Menuに戻り、サインオフします。
- [再スタート]ボタンをクリックして、トラッキング情報をクリアします。



3. [ナビゲーション スクリプトの自動更新]チェック・ボックスを選択します。

🔽 ナビゲーション スクリブトの自動更新

- 4. RAMP-TS 5250セッションでシステムにログオンします。
- 5. メッセージ画面が表示された場合は、Enterキーを押して閉じます。
- 6. i5/OS Main Menuで、以下のコマンドを入力します。

lansa run pslsys partition(dem)

- 7. オプション2を選択して新しい社員を登録します。
- 8. 社員情報の登録画面が表示されたら、F12キーを押して、社員管理メ イン・メニューに戻ります。
- 9. オプション3を選択し、社員を参照します。
- 10. Find Employee画面で、社員番号を入力します。例:A1234
- 11. F12キーを押して、資格情報の検索/保守画面から社員管理メイン・メニュー戻ります。
- 12. F12キーを押して、i5/OS Main Menuに戻ります。

13. サインオフします。

以上で、刷新しようとしている人事システム・アプリケーション・セグ メントの基本的なナビゲーションを追跡しました。

14. [ナビゲーション スクリプトの自動更新]チェック・ボックスの選択を解除し ます。

# RAMP-TS004 ステップ4. Destination画面のスナップショットの作成

このステップでは、2つのDestination画面、NewEmployeeと DisplayEmployeeのスナップショットを作成します。これらのスナップ ショットは、次のチュートリアルでプロトタイプを拡張するために使用 されます。

- 1. ログオンして、NewEmployee画面に移動します。
- 2. RAMP-TS 5250エミュレータの左側のタブ・シート([Screens]、 [AutoGUI]、[Keymaps])が非表示になっていることを確認します。
- 3. [スナップショット]ボタンをクリックします。

検証	スナップ <sup>®</sup> ショット(915×514)	再スタート	🗖 チビゲーション スクリプトの自動更新

 スナップショットが区画実行ディレクトリに保管されたことを示す メッセージが表示されます。

スナッフ*ショットの 保存	×
ファイルヘビットマッフ <sup>®</sup> イメージとしてスナッフ <sup>®</sup> ショットを保存 C:¥PROGRA <sup>®</sup> 1¥LANSAV12¥X_WIN95¥X_LANSA¥x_dem¥execute¥SNAPSHOT_5250_NewEmployee.bmp(862x4	24)
OK	

- 5. DisplayEmployee画面のスナップショットを作成します。
- 6. RAMP情報の部分的な保存を実行します。

まとめ

重要なポイント

- ナビゲーションを追跡する前に、アプリケーションのすべての画面 およびナビゲーションに使用されるすべてのフィールドに名前を付け る必要があります。
- ナビゲーションの追跡と同時に、または追跡を完了した後に画面を 分類できます。
- プロトタイプで使用する5250画面のスナップショットを作成できます。

覚えるべき事柄

- 画面およびナビゲーション・フィールドに名前を付ける方法
- ナビゲーションを追跡し、画面を分類する方法

### **RAMP-TS005**:デザインの検証

このチュートリアルでは、Destination画面のスナップショットを使用し てプロトタイプを拡張します。

目的

• プロトタイプにスナップショットを追加する方法を学習する

👹 Employees	
ファイル(E) 編集(E) 表示(V) ヘルプ(H) ウィントウ(W) (フレームワーク)(管理)	
③作成 🗇 🚍 印刷 🏦 📄 🛇 🔺 🔟 🗖 📮 🗐 🔿 🍥 🐼 象 🛞 🖉 🥢 外分検索	
Employees	
X Ry Date of Birth Ry Salary	×
Cのステップでは、5250 Destination     使索 Employee 記述	^
<ul> <li>画面のステッシンヨットをノロトダイノに</li> <li>A0193</li> <li>SMITHSON FR</li> <li>A1000</li> <li>CMUTUR 2010</li> </ul>	ED
A1004 SMITHSON PA	UL 🙀
■ ( 7*0)* 752/2* 70=>2	×
■ ● 管理	U
※ 没格情報の検索/保守	<u> </u>
行 员 %	
社員会 STEPHEN	
住所 1 6 Melissa Place	
住所 2: West Pennant Hills	
住所3 iNSW Australia 2125	
自宅電話曲号	
都門コード	~
	>
詳細を表示 コード・ウィザード RAD	PAD
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	

この目的を達成するには、以下のステップを完了してください。

- RAMP-TS005 Step 1. モック・アップ・コマンド・ハンドラーへのス ナップショットの配置
- RAMP-TS005 ステップ2. プロトタイプの検証
- まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して

いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類

RAMP-TS005 ステップ1. モック・アップ・コマンド・ハンド ラーへのスナップショットの配置

最初のプロトタイプでは、エンド・ユーザーが情報にアクセスする方法 に焦点を当てて、刷新されたアプリケーションの実行可能なモデルを作 成しました。次に、5250アプリケーションの画面に名前を付けて分類し たので、Destination画面のスナップショットをプロトタイプに添付して 視覚的により完全なものにします。

このステップでは、5250 Destination画面のスナップショットをコマン ド・ハンドラーに配置します。

1. フレームワークで、インスタンス・リストの社員を選択し、社員に 関連付けられているコマンド・ハンドラーを表示します。

de Employees	
ファイル(E) 編集(E) 表示(⊻) ヘルプ(	⊎ ウィンドウ(₩) (フレームワーク) (管理)
③作成 🗇 🖨 印刷 🚡	📄 🔊 🔺 🔟 🗖 📮 🖾 🗇 🧔 🔯 🖉 🖉 🦄 🖓 👘 Унужфата
Employees	
× ● DEM HR ● ERP Prototype 1 ● HR デモアブリク〜ション ● Human Relation ● Application ● Application ● Application ● Code Tables Employees ● Of D? テラング テクニック	A前 By Date of Birth By Salary 社員姓 S ダリストの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去 メートの消去
■ 🍕 管理	>
	このパネルはビジネスオブジェクト Employeesのアクション(またはコマンド) 詳細 を扱います。 この後の作業で、このコマンドハンドラーを実際のプログラムに取り替えることができます。 このパネルは入力可能です。テキストを削除後、独自のテキストを追加できます(RAD-PADから絵やイメージ も追加できます)。
	詳細を表示 コードウィザード RAD-PAD
E 🗄 🗎	メッセージ 準備 ASI002_TS JPN V20PGMLIB 10/07/06 20:10 🥥

- 2. プロトタイプのフィルターと同様に、プロトタイプのコマンド・ハ ンドラーは編集可能です。
- 3. [詳細]コマンドのコマンド・ハンドラーのすべてのテキストを選択

し、削除します。

👹 Employees	
ファイル(E) 編集(E) 表示(V) ヘルプ(E	♪ ウインドウ(₩) (フレームワーク)(管理)
③作成 首 白 印刷 角	🖹 🛇 🛕 🔟 🔲 🕼 🗐 🔘 🏟 🖾 🍓 🕸 🖉 📔 💋 ಶ 🥱
Employees	
LINPTOYCES DEM HR ERP Prototype 1 HR デモアブリクーション Human Relation か気に入り ろ気に入り Code Tables Employees P の ブログラミング テクニック 管理	▲ 名前 By Date of Birth By Salary 社員姓 S ダリストの消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消去 その消 その時代 日のの のの消去 その時代 日のの のの のの のの のの のの のの のの のの の
	詳細を表示 コード・ウィザード RAD-PAD

- 4. [RAD-PAD]ボタンをクリックします。
- 5. [RAD-PAD]ウィンドウで、[スナップショットのみ]ラジオ・ボタン をクリックします。

RAD-PAD	×
98/03/25 ADMIN1 Administratn Part 1 Met requirement D 98/05/03 ADMIN2 Administratn Part 2 D 98/05/05 ECD Economics Degree D 94/06/05 ENG English Degree With distinction D + データ変換に盛換文字が使用された可能性があります。 + F1= ヘルブ F3= 終了 F4=FROMPT F12= 取活し F14=MSG F21= 定更 F22= 副論	>
社員等時の登録 社員施 社員施 社員施 社員范 住所1 住所2 住所3 載度書等 	8
<ul> <li>全て表示</li> <li>スナッフ<sup>5</sup>/чットを除外</li> <li>● スナッフ<sup>5</sup>/чっトのみ</li> <li>フィルタ ファイル名の接頭辞</li> <li>イメージ・パス</li> <li>C: ¥PROGRA<sup>~</sup>1¥LANSAV12¥X_W</li> <li>プラクズ_</li> <li>使用上の注意: webアフリケーションで使用される、またには他の開発者のデスクトップ1こ配布 される フロトタイピッグイメージの場所は、区画のExecuteフォルタ1こ配置するのがハベストです。C: ¥PROGRA<sup>~</sup>1¥LANSAV12¥X_WIN95¥X_LANSA¥x_dem¥execute¥</li> </ul>	
再表示	

- 6. 資格情報の検索 / 保守画面のスナップショットを見つけて、コマン ド・ハンドラーにドラッグします。
- 7. 必要に応じて、コマンド・ハンドラーのスナップショットのサイズ を変更します。



スナップインする画面のスナップショットをコマンド・ハンドラーに表 示すると、刷新後のアプリケーションの外観をかなり簡単に予想できる ことがわかります。

- 8. [作成]コマンドのコマンド・ハンドラーを表示し、[イメージ・パレット]ボタンをクリックします。
- 9. Enrol New Employee画面を見つけて右クリックし、コンテキスト・ メニューを表示して[コピー]を選択します。
- 10. [Employees]ウィンドウをクリックし、イメージを貼り付けます。
- 11. 必要に応じて、イメージのサイズを調整します。

4	er	nþ	lo	ye	e																
Γ													ł	Å	情	報	٥)	ŝ	2 <del>4</del>		<u>^</u>
	41		υ.																		
	fI	Д	10																		
	社	Â	ł <u>t</u>																		
	社	Â	名																		
	住	所																			=
	住	所	2																		
	住	所	3																		
	鄞	æ	800	ulu.															0		
	白	ξų.	电	đ	田田	10															
	鹄	門	电	話	曲	에이															_
	铫	88	٦		F														+		
	100			ŀ																	
	社	Å		두															.00		
	λ	社	Β	付	(	DD	) MIN	IY	()										0/00/00 +		
		覵	Β	付	(	DD	) MN	IY	()										0/00/00 +		
																					~
<																					>
																			コード・ウィザード	RAD-PAD	検索のサンフル

### RAMP-TS005 ステップ2. プロトタイプの検証

プロトタイプが視覚的に完成したら、エンド・ユーザーと一緒に検証 し、設計がエンド・ユーザーの期待と一致していることを確認します。 最も成功しているRAMPソリューションは、すべて注意深いプロトタイ プの作成を基盤として作成されています。 まとめ

重要なポイント

- エンド・ユーザーに現実的なプロトタイプを示すことで、エンド・ ユーザーは設計を理解し、有意義なフィードバックを提供できます。
- 画面ラッパーまたはVisual LANSAコンポーネントを使用するコマンド・ハンドラーの場合は、「RAMP-TS001 ステップ5. フィルターの表示」でフィルターに対して実行した場合と同様にテキストとイメージ・パレットを使用します。
- スナップショットは、プロトタイプ作成ステージの後も役に立ちます。刷新プロセスの一部として画面を変更する場合は、スナップショットをプロジェクト・ドキュメントで使用したり、基準点として使用できます。実際の画面をフレームワークにスナップインした場合は、[Show Snapshot]ボタンを使用してスナップショットを表示できます。

覚えるべき事柄

• プロトタイプを視覚的に完成させる方法

## RAMP-TS006:基本的な照会画面のスナップイン このチュートリアルでは、資格情報の検索/保守画面をフレームワーク にスナップインします。これは、代表的な基本5250照会画面です。 目的

 スナップインした画面をインスタンス・リストに関連付ける方法を 学ぶ

de Employees		
ファイル(Ĕ) 編集(E) 表示(V) ヘノ	プ(H) ウィンドウ(W) <mark>(フレームワーク)</mark> (管理)	
④作成 🖆 🖨 印刷		外小検索
このチュートリアルでは、スナ 画面をインスタンス・リストに 方法を学習します エンド・ユーザーがインスタン、 社員を選択すると、選択し 詳細が画面に表示され	ッブインした リンクする A、リストから た社員の ます	wee 記述 SMITHSON FRED SMITHE Robert SMITHE Robert SMITHE Robert
Tenhessuhe she h	Less Employee : 単純 (A0193-SMITHSON FRED)	
⊞ 😳 プログフミノジ アワ_ゥク Ⅲ  管理		
	2.666 2.757 2.757 2.757 2.66 6 数の検索/係守	Enter H
	社員Na	PROMPT スカリ・
	社員名 FRED	変更
	住所 2	<b>削除</b>
	住所 3	6-465: バューバーを表示
	部門コート	スナップジョットを表示
	課コート: 05 + 人社日付 (DDMMYY): 89/07/03 温融日母 (CDMMYY)	3 + ドキュメンテーション
E E 🔒	メッセージ 準備 ローカル JPN V20PGM	MLIB 10/07/09 17:05 🍛

この目的を達成するには、以下のステップを完了してください。

- RAMP-TS006 Step 1. フレームワークへの基本的な照会画面のスナッ プイン
- RAMP-TS006 ステップ2. 現在のインスタンス・リスト・エントリー を使用するようにスクリプトを変更する
- RAMP-TS006 ステップ3. ファンクション・キーの無効化
- まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証

## RAMP-TS006 ステップ1. フレームワークへの基本的な照会画 面のスナップイン

このステップでは、資格情報の検索/保守画面をフレームワークにスナップインし、テストします。

1. [RAMPツール]ウィンドウの画面とスクリプトのリストで DisplayEmployee Destination画面を選択します。

imployees セッション	·		
保存 🔻	- 新5250アフツケーション	セッション	削除
		検索	次を検索
名前			
ा रुअ्व) ा ा ा ा ा ा ा ा ा ा ा ा ा ा ा ा ा ा	tion (4) ination (3) ABMain mployee layEmployee		
	ial (2)		

DisplayEmployeeの詳細で、Employeesビジネス・オブジェクトの[詳細]コマンド・ハンドラーに画面をスナップインします。

*
· 
*
'±外内 _
<sup>*</sup> 亚外内
*ェ外内 🔨
TALK A.
Table
Table
ucts
ucts
ucts
icts
0010
ucts
ucts Tables
ucts Tables nyees
ucts Tables Dyees Dyees
ucts Tables ovees ovees ovees
ucts Tables byees byees byees
lable ucts ucts ucts ucts

- 3. [保存]を実行します。
- フレームワークを表示し、インスタンス・リストから社員を選択してコマンド・ハンドラーをテストします。



画面に適切に名前を付けて分類し、ナビゲーションを追跡した場合は、 フレームワークで資格情報の検索/保守画面が機能します。 ただし、インスタンス・リストの社員を選択した場合、この変更はコマンド・ハンドラーに反映されません。これは、画面を表示するために生成したスクリプトで、画面をトラッキングするときに使用した従業員番

号が記録されているためです。

RAMP-TS006 ステップ2. 現在のインスタンス・リスト・エン トリーを使用するようにスクリプトを変更する

このステップでは、ハードコードされた従業員番号を、現在選択してい るインスタンス・リストのエントリーの従業員番号に置き換えて、社員 の詳細を表示するスクリプトを変更します。

このステップを学習する場合は、「Javascriptの基本要素」を参照してく ださい。

- 1. 画面とスクリプトのリストでFindEmployee Junction画面を選択しま す。
- 2. 展開して、それに関連付けられているスクリプトを表示します。
- 3. リストからスクリプトを選択し、その内容を表示します。



4. NAVIGATETOファンクションで、以下のステートメントを見つけま す。

SETVALUE("txtEmpno","A1234");

- 5. 引用符で囲んだ従業員番号を選択し、右クリックしてコンテキス ト・メニューを表示します。
- メニューの[インスタント リスト エントリー]オプションを選択します。

#### 7. サブメニューの[Alpha Key 1]を選択します。



#### ステートメントが以下のように変わります。

SETVALUE("txtEmpno",objListManager.AKey1[0]);

この特殊な値には、エンド・ユーザーがインスタンス・リストで選択した社員のIDが使用されます。

8. [変更をコミット]ボタンをクリックし、スクリプトへの変更をコミット します。

📄 🛍 📈 🥱 😋 🗟 🥥 🏈 🗣 💀 🖗 🚱 🚱 🥥 🏈 💾 🛷 スクリーン名検索

- 9. RAMP情報の部分的な保存を実行します。
- 10. フレームワークを表示し、インスタンス・リストから社員を選択し ます。

以上で、選択した社員の適切な詳細が表示されます。何らかの理由によ り社員の詳細を表示できない場合は、フレームワークを再起動します。

#### RAMP-TS006 ステップ3. ファンクション・キーの無効化

フレームワークではマウスを数回クリックするだけで移動できるため、 このステップでは、ナビゲーションで使用する5250キーを無効にしま す。このため、ナビゲーション目的で使用されるファンクション・キー とボタンは不要になります。

刷新されたプロジェクトでは、5250ナビゲーション・ボタンを削除する 必要があります。これにより、アプリケーションはWindowsインター フェースを使用していた人にとってより親しみやすく、快適になりま す。

- 1. [RAMPツール]ウィンドウを表示します。
- 2. 画面とスクリプトのリストでDisplayEmployee画面を選択します。
- 3. [ファンクション キー使用可能]リストで、[終了]、[取消し]、[MSG]キーおよび ボタンを無効にします。

<b>‡</b> -	Caption	利用可能	利用可	FKEY :	^
Enter	Enter	<ul> <li>Image: A set of the set of the</li></ul>	<ul> <li>Image: A set of the set of the</li></ul>	1	
F1	F1			2	
F2	F2			3	
F3	終了			4	
F4	PROMPT	~	<ul> <li>Image: A start of the start of</li></ul>	5	
F5	F5			6	
F6	F6			7	
F7	F7			8	
F8	F8			9	
F9	F9			10	
F10	F10			11	
F11	F11			12	
F12	取消し			13	
F13	F13			14	
F14	MSG			15	-
E16	E15			10	~

- 4. RAMP定義の部分的な保存を実行します。
- 5. フレームワークに切り替えて社員の詳細を表示し、ボタンが表示されていないこと、およびキーが機能しないことを確認します。



[スクリーンの調査]、[スナップショットを表示]、[ドキュメンテーション]、[トレース オン]ボタンは、RAMPの設計時専用ボタンです。

まとめ

重要なポイント

- 自動的に生成されたスクリプトには、ナビゲーションを追跡すると きに選択した、社員のハードコードされた番号が含まれています。イ ンスタンス・リストで現在選択しているエントリーに画面が対応する ように、スクリプトを変更する必要があります。
- [スクリプトポップアップメニュー]を使用して、スクリプトを変更します。

覚えるべき事柄

• 画面がインスタンス・リストとの対話処理を行うようにする方法

RAMP-TS007:データ入力ファンクションのスナップイン このチュートリアルでは、社員情報の登録画面をフレームワークにス ナップインします。これは、代表的な5250データ入力画面です。 目的

- データ入力ファンクションをフレームワークにスナップインする方 法を学習する
- 画面のスクリプトの各種ファンクションを理解する
- 画面がフレームワーク・フィルターにイベントを通知する方法を学習する
- 画面の基本的なエラー処理を作成する方法を学習する



この目的を達成するには、以下のステップを完了してください。

- RAMP-TS007 Step 1. フレームワークへの基本的なデータ入力画面の スナップイン
- RAMP-TS007 ステップ2. 新規従業員番号を通知するようにスクリプトを変更する
- RAMP-TS007 ステップ3. エラー処理の追加

- RAMP-TS007 ステップ4. スクリプトを変更してインスタンス・リストを更新する
- RAMP-TS007 ステップ5. ファンクション・キーの無効化
- まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003: 社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン

RAMP-TS007 ステップ1. フレームワークへの基本的なデータ 入力画面のスナップイン

このステップでは、資格情報の検索 / 保守画面をフレームワークにスナップインし、テストします。

- 1. [RAMPツール]ウィンドウで、画面とスクリプトのリストの NewEmployee Destination画面を選択します。
- NewEmployeeの詳細で、画面をEmployeesビジネス・オブジェクトの [作成]コマンド・ハンドラー内にスナップインします。

あさ 380					
	<b>16</b> 700	上	左		
「op マスクの高さ		Bottom マスクの帖	ā		
RAMP スクリーンレイン	የኃኑ አያብሥ				
💿 セッシ	<i>и</i>	0	固定レイアウト	🔘 ንローレイアウト	
1-9-9h		PSLSYSMain			
保存					
			保存		
			保存		
マンクション キー使用	可能		保存		
ファンクション キー使用 キー Caption	可能	FKEY :	保存 - 関連するコマンド ハンドラ コマント* (タフ*)へのソンク	- 	~
マンクション キー使用 キー Caption Enter Enter	可能 利用可省利用 	] ना FKEY ह ]	保存 - 関連するコマンド・ハンドラ コマント* (タフ*)へのソンク 「印刷」	- オブジェクト内 Products	
マンウション キー使用 キー Caption Enter Enter F1 F1	可能」  利用可省 利用  ▼ □ □	वन FKEY द A	保存 - 関連するコマンド・ハンドラ コマント* (タフ*)へのリンク 「印刷 「販売	- オプジェクト内 Products Products	~
マンクション キー使用 キー Caption Enter Enter F1 F1 F2 F2		307 FKEY € ▲ 1 2 3	保存 - 関連するコマンド・ハンドラ コマント* (タワ*)へのリンク - 印刷 - 販売 - 作成	オプジェクト内 Products Products Products Products	^
マンクション キー使用 キー Caption Enter Enter F1 F1 F2 F2 F3 終了		307 FKEY € ▲ 1 2 3 4	保存 - 関連するコマンド・ハンドラ コマント* (タワ*)へのリンク - 印刷 - 販売 - 作成 - 詳細	- オフ <sup>s</sup> ジェクト内 Products Products Products Code Tables	
<del>マンクション キー使用 キー Caption</del> Enter Enter F1 F1 F2 F2 F3 終了 F4 PROMPT		307 FKEY € ▲ 1 2 3 4 5	保存 - 関連するコマンド・ハンドラ コマント* (タワ*)へのリンク - 印刷 - 販売 - 作成 - 詳細 - 詳細	- オフ <sup>s</sup> ジ*ェクト内 Products Products Products Code Tables Employees	^
マンクション キー使用 キー Caption Enter Enter F1 F1 F2 F2 F3 終了 F4 PROMPT F5 F5		307 FKEY € ▲ 1 2 3 4 5 6	保存 - 関連するコマンド・ハンドラ コマント* (タワ*)へのリンク - 印刷 - 販売 - 作成 - 詳細 - 詳細 - 詳細 - 詳細 - 詳細 - 「新規	オフ <sup>×</sup> ジ <sup>×</sup> ェクト内 Products Products Products Code Tables Employees Employees	^
オー Caption キー Caption Enter Enter F1 F1 F2 F2 F3 終了 F4 PROMPT F5 F5 F6 F6		307 FKEY € ▲ 1 2 3 4 5 6 7	保存 - 関連するコマンド・ハンドラ コマント* (タワ*)へのリンク 印刷 - 販売 - 作成 - 詳細 - 詳細 - 詳細 - 詳細 - 「詳細 - 「詳細 - 「詳細	オフ <sup>×</sup> ジ <sup>×</sup> ェクト内 Products Products Products Code Tables Employees Employees Employees	^
<mark>キー Caption</mark> キー Caption Enter Enter F1 F1 F2 F2 F3 終了 F4 PROMPT F5 F5 F6 F6 F7 F7		307 FKEY € ▲ 1 2 3 4 5 6 7 8	保存 - 関連するコマンド・ハンドラ コマント* (タワ*)へのリンク 印刷 回販売 一作成 一詳細 一詳細 「詳細 「詳細 「詳細 「詳細 「「「キュメント 「ハ*」ト*キュメント 「ハ*ント	オフ <sup>×</sup> シ <sup>×</sup> ェクト内 Products Products Products Code Tables Employees Employees Employees Employees	
<del>マンクション キー使用 キー Caption</del> Enter Enter F1 F1 F2 F2 F3 終了 F4 PROMPT F5 F5 F6 F6 F7 F7 F8 F8		307 FKEY € ▲ 1 2 3 4 5 6 7 8 9	保存 - 関連するコマント <sup>*</sup> ハント <sup>*</sup> ラ コマント <sup>*</sup> (タワ <sup>*</sup> )へのリンク 印刷 - 販売 - 作成 - 詳細 - 詳細 - 詳細 - 詳細 - 「詳細 - 「詳細 - 「詳細 - 「詳細 - 「「新規 - ト <sup>*</sup> キュメント - ヘ <sup>*</sup> ント	オフ <sup>*</sup> シ <sup>*</sup> ェクト内 Products Products Products Code Tables Employees Employees Employees Employees	
<del>マンクション キー使用 キー Caption</del> Enter Enter F1 F1 F2 F2 F3 終了 F4 PROMPT F5 F5 F6 F6 F7 F7 F8 F8 F9 F3		307 FKEY € ▲ 1 2 3 4 5 6 7 8 9 10	保存 - 関連するコマント <sup>*</sup> ハント <sup>*</sup> ラ コマント <sup>*</sup> (タワ <sup>*</sup> )へのリンク 印刷 - 販売 - 作成 - 詳細 - 詳細 - 詳細 - 詳細 - 「詳細 - 「詳細 - 「「「「」」、 - 「「、」、 - 「、、」、 - 「、、」、 - 「、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、	オフ <sup>×</sup> シ <sup>×</sup> ェクト内 Products Products Products Code Tables Employees Employees Employees Employees	~

3. [保存]を実行します。

- フレームワークを表示し、ナビゲーション・ペインでEmployeesビジネス・オブジェクトを選択します。次に、フィルターを使用してインスタンス・リストに入力し、Employeesビジネス・オブジェクトまたはインスタンス・リストの1人の社員を右クリックして、コンテキスト・メニューを表示します。
- 5. [作成]を選択します。



フレームワークで、Enrol New Employee画面が表示されます。

💰 Employee						
社員情報の登録	Enter					
社員Na	PROMPT					
社員名	取消し					
住所 1	MSG					
住所 3	火ニューバーを表示					
	スナップジョットを表示					
部門電話番号	トドキュメンテーション					
課 コード · · · · · · · · · · · · · · · · · · ·	トレースオン					
入社日付 (DDMMYY) 0/00/00 +	スタリーンの調査					
28月1日17 (DDMMYY)						
★ H:404 W:651 メッセージ レコート*サイス*						

- 6. 新しい社員の詳細を入力し、Enterキーを押して保存します。
- 7. [名前]フィルターを使用して、新しく追加した社員を検索します。

次のステップでは、社員の作成画面を変更してインスタンス・リストを 自動的に更新します。

## RAMP-TS007 ステップ2. 新規従業員番号を通知するようにスクリプトを変更する

このステップでは、新しい社員を作成したときに新規従業員番号を通知 するコードを画面スクリプトに追加します。

1. [RAMPツール]ウィンドウで、NewEmployee画面のスクリプトを見つ けます。



- BUTTONCLICKファンクションを見つけ、次にbReturn変数を宣言しているステートメントを見つけます。
- 3. このステートメントを追加し、oEmpオブジェクトを宣言します。これは、従業員番号を渡すために使用されます。

var oEmp = new Object();

#### コードは、以下のようになります。

```
vHandle_BUTTONCLICK: function(sButton)
{
    var bReturn = true;
    var oEmp = new Object();
    if (HANDLE_PROMPT()) return(bReturn); /* If the focus element is au
```

- 次に、Enterキー(KeyEnter)を押す操作を処理するステートメントを 見つけます。
- 5. このステートメントをSENDKEYステートメントの前に追加し、新 しい従業員の従業員番号の値をoEmpオブジェクトに取得します。

oEmp.empno = GETVALUE("txtEmpno");

6. SENDKEYファンクションを変更し、従業員番号ペイロードを渡し ます。

SENDKEY(KeyEnter, oEmp);

コードは以下のようになります。

```
vHandle_BUTTONCLICK: function(sButton)
ł
  var bReturn = true;
 var oEmp = new Object();
  if (HANDLE_PROMPT()) return(bReturn); /* If the focus element is au
  /* <BUTTONCLICK /> - Do not remove or alter this line */
        /* Handle function keys and buttons */
        switch (sButton)
     {
        case KeyEnter:
              oEmp.empno = GETVALUE("txtEmpNo");
              SENDKEY(KeyEnter, oEmp);
              break:
        case KeyF3:
              SENDKEY(KeyF3);
              break;
```

#### 7. 変更をコミットします。

以降のステップでは、従業員番号ペイロードを使用します。

#### RAMP-TS007 ステップ3. エラー処理の追加

このステップでは、コードをスクリプトに追加して、新しい社員が正常 に挿入された場合にメッセージを発行します。

挿入を試行した後(成功したかどうかに関係なく)、NewEmployeeフォームが再表示されます。したがって、画面のARRIVEスクリプトにエラー処理コードを配置できます。次に、新しい社員が正常に追加された場合に[社員の作成]ウィンドウを閉じるためのコードを追加します。

エラー処理の場合、従業員番号を切り取るJavascriptファンクションが必要になります。

1. このファンクションのコードをスクリプトの最後、SYSINFOの直前 に追加します。

/\* \_\_\_\_\_\_ /\* \_\_\_\_\_\_ /\* sStringToTrim: The string to be trimmed left and right \*/

uTrim : function (sStringToTrim) {

return sStringToTrim.replace(/^\s+|\s+\$/g,"");

},

コードは以下のようになります。

```
/* </NAVIGATESWITCH> - Do not remove or alter this line */
 return(bReturn);
}.
```

```
/* _____ *//
/* sStringToTrim: The string to be trimmed left and right */
uTrim : function (sStringToTrim) {
return sStringToTrim.replace(/^\s+|\s+$/q,"");
},
```

NewEmployee画面が再表示されたときのみエラー処理コードを実行しま す。このため、最初に前の画面が何であったかをチェックする必要があ ります。

2. このステートメントをvHandle ARRIVEファンクションのbReturn変 数の宣言の後に追加します。

```
if (oPreviousForm.vName == "NewEmployee")
{
```

挿入の成功と検証エラーを区別するために、従業員番号が操作でブラン クに設定されていたかどうかをチェックするコードを追加します。ブラ ンクではない場合は、ウィンドウを閉じます。

3. このコードをifステートメントに追加します。

/\* Get the currently showing EMPNO \*/ var strCurrEmpno = GETVALUE("txtEmpno"); if ((this.uTrim(strCurrEmpno) == "") && (this.uTrim(oPayload.empno) != Ł /\* Insert was sucessful \*/ /\* Issue a message \*/ MESSAGE("Employee ", oPayload.empno ," created"); AVCLOSEFORM(); }

#### コードは以下のようになります。

vHandle\_ARRIVE: function(oPayload, oPreviousForm)
{
 var bReturn = true;
 if (oPreviousForm.vName == "NewEmployee" )
 {
 /\* Get the currently showing EMPNO \*/
 var strCurrEmpno = GETVALUE("txtEmpno");
 if ((this.uTrim(strCurrEmpno) == "") && (this.uTrim(oPayload.empno) != "") )

if ((this.uTrim(strCurrEmpno) == "") && (this.uTrim(oPayload.empno) !=
{
 /\* Insert was sucessful \*/
 /\* Issue a message \*/
 MESSAGE("Employee ", oPayload.empno ," created");
 AVCLOSEFORM();
 }
}

SHOW\_CURRENT\_FORM(true); /\* Show the form in the framework and show VLF buttons \*/

- 4. [変更をコミット]をクリックし、[保存]を実行します。
- フレームワークで、新しい社員を追加します。社員が正常に追加されたら、メッセージが表示され、ウィンドウが閉じることに注意してください。


## RAMP-TS007 ステップ4. スクリプトを変更してインスタン ス・リストを更新する

このステップでは、新しい従業員を作成したときにインスタンス・リストを更新するコードを画面スクリプトに追加します。

1. エラー処理コードのMESSAGEファンクションの後に以下のステー トメントを追加し、従業員が正常に作成された後、フィルターにシグ ナルを送信してインスタンス・リストを更新します。

AVSIGNALEVENT("Add\_List\_Entry","BUSINESSOBJECT", oPayload.empno)

## コードは以下のようになります。

```
vHandle_ARRIVE: function(oPayload, oPreviousForm)
{
  var bReturn = true;
  if (oPreviousForm.vName == "NewEmployee" )
  {
     /* Get the currently showing EMPNO */
     var strCurrEmpno = GETVALUE("txtEmpNo");
     if ((this.uTrim(strCurrEmpno) == "") && (this.uTrim(oPayload.empno) != "") )
     {
        /* Insert was sucessful */
        /* Insert was suc
```

SHOW\_CURRENT\_FORM(true); /\* Show the form in the framework and show VLF buttons \*/

「RAMP-TS003 ステップ1. 実際の名前フィルターを作成する」でフィル ターを作成した後、インスタンス・リストを更新するシグナルをリッス ンするコードをフィルターに含むように指定しました。このため、この シグナルをリッスンするコードがすでにフィルターにあります。

- 2. [変更をコミット]ボタンをクリックして、コードの変更を保存します。
- 3. RAMP定義の部分的な保存を実行します。
- 4. フレームワークを表示します。
- 5. フィルターを使用して、姓が「S」で始まるすべての従業員を見つけ ます。

6. 「S」で始まる名前の新しい従業員の詳細を入力します。

🍰 Employee	
社員情報の登録	Enter
社員No	PROMPT
社員姓 SALINGER	
任月名 ·····BETTY	取消し
1日的1	MCC
1Eff 2 North Sydney	MDG
住所 3 NSW Australia	オールーを表示
郵便番号	) // E30//
自宅電話番号	スナップ <sup>®</sup> ショットを表示
部門電話番号	
部門 그 - ト · · · · · · · · · · · · · · · · · ·	ト・キュメンテーション
課 コード	11
社員給与	NU-2 72
入社日付 (DDMMYY)	ったい」、小の調査を
退職日付 (DDMMYY)	大77-703前日
✓ ▲ H:418 ₩:702	ッセージ レコート・サイズ

7. [入力]をクリックします。新しく作成した従業員がインスタンス・ リストに表示されます。

🗆 🥝 🔺 📄	
Employee 記述	
A0193 SMITHSON F	RED
A1002 SMYTHE JOH	IN
A1003 SMITHE Rob	ert
A1004 SMITHSON P	AUL
A1005 SMITHS PET	ER
A1006 SMITHERS J	ACK
A1007 SNELL GEOR	GE
A1008 SNEDDON A	LLAN
A1009 SNASHALL D	AMIAN
A8345 SALINGER B	ETTY

**RAMP-TS007 ステップ5. ファンクション・キーの無効化** このステップでは、刷新されたアプリケーションで不要になるファンク ション・キーを無効にします。また、対応するボタンを非表示にしま す。

- 1. [RAMPツール]ウィンドウの画面とスクリプトのリストで NewEmployeeを選択してその詳細を表示します。
- 2. [ファンクション キー利用可能]リストで、[入力]と[プロンプト]以外のすべて のキー/ボタンを無効にします。
- 3. [入力]のキャプションを[保存]に変更します。

<b>‡</b>	Caption	利用可能	利用可	FKEY s	1
Enter	保存	<ul> <li>Image: A set of the set of the</li></ul>	<b>~</b>	1	
F1	F1			2	
F2	F2			3	
F3	終了			4	
F4	PROMPT	<b>~</b>	<b>~</b>	5	
F5	F5			6	
F6	F6			7	_
F7	F7			8	
F8	F8			9	
F9	F9			10	

- 4. RAMP定義の部分的な保存を実行します。
- 5. フレームワークを表示し、クリックして新しい社員を追加して正し いボタンが表示されていることを確認します。



フレームワークをデザイン・モードで実行しているときは、最下部の4つ のボタンのみが表示されます。

## まとめ

## 重要なポイント

- SENDKEYファンクションには省略可能なPayloadパラメータがあ り、これを使用して情報を渡すことができます。
- イベントをフレームワークのフィルターに通知するには、 AvSignalEventファンクションを使用します。

## 覚えるべき事柄

- データ入力ファンクションをフレームワークにスナップインする方法
- 画面のスクリプトの各種ファンクション
- 画面がフレームワーク・フィルターにイベントを通知する方法
- 画面の基本的なエラー処理を作成する方法

## RAMP-TS008: 照会画面の更新モードへの変更

このチュートリアルでは、EmployeeDetails画面を更新モードに変更して 変更できるようにします。

目的

- 照会画面を更新画面に変更する方法を学習する
- 更新画面とその直近のJunction間のナビゲーションを処理する方法 を学習する
- スクリプトの構造について詳しく理解する

着 Employees		
ファイル(E) 編集(E) 表示(V) ヘルプ(H)	ウィントウヘ₩)(フレームワーク)(管理)	
④作成 台 串印刷 沿	S 🔺 🗵 🗖   🖉 🖾 🖉   🙆 🖾 🍇 😹	② 外ック検索
Employees		
* DEN 10	× 名前 By Date of Birth By Salary	×
R	L貝姓 検索	Employee 記述
1	SALINGER	A1005 SMITHS PETER
	リストの消去	A1006 SMITHERS JACK
このチュートリアルでは、更新		A1007 SNELL GEORGE A1008 SNEDDON ALLAN
モートでDisplayEmployee画面を まテレズ 亦再 できる トラにオる 古法		A1009 SNASHALL DAMIAN
ないして変更できるように9 の力法、 お上び保管後に画面とその直近の		A8345 SALINGER BETTY
Junction間の対話処理を扱う方法		x
について学習します。	A1009-SNASHALL DAMIAN)	
L J	愛格 情報の 検索/ 様	マ 保存
	社員Na: A100	9 PROMPT
	社員班 SNAS	HALL パーカーカー
	住所1	Sackville Road
	住所 2 INGL	EBURN. スナッフ ジョットを表示
	住所3 NSW. 自宅電話番号 605	ドキュメンテーション
	部門コート AUD	+
	課コード 02 +	
	2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2. 2	2/01 + 又卯一次0周首
	▲ メッセージ 準備 AS1002_TS JP	N V20PGMLIB 10/07/13 14:58 🥥

この目的を達成するには、以下のステップを完了してください。

- RAMP-TS008 Step 1. Display Employee画面を入力可能にする
- RAMP-TS008 ステップ2. 保存後のDisplayEmployeeの再表示
- RAMP-TS008 ステップ3. ボタン・キャプションの変更

#### • まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン

RAMP-TS008 ステップ1. Display Employee画面を入力可能にする

このステップでは、DisplayEmployee画面を更新モードで表示します。 チュートリアル「RAMP-TS006:基本的な照会画面のスナップイン」で は、基本的な読み取り専用照会画面をフレームワークにスナップインす る方法を示して、いくつかの基本的なステップを示しましたが、通常、 実際の刷新されたアプリケーションでは、画面を入力可能として表示し ます。

「RAMP-TS004 ステップ1. 画面の命名」では、DisplayEmployee画面の [txtSurname]フィールドに名前を付けました。このフィールドは、画面が 変更モードの場合のみ表示されます。したがって、これを使用して画面 のモードを判定および設定できます。

- 1. [RAMPツール]ウィンドウで、画面とスクリプトのリストの DisplayEmployee画面を選択します。
- 2. 展開して、画面に関連付けられているスクリプトを表示します。
- vHandle\_ARRIVEファンクションで、bReturnの定義の後に以下の コードを追加し、[txtSurname]フィールドが存在するかどうかを確認し ます。存在しない場合は、画面を変更モードに設定します。

コードは、以下のようになります。

i	<pre>/* ====== */ /* ====== vHandle_ARRIVE ====== */ /* Handle arrival at this Destination */ /* oPayload: The payload supplied by the event initiator */ /* oPreviousForm: Reference to previous object Form*/ vHandle_ARRIVE: function(oPayload, oPreviousForm) {    var bReturn = true;</pre>	
	<pre>if (CHECK_FIELD_EXISTS("txtSurname")) {    SHOW_CURRENT_FORM(true); /* Show the form in the framework and show VLF buttons    HIDE_5250_BUTTONS(); /* Hide any 5250 style buttons displayed    GET_FORM_MESSAGE(22); /* Extract messages and hide the message line    SETBUSY(false); /* Last thing done - turn off the busy state    }    else    {    SENDKEY(KeyF21);    } }</pre>	*// *//
	<pre>/* <arrive></arrive> - Do not remove or alter this line */    return(bReturn); },</pre>	
	/* ====================================	

- 4. 変更をコミットします。
- 5. RAMP定義の部分的な保存を実行します。
- 6. フレームワークを表示します。
- 7. インスタンス・リストから社員を選択します。資格情報の検索/保 守画面が変更モードで表示されることに注意してください。
- 8. 社員の名前を変更します。
- 9. Enterを押します。

保管後にFindEmployee画面が表示されます。

👹 Employees		
ファイル(E) 編集(E) 表示(V) ヘルプ()	±り ウィントや(W) (フレームワーク) (管理)	
④作成	🖹 💫 🔺 💷 🛑 🐺 🛱 💭 🗎 🏟 🖄 📓	外吻検索
Employees		
★ DEM HR ERP Prototype 1 HR デモアブサウーション Human Relation か お気に入り Code Tables Employees ● ブ*ログ*ラミンウ* テウニック ● 管理	名前       By Date of Birth       By Salary         社員姓       検索         SALINGER       A100         ソリストの消去       A100         く       ●         Employee : 詳細 (A8345-SALINGER BETTY)         ●       詳細         ▲ イペント       トドキュメント         愛悟 情報 の快楽 / 保守         社員 №       ▲ 8345 +	× ovee 記述 へ S SMITHS PETER 6 SMITHERS JACK 7 SNELL GEORGE 8 SNEDDON ALLAN 9 SNASHALL DAMIAN 5 SALINGER BETTY ✓ × × × × × × × × × × × × ×
	メッセージ 準備 ASIDO2_TS JPN 1	V20PGMLIB  10/07/13  13:49 🥥

これは、フレームワークのナビゲーション・モデルに合わせて変更する 必要がある一般的な5250の動作です。次のステップでは、社員の詳細が 保管された後に資格情報の検索/保守画面を再表示するFindEmployeeス クリプトを作成します。

## RAMP-TS008 ステップ2. 保存後のDisplayEmployeeの再表示

このステップでは、社員の詳細が保管された後、資格情報の検索/保守 画面に戻るようにFindEmployee画面のスクリプトを変更します。

- 1. FindEmployeeのスクリプトで、vHandle\_ARRIVEファンクションを見つけます。
- 2. 宣言var bReturn = true;の下に新しいオブジェクトを作成します。

var oEmp = new Object();

3. 以前の画面がDisplayEmployeeであるかどうかをチェックするコード をステートメント/\* <ARRIVE /> - Do not remove or alter this line \*/の下 に追加します。

if (oPreviousForm.vName == "DisplayEmployee")
{

ifステートメントでは、フィルターを使って新しい社員の情報によりインスタンス・リストを更新する必要があることをフレームワークに通知します。

oEmp.empno = GETVALUE("txtEmpno"); AVSIGNALEVENT("Update\_List\_Entry","BUSINESSOBJECT", oEmp.ei

5. 以下のコードを追加して、DisplayEmployee画面に戻ります。

NAVIGATE\_TO\_SCREEN("DisplayEmployee");

コードは以下のようになります。



- 6. 変更をコミットし、[保存]を実行します。
- 7. フレームワークで、社員の詳細を更新します。DisplayEmployee画面 が再表示され、インスタンス・リストに社員の詳細の変更が反映され ています。

🔞 Employees		
ファイル(E) 編集(E) 表示(V) ヘルプ(	Ⴞ) ウィンドウ(₩) (フレームワーク) (管理)	
③作成 台 白刷 角	📔 🛯 🔺 💷 🗖 🐺 🖽 🖓 🖾 🚳 📾 🛤 🖉 📔 👂	心検索
Employees		
E SE DEN HR	二名前  一 By Date of Birth 一 By Salary   □ 〇 介   □	x
🕀 💕 ERP Prototype 1 📼 🦉 HR ተኛዋንን የሀሳ- ዓቃን		e 🔒
🗄 🚰 Human Relation	SMYTHE	THSON ERED
🗉 🔆 My Application	A1002 SMY	THE JOHN
● ☆ お気に入り	✓ 9次ND/月去 A1003 SMIT	THE Robert
🖉 🚺 Code Tables	A1004 SMIT	THSON PAUL
Employees	A1005 SMI	
田 ● 7・ログ・ラミング・デクニック     田 ● 第1		
🗉 🛷 ह.त		×
	Employee : ## (A1002-SMYTHE JOHN)	<u>u</u>
	🛄 詳細   🔔 イベント   📄 ドキュメント	
	資格 情難 の 検 魚 / 俳 守	Enter N
	社員Na	PROMPT スクリ・
	11月9日 · · · · · · · · · · · · · · · · · · ·	亦面
	住所 1	AX
	住所 2 WERRINGTON.	肖耶余
	住所 3 NSW. 自实 要該 最 号 047 539 0442	メニューバーを表示
	部門그-ト ADM +	ZtoJ <sup>®</sup> agh友表示
	課 コード	11/1 / / / Cach
	人社日付 (DDM-IYY)	トキュメンテーション
•••	▽ ▲ メッセージ 準備 ASI002_TS JPN V20PGMLIB	10/07/13 14:07 🍛

## RAMP-TS008 ステップ3. ボタン・キャプションの変更

このステップでは、[入力]ボタンのキャプションを[保存]に変更します。

- 1. [RAMPツール]ウィンドウで、DisplayEmployee画面を選択します。
- 2. [ファンクション キー利用可能]リストで、[入力]ボタンのキャプションを選択し、[保存]に変更します。

<b>‡</b> -	Caption	利用可能な5250	利用可能なと	KEY 🔺	^
Enter	保存			1	
F1	F1			2	
F2	F2			3	

- 3. [変更]ボタンと[削除]ボタンを非表示にし、無効にします。
- 4. RAMP定義の部分的な保存を実行します。
- 5. フレームワークに切り替えて、社員の詳細を表示し、ファンクション・キーのキャプションを確認します。

## まとめ

重要なポイント

- ナビゲーションを追跡して入力可能画面を作成できます。RAMPは 関連するスクリプトを自動的に更新します。
- 正常に保存した後、以前のJunction画面が表示されます。Junction画面に関連付けられているスクリプトのvHandle\_Arriveファンクションにコードを追加できます。

## 覚えるべき事柄

- 画面を更新モードでスナップインする方法
- 保存後にDestination画面とその直近のJunction間のナビゲーションを 処理する方法

# **RAMP-TS009**:ナビゲーションの追跡

このチュートリアルでは、アプリケーション・トレースを使用し、ある Destination画面から別のDestination画面に移動するときに発生する内容に ついて理解します。

目的

- アプリケーション・トレースを使用する方法を学習する
- ナビゲーション・スクリプトで使用するファンクションとコマンド について理解する

Employees     Eile Edit View Help Windows (Framework) (Administration)     Cのチュートリアルでは、アプリケーション・					
Spooled Files ¥ Sign Off 一 Reports □ Details Using SETCURSOR Destinction 回答 NO Particulture 回答 No Particulture 回答					
And the strength of the second strength of the	Destination画面から別のDestination画面 に移動するときに発生する内容について				
Application Level Trace Details	Application Level Trace Details				
Component Window Event					
VF_CH006 MAIN Posti	ng screen. Current form is DisplayEmployee (13:	y market and the second se			
VF_CH006 IMAIN EX	(ecution of script associated with DisplayEmploy) IKEV E21 completed. Server response will be associated associated by the server response will be associated by the server response by the serv	Fueleura Description			
VF_CH006 MAIN Executio	n of script associated with DisplayEmployee corr				
VF_CH006 MAIN Form with r	ame "DisplayEmployee" has arrived. (13:12:46:5	A0070 BROWN VERONICA			
VF_CH006 MAIN Execution	ig vHandle_ARRIVE function in form DisplayEmpl	A1031 BLOGGS JOHN			
VE_CHOOS MAIN CHEC	K_FIELD_EXISTS returned true (13:12:46:583)	A2005 BURGESS KEVIN			
VF_CH006 MAIN SHOW	W_CURRENT_FORM executed to show the currer $\checkmark$				
Save Trace to File Clear Trace JUHN ALANJ					
I ATresTech					
		Show Menu Bar			
Code Tables					
Employees	Employee Number	Show Snapshot			
🗄 💕 Proc_And_Close	Employee Given Name(s)	BLACK     EB ED 10HN Documentation			
🗄 🙀 RAMP Test	Street No and Name	· · 70 MAIN S			
🗄 👹 SubType Test	Suburb or Town	NEWTOWN N     Turn Trace Off			
🗄 💓 Tst Conn	State and Country	AUSTRALIA     Probe Screen			
H VisualStyle	Home Phone Number	· · 344 -22 344			
XL Spreadsneet     Al Programming Techniques	Department Code	FLT +			
a and a second design of the s	Start Date (DDMMYY)	3/08/92			
	Termination Date (DDMMYY)	0/00/00			
	Messages Ready VLFPGMLIB ENG	G VLFPGMLIB 8/04/09 14:40			

この目的を達成するには、以下のステップを完了してください。

- RAMP-TS009 ステップ1. トレースの開始とDestination画面の再表示
- RAMP-TS009 ステップ2. トレースの検証
- まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更

RAMP-TS009 ステップ1. トレースの開始とDestination画面の 再表示

このステップでは、アプリケーション・トレースを開始し、Destination 画面を再表示して、Destination画面が表示されたときに何が発生するか を理解します。

- 1. 人事アプリケーションのインスタンス・リストで従業員を選択しま す。
- 2. 従業員の詳細が表示されたら、フレームワーク・メニューを使用してアプリケーション・トレースを開始します。



- 3. ここで、インスタンス・リストから別の従業員を選択します。トレースの詳細がウィンドウに表示されます。
- 4. [トレース ファイルを保存]ボタンをクリックして、トレースの詳細

# をファイルに保存し、メモ帳で開きます。

アフ <sup>°</sup> リケーション	レヘットトレースの詳細			
コンホペーネント	<u> </u>	イላንኑ		×
VF_SY150 VF_SY150 TRNCOM05 VF_SY150 VF_SY150 TRNCOM05 VF_LM002 VF_SY150 VF_SY150 TRNCOM05 TRNCOM05 TRNCOM05	MAIN MAIN MAIN MAIN	Checking lock status (VF_FPM0 Returning lock status FALSE (VI Command Handler - uInitialize e Checking lock status (VF_UM03 Returning lock status FALSE (VI Command Handler - uExecute e GetCurrentInstance returns Vis Checking lock status (VF_UM04 Returning lock status FALSE (VI uChildWindowClosing Command Handler - uTerminate Command Handler - DestroyIns	9:mExecuteCommand) F_FPM09:mExecuteComm executed 37.TabChanging) F_UM037.TabChanging) executed sualID1(A1001) VisualID2( 6:CloseQuery) F_UM046:CloseQuery) e executed stance executed	nand) (J Ben) AKeys1-5(A: =
<				h. ▲
			トレースファイルを保存	トレース消去

RAMP-TS009 ステップ2. トレースの検証 このステップでは、トレースを検証します。



まとめ

重要なポイント

- トレースを使用して、ある画面から別の画面にRAMPが移動する方法を理解することができます。
- 問題が発生した場合は、トレースの使用を開始して解決します。
   「デバッグと診断」を参照してください。
- 付属のJava ScriptファンクションTRACE()を使用すると、独自のトレース・ステートメントを追加することができます。「トレース」を参照してください。
- メッセージ・ボックスを表示すると、問題を最も簡単に早く解決で きることがあります。「スクリプトでのALERT\_MESSAGEの使用」 を参照してください。

覚えるべき事柄

アプリケーションをトレースする方法

## RAMP-TS010:特別なフィールド・ハンドラーの使用

簡単なVisual LANSAフォームをフィールドに関連付けることによって、 5250 RAMP画面で高度なプロンプトを簡単に実行できます。このチュー トリアルでは、[部門コード]フィールドに特別なフィールド・ハンド ラーを追加してユーザーが部門リストからコードを選択できるようにし ます。

目的

- 特殊なフィールド処理を使用する基本について学習する
- 5250画面に価値を付加する方法を学習する



この目的を達成するには、以下のステップを完了してください。

- RAMP-TS010 Step 1. フィールドに名前を付ける
- RAMP-TS010 ステップ2. フィールドをハンドラーに関連付ける

• RAMP-TS010 ステップ3. 特別なフィールド・ハンドラーのテスト

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更
- RAMP-TS009:ナビゲーションの追跡

## RAMP-TS010 ステップ1. フィールドに名前を付ける

このステップでは、DisplayEmployee画面の[部門コード]フィールドに名 前を付けて、そのフィールドに特殊なフィールド処理を追加できるよう にします。

- 1. [RAMPツール]ウィンドウで、RAMP-TS 5250セッションを開始しま す。
- 2. DisplayEmployee画面に移動し、変更モードにします。
- 3. 画面命名エリアを表示し、部門コードを表示しているフィールドを 見つけて、utxtDepartmentという名前を付けます。
- 4. また、[セクション コード]フィールドにSECTIONという名前を付け ます。



## 5. [保存]ボタンをクリックして、画面定義を保存します。

# RAMP-TS010 ステップ2. フィールドをハンドラーに関連付ける

このステップでは、[部門コード]フィールドを特別なハンドラーに関連 付けます。

- 画面とスクリプトのリストで[セッション]を選択します。セッション・オブジェクトでは、セッションのすべての画面のフィールドと特別なフィールド・ハンドラーを関連付けます。
- 2. [特別なフィールド受け渡し]テーブルで、[utxtDepartment]フィール ドと特別なハンドラーDF\_PRM04を指定します。ハンドラーを呼び出 すキーとしてF2キーを使用します。

■ RAMP ツール - デフォルト セクション (RAMP-TS)			
テウォルト セッション	RAMP-TS 5250 セッション 詳細		
保存 🔹	-セッション - デウォルト セッション キャラション デ・フォルト セッション		
	ユーザー オノシェット S417 RAMP_DEFAULT_SESSION 名前の確認		
名前 ク 雇 - セッション - Junction (4)	RAMPの既定値のレイアウト サイズ*       高さ 380     幅 700       上     左       Top マスクの高さ     25       Bottom マスクの幅		
MainLopin 2     150SMainMenu 2     PSLSYSMain 2	-RAMP スクリーン レイアウト スタイル ● 固定レイアウト ○ フローレイアウト		
FindEmployee 2     Employee 2     Pestination (3)     PSLTABMain 2	- スケロールパーー		
DisplayEmployee 2     Special (2)     Elimitation (2)	<ul> <li>✓ 不明な52507ォームの表示時にフレームワークをロックする</li> <li>● 既存のコネクションのユーザー/パマワートを再利用する</li> </ul>		
DisplayMessages 2	セッションを常にユーザー オフジェット 名/タイフ を持ったサーバーヘリンクする		
፹ 🕋 አንሃን°ト (9)	RAMP-TS 最大口/7才7待線時間(秒) 特別はファールが受けての		
	52507/		
	1 utxtDepartment F2 DF_PRM04		
	2 F4		
	3 F4		
	4 F4		
	5 F4		

3. 画面とスクリプトのリストでDisplayEmployee画面を選択します。

4. F2キーを有効にします。

🔲 RAMP ツール ー デウォルト セッション	/ (RAMP-TS)
テラオルト セッション	RAMP-TS 5250 セッション 詳細
	BisplayEmployee 5250画面 スナップジョット
保存 👻	
	\$175m
	17/74/ hil1*
名制	-RAMPの既定値のレイアウト サイズ
	高さ 380 幅 700 上 左
🖃 🔛 Destination (3)	Top 77加高式 25 Pottom 77加加
NewEmployee     DisplayEmployee	RAMP 2/9-> 1/7/1 29/1
E Special (2)	
🗉 💽 አንሃን°ኑ (9)	-9-7*7h By
	PSLSYSMain
	(277 (277
	キー Caption 利用可能なVLF mp FKE) へ
	Enter Enter
	F4 PROMPT
	, <u> </u>
< >	

5. RAMP定義の部分的な保存を実行します。

RAMP-TS010 ステップ3. 特別なフィールド・ハンドラーのテ スト

このステップでは、特別なフィールド・ハンドラーをテストします。 1. フレームワークに切り替えます。

- 2. 人事アプリケーションで、社員の詳細を表示します。
- 3. [部門コード]フィールドにカーソルを置き、F2キーを押して特別な フィールド・ハンドラーを表示します。

🐇 Employees 📃 🗖 🗙					
ファイル(E) 編集(E) 表示	示♡ ヘルフ*(Ⴞ) ウィンドウ(艸) (フレームワーク) (管理)				
③作成 🗇 🗅 🖻 🔟 🛕 🕤 🗖 📮 🖾 🗊 🔿 🍥 🐼 🧏 🚳 🦻 <mark>外の検索</mark>					
Employees					
		l.			
🕀 🍰 HR デモアブリケー	□ 名前 □ By Date of Birth □ By Salary □ 〇 介 ■				
∃ 💕 Syste∎ i Ser ∃ 🔶 お気に入り	社員姓 法索 Employees 記述				
🗏 🚱 人事	A0070 BROWI	N VERON 🗸			
Employees		X			
Telephone S	🖉 🐇 Employees : 🗱 (A0070-BROWN VERONICA)				
Ⅲ 😡 プログラミング ァ Ⅲ 🔍 管理	□詳細 🛕 イヘツト 📄 ドキュメント				
資格情報の検索/保守					
	社員Na	ション			
	社員姓 BLAKE 社員名	v			
	住所1 3 Woodbury ス別ーンの	周査			
	住所2				
	自宅電話番号				
		^			
	まコード	N			
	送版日付(DDMMYY) GROUP ACCOUNTS				
	INFORMATION SERVIC     LEGAL DEPARTMENT	LS			
	取得日 资格	TIO			
	(DDMMYY) コード 資格記述				
< > OS /12 /10 ADWIN1 Administrate Part					
ご ご  ※カセージ 準備 AS/400 TS JPN V20PGMLIB 10/07/16 11:29					

4. ツリー・エントリーをダブルクリックします。選択した部門コード が5250画面のフィールドに即座に戻されます。5250画面にSECTIONと いうフィールドがあるため、これも更新されます。これは、サンプ ル・フィールド・ハンドラーDF\_PRM04に、フィールドSECTIONを処 理するコードが含まれているためです(このフォームのソース・コード を確認してください)。

- 5. ダブルクリックせずにツリーのアイテムを選択します。アイテムが 即座に更新され、5250フォームに戻ります。
- 6. 5250フォームの任意の場所をクリックして、DF\_PRM04を閉じま す。
- 7. フィールド・ハンドラーが表示されたら、"M"と入力します。ハン ドラーは最も一致する部門を推測しようとします。

注:DF\_PRM04は従来のF4プロンプタです。その動作はすべて自分で決定 します。このプロンプタ(およびその他のDF\_PRMnn付属サンプル)を使 用して、さまざまな種類のプロンプトを試用および変更できます。これ により、独自の特別なフィールド受け渡しコンポーネントを作成する方 法を理解できます。 まとめ

重要なポイント

- 特別なフィールド受け渡しは、フィールドの高度なプロンプト機能 です。
- プロンプトを表示するフィールドの名前、使用するファンクション・キー、プロンプタとして使用するVisual LANSAフォームを指定します。
- System iのプロンプトとは異なり、Visual LANSAのプロンプタ・ フォームでは、必ずしもSystem iサーバーとの対話処理が行われるわ けではありません。このため、処理速度が向上します。
- 特別なフィールド受け渡しを使用して、5250デバイス上では実現で きない機能を提供できます。
- 特別なフィールド受け渡しに関する詳細なチュートリアルについて は、「RAMP-TSAD03:特別なフィールド受け渡し」を参照してくだ さい。
- また、トピック「高度なプロンプト」も参照してください。

#### 覚えるべき事柄

 刷新された5250画面で特別なフィールド受け渡しをフィールドに関 連付ける方法

# RAMP-TS011: 付属のドキュメント・コマンド・ハンドラーの スナップイン

このチュートリアルでは、付属の汎用的なドキュメント・コマンド・ハ ンドラーをRAMPアプリケーションにスナップインする方法を学習しま す。

目的

- アプリケーションで付属の汎用的なコマンド・ハンドラーを使用する方法を学習する
- 刷新されたアプリケーションに価値を簡単に追加する方法を理解する

👹 Employees				
77(11(5) 经生(5) 未一八八 人	<u>*_****</u>			
このチュートリアルでは、付属の	2013月的な 🖉 🗖 🗖 🕼 🗊 🔿 🤺 外の検索			
ビジネス・オブジェクトにスナップ	インします。 By Date of Birth By Salary G A P	X		
このコマンド・ハンドラーを使用し ドキュメントをサーバーのデータ 保管します。ドキュメントを作成 リネーム、削除、開いて編集す	た、社員の タベースに は、コピー、 することが	記述 JONES BEN JACKSON		
できます。 Cesます。	Employee : ドキュウ、(A1001-JONES BEN)	×		
〒 @ プログラミング テウニック 〒 @ 管理				
	Name         Document         Status           Blue hills.jpg         28,521         On Server           Water lilies.jpg         83,794         On Server			
	Cleave Request For 31,232 SAVE PEND 保留中の変更を保存			
🖃 📰 🗃 タクセージ 準備 ASI002_TS JPN V20PGMLIB 10/07/14 1054 🥥				

この目的を達成するには、以下のステップを完了してください。

- RAMP-TS011 ステップ1. DX\_DOCSコマンド・ハンドラーをスナッ プインする
- RAMP-TS011 ステップ2. ドキュメントを追加する

- RAMP-TS011 ステップ3. ドキュメントを操作する
- まとめ

# 始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更
- RAMP-TS009:ナビゲーションの追跡
- RAMP-TS010:特別なフィールド・ハンドラーの使用

# RAMP-TS011 ステップ1. DX\_DOCSコマンド・ハンドラーをス ナップインする

このステップでは、付属のドキュメント・コマンド・ハンドラー DX\_DOCSをEmployeesビジネス・オブジェクトにスナップインします。

- 1. [フレームワーク]ウィンドウで、Employeesビジネス・オブジェクト をダブルクリックし、そのプロパティを表示します。
- 2. [使用可能コマンド]タブを表示します。
- 3. DX\_DOCSを[ドキュメント]コマンドのコマンド・ハンドラーとして 指定します。

使用可能	」 ~コマント・タイプを選択して下さい	
🔶 e-mail	🔵 ビジネス オフジェクト コマント 💿 インスタンス コマン	\h*
A*21	<u></u>	
h*+1xyh	順序 3	
œt°テ°オ		
三書業務	ליאלאליםליאליאלים	のサイズ
🚱 新規	常にアクティフ 省略値 👱 ロントゥ	
	省略値のコマンド いいえ マ パンドノ	
	Web 7'7	7 <b>7</b> °
	Web 0/12/13/261-41	
	▼ Windows の使用を許可 _任意引	50
	♥ ポップアップメニューの表示 文字引援	1:
	▼ ミニケールハニエに表示 又子も度	(2:
	他のコマントなフを隠す 数字引援	1:
	「「アウセス制版】 *** マス 1**	
		( 2:
	▶ 非表示コマントとして実行	
	-10F 70F 7-	
	> 7'+*-++++ DY DOCS	0
	0 424 47F 0A_0003	
	() サンプル - RAD-PAD RADPAD 5028A2	CAB0B14A7D97265CE43D22E3B4.HTM

- 4. Employeesビジネス・オブジェクトのプロパティを閉じます。
- インスタンス・リストから社員を選択し、[ドキュメント]タブを表示します。付属のドキュメント・コマンド・ハンドラーがスナップインされ、使用可能になっています。

👹 Employees				
ファイル(E) 編集(E) 表示(V)	ヘルプ(出) ウィントウ(₩) (フレームワーク) (管理)			
③作成 🗇 🖨 印刷	1 1 🖹 🖹 🔺 🔟 🗖 📓 🗊 🔿 🚳 🐼 🖓 🖉 🔧 🕅 🗡 外機素			
Employees				
E DEN HR	× × × × × × × × × × × × × × × × × × ×			
Ⅲ 2010 ERP Prototype 1 Ⅲ 2010 HR 疗*モアフ*リケーション	社員姓 検索 記述			
Human Relation	J A1001 JONES BEN			
	A1234 JACKSON STEPHEN			
🖃 💋 人本 Code Tables				
Employees	🐇 Employee : אלגב¥ (A1001-JONES BEN) 🗖			
Ⅲ 💿 プログラミング テクニック Ⅲ 🔍 管理	■ 詳細 ▲ イベット ■ ドキュメント			
	Documents for A1001			
	Name Document file s Status			
<	保留中の変更を保存			
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □				

# RAMP-TS011 ステップ2. ドキュメントを追加する

このステップでは、付属のドキュメント・コマンド・ハンドラーを使用 する方法を学習します。

1. Windows エクスプローラでドキュメント(WordドキュメントやPDFなど)をコピーします。

🔄 al		
ファイル(E) 編集(E) 表示(M) お気に入り(A)	ツール(1) ヘルプ(11)	
🔇 戻る • 🕥 · 🏂 🔎 検索 💫 7:	ォルダ 💷-	
アドレス(D) 🚞 C:¥a1		
フォルダ	× _ 名前 ▲	サイズ 種
<ul> <li>Projects</li> <li>Settings</li> <li>StartPages</li> <li>Templates</li> <li>マイ ピクチャ</li> <li>マイ ピクチャ</li> <li>マイ ミュージック</li> <li>個人用図形</li> <li>マイ コンピュータ</li> <li>ローカル ディスク (C:)</li> <li>450b5bbe920e07f8ae78f45658e6</li> <li>a1</li> <li>Dell</li> <li>Documents and Settings</li> <li>Inetpub</li> <li>LANSAPublishedWAM</li> <li>CANSAPublishedWAM</li> <li>SPIN0310</li> <li>WINDOWS</li> </ul>	Leave Request Form.doc	31 KB Mit (1) 編集(E) 新規作成(N) 印刷(P) プログラムから開く(H) 送る(N) ごる(N) つつうラムから開く(H) (1) ジョートカットの作成(S) 削除(D) 名前の変更(M) プロパティ(R)

- 2. フレームワークで[ドキュメント]タブを表示します。
- 3. ドキュメント・コマンド・ハンドラーの右側のエリアを右クリック し、コンテキスト・メニューから[貼り付け]を選択します。
| 👹 Employees                    |                                                   |
|--------------------------------|---------------------------------------------------|
| ファイル(E) 編集(E) 表示(V)            | ヘルプモH) ウィンドウ(₩) (フレームワーク) (管理)                    |
| ④作成 一 一 印刷                     | □ □ □ □ □ ◎ □ ◎ □ ◎ □ ◎ □ /////////////           |
| Employees                      |                                                   |
| ★ DEM HR     G ERP Prototype 1 | □ 名前 _ By Date of Birth _ By Salary               |
| 🗄 🌉 HR デモアブリケーション              | 社員姓 使索 Employee 記述                                |
| Human Kelation                 | J A1001 JONES BEN                                 |
| 🗉 🏠 お気に入り                      | A1234 JACKSON STEPHEN                             |
| 🖃 💋 人事                         |                                                   |
| Employees                      | 🤞 Employee : ドキュメント (A1001-JONES BEN) 🗖           |
| Ⅲ 💿 プログラミング テウニック Ⅲ ⑧ 管理       |                                                   |
|                                | Documents for A1001                               |
|                                | Name Document file s Status 表示业                   |
|                                | アイコンの整列Φ<br>最新の情報に更新(E)                           |
|                                | このフォルダのカスタマイズ(E)                                  |
|                                | 貼り付けの                                             |
|                                | ショートカットの貼り付け(ら)                                   |
|                                | 新規作成(₩) →                                         |
|                                | プロパティ( <u>R</u> )                                 |
|                                | 保留中の変更を保存                                         |
| < >                            |                                                   |
| 😑 📰 📔                          | メッセージ 準備 ASI002_TS JPN V20PGMLIB 10/07/14 10:08 🥥 |

ドキュメントがコマンド・ハンドラーに追加されます。

d Employees		
ファイル( <u>E</u> ) 編集( <u>E</u> ) 表示( <u>V</u> )	ヘルフ*(円) ウィントや(W) (フレームワーク) (管理)	
④作成 📛 🖨 印刷	范 🖻 🕥 🔺 🔟 🗖 🐺 🖾 🖉 🧶 🥸 🖉 📕 🥠 妙枝素	
Employees		
X	x	×
	二名前 By Date of Birth By Salary 〇 ② 企 合	
	社員姓 検索 Employee 記述	
• Wy Application	J A1001 JONES BEN A1234 JACKSON STEPHEN	
■ 合 お気に入り		
🖻 🚺 🔨 🖓 Code Tables		×
aligned by the set of the set o	🐇 Employee : אלא אלא (A1001-JONES BEN)	
■ 🔍 管理		
	Name Document file s Status	
	Leave Request Form.doc 31,232 SAVE PEND	
	Leave Reques	
	保留中の変更を保存	
< >		
	メッセージ 準備 ASJ002_TS JPN V20PGMLIB 10/07/14 10:15 🥥	

- 4. [保留中の変更を保存]ボタンをクリックし、ドキュメントをサー バー上の付属のデータベース・ファイルDXDOCSに保管します。
- 5. 写真やその他のイメージなど、別のファイルをコマンド・ハンド ラーに追加し、コマンド・ハンドラーを保管します。

ドラッグ・アンド・ドロップを使用してドキュメントを追加したり、コ ンテキスト・メニューを使用して新しいドキュメントを作成することが できます。

## RAMP-TS011 ステップ3. ドキュメントを操作する

このステップでは、ドキュメント・コマンド・ハンドラーでドキュメントを編集および削除する方法を学習します。

1. ドキュメントのリストからドキュメントを選択します。これは、最初にドロップした場所の右側のエリアに表示されます。

ファイル(E) 編集(E) 表示(M) ヘルプ(H) ウィンドウ(M) (フレームワーク)(管理)         ③ 作成       □ 毎 印刷       □ ▲ 例       □ ▲ 例       □ ▲ 例       □ ▲ 例       ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	
③作成 🗇 🛱 印刷 🗅 🖻 🕥 🔺 🗵 🗖 🕼 😳 🕢 🧐 🖂 🤹 🗟 🛛 <mark>外分検索 …</mark>	
Employees	
● Bell HR       ● Code Tables         ● Code Tables       ● Employees         ● Employees       ● Exit         ● Employees       ● Exit         ● Employees       ● Exit         ● Employees       ● Exit         ● Employee       ● Exit         ● Employee	
	,

- 2. ドキュメントをダブルクリックして開きます。
- 3. ドキュメントを閉じて、フレームワークを開きます(表示されていない場合)。
- ドキュメントを選択し、右クリックしてコンテキスト・メニューを 表示します。
- 5. [削除]を選択します。ドキュメント・コマンド・ハンドラーに削除 が保留されていることが示されます。

de Employees	
ファイル(E) 編集(E) 表示(⊻)	ヘルプ(H) ウィントウ(W) (フレームワーク) (管理)
④作成 📋 🖨 印刷	① 🗎 🕥 🔺 🔟 🗖 📕 🖾 🗇 🔕 🕸 🌚 🖉 🖊 好 沙検索
Employees	
E Se Den Hr	A前 By Date of Birth By Salary
⊕ 🍯 ERP Prototype 1 ਦ 🏅 HR デモアプリケーション	社員姓 使索 Employee 記述
🕀 🌠 Human Relation	J A1001 JONES BEN
田 英 application 田 会 お気に入り □ 会 トゥ	A1234 JACKSON STEPHEN
🖃 💓 🕂 🗭 🍰 Code Tables	
Employees	🤹 Employee:ドキュメント (A1001-JONES BEN)
Ⅲ 🐼 ノ゙ログフミノグ アワニゥク Ⅲ 🔍 管理	
	Documents for A1001
	Name Document file s Status
	K Leave Request Form.doc 31,232 DEL PEND
	保留中の変更を保存
< >	
🖻 📰 🗃	メッセージ 準備 ASI002_TS JPN V20PGMLIB 10/07/14 10:42 🥥

6. [保留中の変更を保存]ボタンをクリックし、ドキュメントを削除し ます。 まとめ

重要なポイント

- 付属のコマンド・ハンドラーを再利用することで、RAMPで刷新しようとしている任意の5250アプリケーションに大きな価値を付加できます。
- ドキュメント・コマンド・ハンドラーは、任意のビジネス・オブ ジェクトで使用できます。
- ドキュメント・コマンド・ハンドラーでは、ドキュメントを作成、 削除、リネーム、コピーしたり、開いて編集することができます。
- コピーと貼り付けおよびドラッグ・アンド・ドロップを使用して、
   ドキュメントを追加することができます。
- ドキュメント・コマンド・ハンドラーのリスト・エントリーの横に あるアイコンは、ドキュメントのステータスを示しています。

覚えるべき事柄

 付属の汎用ドキュメント・コマンド・ハンドラーDX\_DOCSの使用 方法

# RAMP-TS012: 付属のNotesコマンド・ハンドラーのスナップ イン

このチュートリアルでは、付属の汎用Notesコマンド・ハンドラーを使用 する方法を学習します。

目的

- 刷新されたアプリケーションにすばやく価値を付加する簡単なもう 1つの例を示す
- 汎用コマンド・ハンドラーの使用をその他のビジネス・オブジェクトに拡張する方法を考える



この目的を達成するには、以下のステップを完了してください。

- RAMP-TS012 ステップ1. DF\_T3201コマンド・ハンドラーをスナッ プインする
- RAMP-TS012 ステップ2. ノートを追加する
- まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更
- RAMP-TS009:ナビゲーションの追跡
- RAMP-TS010:特別なフィールド・ハンドラーの使用
- RAMP-TS011: 付属のドキュメント・コマンド・ハンドラーのス ナップイン

# RAMP-TS012 ステップ1. DF\_T3201コマンド・ハンドラーをス ナップインする

このステップでは、汎用のノート・コマンド・ハンドラーをEmployeesビ ジネス・オブジェクトの[イベント]コマンドに関連付けます。

- 1. Employeesビジネス・オブジェクトのプロパティを表示します。
- 2. [使用可能コマンド]タブで、[イベント]コマンドをコマンド・ハンド ラーDF\_T3201に関連付けます。
- 3. Employeesビジネス・オブジェクトのプロパティを閉じます。
- インスタンス・リストから社員を選択し、[イベント]タブを表示します。付属のノート・コマンド・ハンドラーがスナップインされ、使用可能になっています。

👹 Employees		
ファイル(E) 編集(E) 表示(V) ^	ヘルプ(世) ウィンドウ(W) (フレームワーク) (管理)	
③作成 🗇 🖨 印刷	- 19 🖹 🖄 🛕 🔟 🔲 🦉 🕲 🖗 🖗 🖉 🖌 🕅 外が検索・	
Employees		
× ■ DEM HR ERP Prototype 1 HR テ*モアフ*リウ-ション Human Relation W Application A つ Code Tables Employees M の フ*ロン*ラミンウ* テウニッウ ★ 管理	本前 By Date of Birth By Salary 社員姓 」 、 をmployee 記込 A1001 JON A1234 JACKSON STEPHEN) ● 詳細 イペント (A1234-JACKSON STEPHEN) ● 詳細 イペント トドキュエント Current o Category ? Note (Plain 1 1 Category priority Attachments	素 IES BEN IES BEN IES NON STEPHEN × (呆存(S) 作 F 6 友(N) 肖 / 序 6 友(N)
		11:11 🥥

### RAMP-TS012 ステップ2. ノートを追加する

このステップでは、社員のノートを追加する方法を学習します。また、 ドキュメントをノートに添付し、分類して優先順位を設定することもで きます。

- 1. [保存]および[作成]ボタンを使用して、社員のいくつかのノートを追加します。
- 前のチュートリアルと同様に、切り取りと貼り付けまたはドラッグ・アンド・ドロップを使用して、添付ファイルをノートに追加します。
- 3. ノートのカテゴリを作成し、その優先順位を設定します。例:

👹 Employees		
ファイル(Ё) 編集(Ё) 表示(⊻)	) ヘルプ(世) ウィントウ(逊) (フレームワーク) (管理)	
③作成 🗇 🖨 印刷	范 🖹 🛇 🔺 🔟 🗖 🐺 🖾 🔿 🧔 🖾 🖓 🖾 🖓 😡   😽 🖓	
Employees		
x DEM HR ERP Prototype 1 ERP Prototype 1 HR デ <sup>*</sup> モアフ <sup>*</sup> リケーション Human Relation My Application 会 お気に入り 人本	名前     By Date of Birth     By Salary       社員姓     検索       J     Altion       A1001     JONES BEN       A1234     JACKSON STEPHEN	×
Code Tables	🤞 Employee : イベント (A1001-JONES BEN)	
■ 🤹 管理	単細 A A A A A A A A A A A A A A A A A A	保存(5) 作成(N) 商IIR会
< >	Category TRAVEL Priority Category TRAVEL Prior	17 JP 48
E E 5	メッセージ 準備 ASJ002_TS JPN V20PGMLIB 10/07/14 1	1:30 🥥

まとめ

重要なポイント

- 付属のコマンド・ハンドラーを再利用することで、RAMPで刷新しようとしている任意の5250アプリケーションに大きな価値を付加できます。
- ノート・コマンド・ハンドラーは、任意のビジネス・オブジェクト で使用できます。
- このコマンド・ハンドラーを使用することで、エンド・ユーザーは ビジネス・オブジェクトのインスタンスに複数のノートを入力できま す。
- ノートは、データベース・ファイルFPNOTEに文字列フィールドとして保管されます。
- 添付ドキュメントは、データベース・ファイルFPDOCに保管されます。
- 添付ドキュメントをノートに追加することができます。
- ノートのカテゴリ値を指定して、独自の基準に従ってノートをソートできるようにすることができます。
- 作成/更新ユーザーと、作成/更新日時がリストに自動的に記録され ます。

#### 覚えるべき事柄

- このチュートリアルで示している考え方は、他のビジネス・オブ ジェクトに簡単に拡張することができます。
- 例えば、製品ビジネス・オブジェクトには、プレス・リリース、パンフレット、価格リストが関連付けられていることがあります。また、販売開始、顧客の苦情、リコール、耐用年数などの多くのイベントが関連付けられていることがあり、それぞれに多数のドキュメントが関連付けられている場合があります。
- また、スプール・ファイル用の付属の汎用コマンド・ハンドラー、 DF\_T3101もあります。

# RAMP-TS013:インスタンス・リストのデータをExcelに送信 する

このチュートリアルでは、アプリケーションをMicrosoft Excelと統合する 方法を示します。

目的

 Microsoft Excelにデータを送信するコマンド・ハンドラーを作成す る方法を学習する



この目的を達成するには、以下のステップを完了してください。

- RAMP-TS013 ステップ1. コマンド・ハンドラーを作成する
- RAMP-TS013 ステップ2. コマンド・ハンドラーをスナップインおよ びテストする
- まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更
- RAMP-TS007:ナビゲーションの追跡
- RAMP-TS010:特別なフィールド・ハンドラーの使用
- RAMP-TS011:付属のドキュメント・コマンド・ハンドラーのス ナップイン
- RAMP-TS012: 付属のNotesコマンド・ハンドラーのスナップイン

RAMP-TS013 ステップ1. コマンド・ハンドラーを作成する

このステップでは、コード ウィザードを使って、Microsoft Excelへの データの送信に使用できるSpreadsheetsコマンド・ハンドラーを作成しま す。

- 1. [フレームワーク]ウィンドウで、Employeesビジネス・オブジェクト のプロパティを表示します。
- 2. [使用可能コマンド]タブを表示します。
- 3. [スプレッドシート]コマンドを有効にします。



- 4. 従業員のプロパティを閉じます。
- 5. [フレームワーク]メニューからコード ウィザードを起動します。
- 6. Employeesビジネス・オブジェクトおよびSpreadsheetsコマンド・ハンドラーを選択します。
- 7. プラットフォームとして[標準MS Windows]を選択します。
- 8. 生成するコードの種類として、[CSVファイルでMS-Excelへデータを送信] を選択します。
- 9. [次へ]をクリックします。
- 10. 物理ファイルとして[PSLMST]を選択し、デフォルトのビジュアル 識別子およびプログラム的識別子を受け入れます。

💷 コート* ウィザ*ート*		
コード生成するオフジェクトを選択して下さい	CSVファイルでMS-Excelへデータを送信	
<ul> <li></li></ul>	ビッネス・オフジェクトで使用することを決定した識別/フロトルを指定します ビッネス・オフジェクトで使用することを決定した識別/フロトルを指定してください。 別/フロトコルを推論することを読みます。 このビジネス オフジェクトに一番近い物理 ファイル: ビジュアル識別子 (VisualID1 と VisualID2 の値を生成するため)	。物理ファイルがこの アシスタントは自動的に基本的 -
	7/~Illi*名 Type 記述	査 選択分の削除
J/I/9>By Date of Birth	1 EMPNO ALPHA 社員Na.	今一て著山民会
ALL コマント バント・ラー->フレームワークにつして	2 SURNAME ALPHA 社員姓	王、仁州、伊尔
(3) コマント* バント*ラー->終了	3 GIVENAME ALPHA 社員名	
🐣 TZ'/ド Ji'/ドラ>Assistant Example 1 💆	4	-
再表示	5	×
一生成したいプラットフォームを選択 ● MS Windows (Visual LANSA の使用) ● Web (LANSA for the Web の使用)	この物理ファイルのフィールトを追加 キーの追加	全て追加
○ Web (WAM コンポーネントの使用)	- / U/ 74用廠方丁 (Akey1,2,3,4,5 C Nkey1,2,3,4,5 C)1世企主が、9 つ/この	<i>"</i>
○ Web (AJAX スタイル コンポーネントの使用)	フィーい <sup>*</sup> 名 Type 記述	・ 選択分の削除
生成するコードの種類を選択して下さい	1 EMPNO ALPHA 社員No.	全て削除
CRUD コマント*バント*ラ		
リストを保守するコマント・ハント・ラ	3	<b>v</b>
新編のレントルトコの10月34) CSV77化でMS-Excelへデータを送信 UExecute ルーチンのインスタンス (シスの一部分) がまたのとつとは、excente ルーチン(トコの一部分)	この物理ファイルのフィールトを追加 キーの追加	全て追加
*8510/1/A3/A UEXecute (-7) (/~A0)-ap7)	<	>
	《 戻る 次へ	

- 11. [次へ]をクリックします。
- 12. コマンド・ハンドラーの上部にPSLMSTファイルのすべてのフィー ルドを含めるように選択します。

■ コート <sup>*</sup> ウィザ <sup>*</sup> ート <sup>*</sup>				
コート生成するオフジェ外を選択して下さい		CSV77	イルでMS-Excelへデータを送信	
<ul> <li>□ よう ビ・ジ・ネス オフ・ジ・エクトーンEmployees</li> <li>□ コマント・ハント・ラーーン詳細</li> <li>□ コマント・ハント・ラーン新規</li> <li>□ コマント・ハント・ラーント・オコメント</li> </ul>	こンドューザーIこ選択 Dフィールドから、スフ はいでください。 マンド、ハンドラーの上	を許可する金 心外シートにう 部に表示す・	全てのフィールトを先頭の領域に指定 送りたいフィールトを選択することが るフィールト	記してください。エント <sup>・</sup> ユーサ できます。リスト領域には
A コイント ハント フー・ノイト ノト コーマント ハント・ラー->スフ・レット・シート	7/-11/名	Туре	記述	<ul> <li>選択分の消</li> </ul>
7/1/2>名前	1 EMPNO	ALPHA	社員Na	全て買服
コノルターー>By Date of Birth	2 SURNAME	ALPHA	社員姓	± CHUM
ノイルバー・フタン Salary	3 GIVENAME	ALPHA	社員名	
(3) コマント・バント・ラーン終了	4 ADDRESS1	ALPHA	住所1	
🔼 👗 אַיזע אוידר 🕹 🕹 🕹 🕹 איזע אוידר 🕹	5 ADDRESS2	ALPHA	住所2	
再表示	6ADDRESS3	LALPHA	11∓Ph:3	
一生成したしいプラットフォームを選択 ● MS Windows (Visual LANSA の使用) ● Web (LANSA for the Web の使用) ● Web (WAM コンボーンントの(使用)	の物理ファイルのフィ ます マンドハンドラーの下	ールトを追加 部のリストに書	PSLMST キーの追加 表示するフィールド	全て追加
○ Web (WHM コンホーキン(シルスカ) ○ Web (A.IAX 7な/ル コンポーネンルの(使用)	フィールド名	Туре	記述	選択分の肖
生成するったの経験を選択して下すい	1			今て高山居住
	2			± CHUR
リストを保守するコマント・ハント・ラ	3			
新規のコマント・ハント・ラのアウトライン				
uExecute ルーチンのイルなりス(ソースの一部分)				
複数のインスタンス uExecute ルーチン(ソースの一部分)				Ň
			《戻る コード生態	成 キャンセル

13. [コード生成]をクリックします。

次のページ、[生成されたソース コード]にコマンド・ハンドラーのソース・ コードが表示されます。ここで、このコードを含むコンポーネントを作 成する必要があります。

- 14. コマンド・ハンドラーの名前として、iiiRMP02を指定し、説明とし てSpreadsheetコマンド・ハンドラーを指定します(iiiは自分のイニシャ ルです)。
- 15. [生成]ボタンをクリックし、コンポーネントを作成します。

しばらくして、コマンド・ハンドラー・コンポーネントがVisual LANSA エディターに表示されます。

16. Visual LANSAエディターでコンポーネントをコンパイルします。

# RAMP-TS013 ステップ2. コマンド・ハンドラーをスナップインおよびテストする

このステップでは、Spreadsheetsコマンド・ハンドラーをフレームワーク にスナップインしてテストします。

- 1. [フレームワーク]ウィンドウで、Employeesビジネス・オブジェクト のプロパティを表示します。
- 2. [使用可能コマンド]タブを表示します。
- 3. Spreadsheetsコマンド・ハンドラーを選択します。
- [ビジネス オブジェクト コマンド]オプションを選択します(コマンド・ハンドラーは1人の社員インスタンスではなくすべての社員に使用できます)。
- 5. [他のコマンドタブを隠す]オプションを選択し、[作成]のコマンド・ タブがウィンドウに表示されないことを確認します。
- 6. Windowsコンポーネントとして、コマンド・ハンドラーの名前 (iiiRMP02)を指定します。

💰 ድንቶス ተፓንድታትወን ገለትንት – Employees										
識別子 アイコン ビジュアル スタイル フィルター フィルター設定 使用可能コマンド コマンドの表示 ユーザー フロハディ サフタイフ インスタンス リストク ላ 🕨										
使用可/不可の設定を切り替えるにはリストから ドラッグアントドロップします…										
選択可能コマント <sup>S</sup> AJAXEX1 AJAXEX2 AJAXEX2 AJAXEX2 AJAXEX3 Assistant Assistant Assistant Assistant CRUD e-mail Example 1 Example 2 Example 2 Example 3 Example 5 Example 5 Example 6 Example 7 Example 8 Fax Filter Pane MS Wordを URL アジスタント T	コマンド タイクを選択して下さい ● ビジネス オフジェクト コマンド       インスタンス コマンド         順序       1         コマンド オフション 常価 高さ クインドウ ツ Web の使用を許可 マ Windows の使用を許可 マ ボッフアッフジェューの表示 マ だッールバー上に表示 マ 他のコマント ないたうー ククレ やの ビー 生意引数 2: サウレス制限 ● 非表示コマント どして実行       ウインドウのサイズ・ 幅 高さ クインドウ Web フラウザ Web フラウザ Web フラウザ Web フラウザ Web フラウザ Web フラウザ Web フラウザ Web フラクザ Web フラクザ Web フラクサ Web フラク Web フタ Web フラク Web フタ Web									
	閉じる									

- 7. Employeesビジネス・オブジェクトのプロパティを閉じます。
- 8. フレームワークを保存して再起動します。
- 9. Employeesビジネス・オブジェクトのフィルターを使用して、社員を 選択します。
- 10. インスタンス・リストの一部の社員を選択します(Ctrlキーを押しな がら、マウスでクリックします)。
- 11. ナビゲーション・ペインで[Employees]を右クリックするか、インス タンス・リストの社員を右クリックし、コンテキスト・メニューから [スプレッドシート]を選択します。



12. コマンド・ハンドラーで、Excelに送信するフィールドを選択し、 [Just the selected instance list entries]オプションを選択します。

🤞 Employee : スプレットシート		
<ul> <li>?</li> <li>값社員№</li> <li>값社員姓</li> <li>값社員名</li> <li>《住所1</li> <li>《住所3</li> <li>郵便番号</li> <li>《自宅電話番号</li> <li>》部門電話番号</li> <li>》部門電話番号</li> <li>》部門□ード</li> <li>『課□ード</li> <li>? 課□ード</li> <li>? 社目付(DDMMYY)</li> <li>『退職日付(DDMMYY)</li> <li>》 月給</li> </ul>	All Instance List entries     Just the selected instance list entries	MS-Excel(2出力
	H:447 W:551 メッセーシ	レコート・サイズ

13. [MS-Excelに出力]ボタンをクリックします。

選択した社員のデータを含むMS Excelスプレッドシートが表示されま

0.	EOMRMP02_20100714133842.csv - Microsoft Excel _ = X									
	ホーム	挿入	ページ レイアウト	<u> </u>	データ 木	交閲 表示	チーム		0 -	= x
-	<u> </u>	MS Pゴシック	- 11 -	= =	こう 標準	-	▲ 計挿入・	Σ -	A7 #	
8tn		BI <u>U</u>	A A		·	% , 7/	学前除 -	💽 * 👌	┃┃ ■■	et-
100	<b>3</b>	- 🗞 - 🛓	<b>A</b> - <u>7</u> - <u>#</u> -		<b>◆.0</b>	.00	• た書聞 •	2-	7イルター 選択	
クリップ	9ボード 喧	フォント	G.	配置	- 数	値回	セル		編集	
	A1	•	(• <i>f</i> <sub>x</sub>	社員No.						¥
	A	В	С	D	Е	F	G	Н	Ι	
1	社員No.	社員姓	社員名	住所1	住所2	自宅電話番	部門電話番部	"]コード	課コード	社員
2	A0070	BROWN	VERONICA	12 Railway	Baulkham I	(02) 9609	(02) 9647 :INF		DV	5C ≡
3	A0090	BLOGGS	FRED JOH	70 MAIN S	NEWTOWN	344-22344	654 6475 ) FL1	Г	3	20
4	A1 031	BLAKE	JOHN	3 Woodburg	Winston Hi	(02) 9668	(02) 9922 !MIS	3	E	60
5	A3564	BROWN	FREDDY	121 SMITH	Newtown	(02) 567-6	(02) 456-5 AD	М	4	30
6										
7										
8										
14 4	EON	IRMP02_201.00	0714133842	1				_		
עדב	K ScrollLo	ock					100%	( 🕞 🗌		

Excelで社員データを操作できるようになりました。

まとめ

重要なポイント

- 真の価値が5250アプリケーションにまったく付加されないドロッ プ・ダウンなどとは異なり、アプリケーションをExcelなどのデスク トップ・ツールと統合すると、新しい情報や機能がアプリケーション のユーザーに提供されるため、5250アプリケーションに真のビジネス 価値が付加されます。
- この機能を使用すると、MS-Excelの機能を容易に活用できます。 ユーザーはMS-Excelで情報を取得した後、その情報を使用して、グラ フやチャートの描画、ピボット・テーブルの作成、ハード・ドライブ への保存、印刷、電子メールによる他のユーザーへの送信など、必要 な操作を実行できます。MS-Excelは、IBM iデータベースに保持され ている価値の高い情報を使用するためのゲートウェイを提供します。 また、MS-Excelは一般的なレポート・アクティビティの優れた出発点 でもあります。
- スプレッドシート・データの抽出は、エンド・ユーザーが独自の ツールを使用して実装するのではなく、ソフトウェア開発者によって 実装されます。したがって、より安全であり、そのパフォーマンスは 確認されています。

覚えるべき事柄

 インスタンス・リストのすべてのエントリーまたは選択したエント リーのデータをMS Excelに送信するコマンド・ハンドラーを作成する 方法

# RAMP-TS014:HRデモ・アプリケーションへのRAMP画面の スナップイン

このチュートリアルでは、新規に作成したRAMP画面をHRデモ・アプリ ケーションに追加します。

目的

- RAMP画面とフレームワーク・コンポーネントをアプリケーション に組み込む方法を学習する
- インスタンス・リストとRAMPコマンド・ハンドラーが相互に作用
  する方法について詳しく理解する

Se 17	-7										
77110	F		76-47-5)	(管理)	)						
0	このチュートリアルで	は、RAMP画面を		🔅 🖂 💈	<b>8</b> (1)	16 I		ケイック検	索		
Enter	HRデモ・アプリケーション	にスナップインして、	or teleph		her or s	elect a s	skill tyne	then or	229		_
enter	刷新された5250画面を	フレームワーク・			ber, or a		skin type,	, anon pr		0	_
	コンボーネントと簡単に	一致または組み									X
- 62	合わせることができるこ	とを確認します。	🔺 🔟 🖡		<b>N</b>						
			□-ト*/ ID	住所1	住所2	住所3	郵便	部門	自宅	部	課 🔺
	お気に入り	ONICA	A0070	12 Ra	Baulk	NSW	2153	(02)	(02)	I	DV 📄
1	ハー フ・ロク・ラミンク・ テクニック	010	A0090	70 M	NEW	AUST	2220	654 6	344-2	F	03
H 🔍	管理	SMIT	A1001	29 Ar	DEE	NSW/	2001	406.6	977 6	A	02
		WOOD FY	A1015	59 Da	BEXIEV.	NSW.	2000	789 4	450 1	A	01
					11	1					>
											×
		🍓 リソース : 👌 ai	Is RAMP-	ts (Jon	IES, BEN-	-A1001)					
			ails RAMP-T	5 🛕 11	NYN 🔟	化小	10 外ムシー	k 📄 ka	f=x?/h   1	ート	
		```		j	資格情報	前の検索	/保守		保存		XIa-
		社員Na				: /	A1001		終了		1/4-1
		社員姓					JONES		-		
		社員名		• • • •	• • • •		BEN		PROMPT		rı
		任所1		• • • •	• • • •		144 Frog	9	取消し		スクリ・
		住所3					NSW	_	MSG		
		自宅電話番号					799 5268	8	1100		
		部門コード					ADM +		変更		
•	] 📑 Terminal error: 機省 🕶	~ ×>+	-ジ 準	備 A	S/400 TS	JPN	V20PGML	IB 10/0	7/15 17:0	7 🥥	

この目的を達成するには、以下のステップを完了してください。

- RAMP-TS014 ステップ1. HRデモ・アプリケーションへのRAMP画
   面のスナップイン
- RAMP-TS014 ステップ2. SETVALUEステートメントを変更する

• まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更
- RAMP-TS009: ナビゲーションの追跡
- RAMP-TS010:特別なフィールド・ハンドラーの使用
- RAMP-TS011: 付属のドキュメント・コマンド・ハンドラーのス ナップイン
- RAMP-TS012: 付属のNotesコマンド・ハンドラーのスナップイン
- RAMP-TS013: インスタンス・リストのデータをExcelに送信する

# RAMP-TS014 ステップ1. HRデモ・アプリケーションへの RAMP画面のスナップイン

このステップでは、NewEmployee画面とDisplayEmployee画面をHRデモ・ アプリケーションにスナップインします。

- フレームワークで、HRデモ・アプリケーション、リソース・ビジネ ス・オブジェクトの順に選択します。
- 2. リソース・ビジネス・オブジェクトのプロパティを表示します。
- 3. [使用可能コマンド]タブを表示します。
- [コマンドの定義]ボタンをクリックし、新しいコマンドを追加します。

ጋ ከ セス: VF_P	R003	
	-	
'礼(省略値)	-	
×12-		コマンドの定義
		閉じる

- 5. [コマンド]ウィンドウで、[新規作成]をクリックし、新しいコマンド を作成します。
- 6. コマンドのキャプションを[RAMP-TS詳細]にします。

אלאב 🗖					
1721	顺序	^	識別子 ツールバーとメニュー ビットマ	ヮプとアイコン その他のオブション 使用方法	
Details R	1		4+07 5/12	Details RAMP-TS	(JPN)
e-mail	1		L	Decarts NAME-15	
Example 1	1		ビント:		(JPN)
Example 2	1		顺序	1	
Example a	-				
Example 5	1		内部識別子:	0C2F8EA045784A13825549D8D7C9BADC	
Sexample 6	i		コーザー オコミノーカト カイコッ		
Example 7	1		1-9-11919	0C2F8EA045784A13825549D8D7C9BADC	名前の確認
SExample 8	1		キャフションとアクセス キー:	Details RAMP-TS	(JPN)
ዀ Fax	1		Surphthale	New	
🕍 Filter Pane	1		24-1071.	None	
🌷 HTMLを使用	1		図 Web の使用秀許可		
List Pane	!				
MS Wordを			Windows の使用を許可		
A RUDAUL		Y	最終変更	20100714-143713-V20PGMLIB	
削除	新規作成				
					閉じる

- 7. [コマンド]ウィンドウを閉じます。
- 8. [使用可能コマンド]タブに戻り、[RAMP-TS詳細]コマンドを[使用可能]リストにドラッグします。



- 9. [作成]コマンドを[使用可能]リストにドラッグします。
- 10. フレームワークを保存して再起動します。
- 11. RAMPツールを起動します。5250識別セッションを起動する必要は

ありません。

- 12. [RAMPツール]ウィンドウの画面とスクリプトのリストで NewEmployeeを選択します。
- 13. [関連するコマンドハンドラ-]で、画面をリソース・ビジネス・オブジェ クトの[作成]コマンドに関連付けます。

-関連するコマント゛ハント・ラー			
コマント* (タフ*)へのリンク	わ゙ジェクト内	2-ザー オブジェウト タイプ	~
■帳票	課	DEM_ORG_SEC	
🔲 地域	課	DEM_ORG_SEC	
■ 詳細	<u> </u>	DEM_ORG_SEC_EMP	
一 た	リソース	DEM_ORG_SEC_EMP	
「へごント」	リソース	DEM_ORG_SEC_EMP	
E h * t = X > h	リソース	DEM_ORG_SEC_EMP	
🔲 9789-F	<u> </u>	DEM_ORG_SEC_EMP	
イメージ	<u> </u>	DEM_ORG_SEC_EMP	
■ 帳票	リソース	DEM_ORG_SEC_EMP	
🔲 Details RAMP-TS	リソース	DEM_ORG_SEC_EMP	-
☑作成	<u> </u>	DEM_ORG_SEC_EMP	
	ユーザーと権限	VF_USER_OBJECT	
■権限	ユーザーと権限	VF USER OBJECT	<b>Y</b>
	再表示		

- 14. 次に、DisplayEmployee画面をリソース・ビジネス・オブジェクトの[RAMP-TS詳細]コマンドに関連付けます。
- 15. [保存]を実行します。
- 16. [フレームワーク]ウィンドウで、HRデモ・アプリケーションとリ ソース・ビジネス・オブジェクトを選択します。
- 17. 右クリックし、コンテキスト・メニューから[作成]を選択して NewEmployee画面を表示します。



- 18. NewEmployee画面を閉じます。
- 19. フィルターを使用し、インスタンス・リストに社員を表示します。
- 20. 社員を選択し、[RAMP-TS詳細]コマンド・ハンドラーをクリックし ます。「DisplayEmployeeに移動できません」というようなエラーが表 示されます。

🤹 J)/-x										
ファイル(E) 編集(E) 表示(V) ヘル	·プ(H) ウィントウ(W)	(76-47-	-ク) (管	理)						
🔇 作成  🚔 印刷	â 🖻 🖻 🔺 🖸		<b>G</b> 🔟 (	ି   💿	🖂 😵	<b>感</b> []			外纳续索	
Enter an employee number, a	full or partial nam	ne or tele	ephone n	umber, o	r select	a skill ty	pe, then	press	b	2
X										×
⊕	🕁 😳 🗖 🚞	1		1						
🖃 🏅 HR デモアブリケーション	名前	]-ト*/ID	住所1	住所2	住所3	郵便	部門	自宅	部 課	^
Se 197-7	BROWN, VERONICA	A0070	12 Ra	Baulk	NSW	2153	(02)	(02)	I DV	13
Human Relation	BLOGGS, FRED JO	A0090	70 M	NEW	AUST	2220	654 6	344-2	F 03	
By Application	JONES,BEN	A1001	144 F	PYMB	NSW.	2001	798 0	799 5	A 01	_
🗉 😭 お気に入り	WOODS READLEY	A1003	29 Ar	BEXIEV	NSW.	2000	406 6 789 4	450 1	F 02	
🗄 🌠 大事	ROBINSON, MARY	A1025	14 W	ST IV	NSW.	2005	456 1	126 3	A., 01	~
■ ④ プロクラミンク プリーック										X
а 🗝 в-т	💁 リソース : Detail	s RAMP-	TS (JO)	NES, BEN	A1001)	(				
		als RAMP-T	SA	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	14-22	The ballion		+. ruh 1	S ~ k	
			- <u>A</u> 1		1.4 7	- /IA/	r <u>e</u> r	11/01	N T	
			1	Unable to r	avigate to	form FindE	mployee			
								_		
		現	在の5250	フォームを表	示 (この本)	いはデザイン	モートだけて	行使用可能	E)	
• • •			メッセージ	準備	ASIO	02_TS   :	IPN V2	OPGMLIB	10/07/14	15:24 🥥

# 21. [現在の52507ォームを表示(このボタンはデザインモードだけで使用可能)]ボ タンをクリックし、ナビゲーションが停止した画面を表示します。

102 7 . D.4.3. DAMD TO ( IONEC DEN A1001)	
🨹 アノース: Details RAMP-TS (JONES, BEN-ATOUT)	📄 FifaxyF   🕥 7-F
	保存
社員No ADM +	PROMPT
	メニューバーを表示
	スナッフペショットを表示
	トドキュメンテーション
	トレース オン
	スタリーンの調査

それは、FindEmployee画面です。[従業員番号]フィールドの値として部 門コードが使用されています。次のステップでは、スクリプトを変更し てインスタンス・リストから従業員番号を取得します。

# RAMP-TS014 ステップ2. SETVALUEステートメントを変更する

このステップでは、HRデモ・アプリケーションのフィルターを検証し、 FindEmployee画面に関連付けられているスクリプトを変更して、フレー ムワーク内の複数の場所で使用できるようにします。

- 1. リソース・ビジネス・オブジェクトのプロパティを表示します。
- [フィルター]タブ、[フィルター登録の設定]タブの順に選択します。
   フィルターはDF\_FILT9です。このフィルターは、インスタンス・リストに入力するために使用されるAkey値を決定します。

, フィルター フィルター設定 使用	可能コマンドーコ	マンドの表示	ユーザー プロパラ	fr #73917°	インスタンスリス
識別子 アイコン フィルター登録	の設定				
常にアクティブ 省略値	2				
-フィルター ハンドラー					
<ul> <li>● コンホ<sup>®</sup>ーネント</li> </ul>	DF_FILT9			۹.	
○ サンフ% - RAD-PAD	_RADPAD_	84258E559F	CD47C5BFCB7	B13D9CE889	З.НТМ

- 3. リソースのプロパティを閉じます。
- 4. フレームワークを閉じます。
- 5. Visual LANSAエディターに切り替えて、再利用可能なパーツ DF\_FILT9を見つけて開きます。
- 6. フィルター・ソースでAddtoListメソッドを検索します。

Invoke Method(#avListManager.AddtoList) BusinessObjectType(DEM\_ORG\_SEC\_EMP) Visualid1(#FULLNAME)
Visualid2(#EmpNo) Akey1(#deptment) Akey2(#section) Akey3(#Empno) NColumn1(#PostCode)
AColumn1(#ADDRESS1) AColumn2(#Address2) AColumn3(#Address3) AColumn4(#PhoneBus)
AColumn5(#PhoneHme) AColumn6(#Deptment) AColumn7(#Section) NColumn2(#AdditionTotal)
SetasCurrent(#AdditionOption) ExecuteDefaultCmd(#AdditionOption)

7. メソッドのAkey値を検証します。*Empno*フィールドがAkey3である ことがわかります。

FindEmployee画面に関連付けられているスクリプトは、Akey1値を使用 してインスタンス・リストから従業員番号を取得することに注意してく ださい(これは、[名前]フィルターで使用されるAkey値であるためです)。

- 8. DF\_FILT9を閉じます。
- 9. フレームワークを起動します。
- 10. RAMPツールを起動します。
- 11. 画面とスクリプトのリストでFindEmployee画面に関連付けられているスクリプトを見つけます。
- 11. SETVALUEステートメントを以下のように変更します。

var wBusinessObject = objBusinessObject.uUserObjectType; if (wBusinessObject == 'EMPLOYEES') SETVALUE("txtEmpno",objListManager.AKey1[0]); if (wBusinessObject == 'DEM\_ORG\_SEC\_EMP') SETVALUE("txtEmpno",objListManager.AKey3[0]);

このステートメントは、画面を呼び出しているビジネス・オブジェクト の名前に従ってAKey値を設定します。

- 12. 変更をコミットし、[保存]を実行します。
- 13. フレームワークを表示します。
- 14. HRデモ・アプリケーションで[リソース]を選択し、インスタンス・ リストに入力します。
- 15. [RAMP-TS詳細]コマンド・ハンドラーを表示します。

🧶 J)/-x		
ファイル(E) 編集(E) 表示(V) ヘルプ(	d) ウィントウ(₩) (フレームワーク) (管理)	
📀 作成 📔 💼 🖻 🔟 🛕	🔊 🗖 🖉 🖽 🔿 🔞 🐼 🖓 🐼	りイック検索
Enter an employee number, a fu enter :	l or partial name or telephone number, or select a skill type, then press	s 🔽 🔽
KR テ*モアフ*リケーション     MR テ*モアフ*リケーション     My-ス	🛨 😳 🗆 🚔 🔺 🔟 🍋 🖻 📎	×
<ul> <li></li></ul>	名前         コード / ID         住所1         住所2         住所3         郵便…         部門…         自           BLOGGS,FRED JO…         A0090         70 M…         NEW…         AUST…         2220         654 6…         3 <sup>o</sup>	自宅 部. 課 A
<ul> <li>▲ 人争</li> <li>★ 100 7*ログ*ラミング* テクニック</li> <li>★ 管理</li> </ul>	SMITHSON,FRED         A0193         121 C         Windsor         NSW         2034         (02)         (0           MISS SIMPSON,A         A0907         33 an         anne         annes         2145         090909         0'           JONES,BEN         A1001         144 F         PYMB         NSW.         2001         798 0         79	02) A 05 90909 A 03 995 A 01
	SMYTHE,JOHN         A1002         20 Co         WER         NSW.         2100         798 4         0-           SMITHE,Robert         A1003         29 Ar         DEE         NSW.         2000         406 6         9	47 6 A 02 77 6 F 02
	🍓 リソース:Details RAMP-TS (SMITHSON,FRED-A0193)	
		x>h   🕥 /-h
	資格情報の検索/保守	保存
	社員№	終了
	社員班 · · · · · · · · · · · · · · · · · · ·	PROMPT
	住所1 121 Cutler Ave 住所2 Windsor	取消し
	住所3 NSW	MSG
	日七电話 番坊 · · · · · · · · · · · · · · · · · ·	変更
		メニューバーを表示
	入社日内(DDMMYY)	トッキュメンテーション
		トレース オン
	取得日 資格	スツーンの調査
😇 📰 🔳 Terminal error: 機能牛一 :	は使用 ▼ ▲ ダッセージ 準備 AS/400 TS JPN V20PGMLI	IB 10/07/15 16:58 🥥

- 以上で、HRデモ・アプリケーションは、刷新された5250画面とフレーム ワーク・コンポーネントの組み合わせで構成されています。
- 16. Employeesビジネス・オブジェクトの[詳細]コマンド・タブが引き 続き機能していることも確認します。

### まとめ

#### 重要なポイント

- RAMPで処理した5250画面をVisual LANSAで作成したコマンド・ハンドラーと容易に組み合わせることができます。
- このフィルターは、画面をインスタンス・リストにリンクするため にどのAKey値を使用する必要があるかを決定します。

## 覚えるべき事柄

- RAMP画面とVisual LANSAのコマンド・ハンドラーを組み合わせる 方法
- AKey値を確定する方法

RAMP-TS015: 画面バリエーションの概要と処理

このチュートリアルでは、画面署名と画面識別の背後にある概念につい て学習します。

目的

- 画面署名について理解する
- 異なる署名を使用して、2つの画面の1つの画面定義を作成する方法 を学習する
- 他の画面と署名を共有している画面を一意に識別する方法を学習する
- 画面名のバリエーションを使用する方法を学習する

Keymaps AutoGUI	Screens	K Session Display Help	🔎 - Auto - 💌 🔍 🧔
一致の候補	Z	社員管理メイン・メニュー	
1D 名前 14 PSLSVSMain <新しい定義> 最有力候補: PSLSVSMain 画面の定義 保存 キャンセル 削除 裏名: A.D@D00198.SFL004C 女A.D@D00198.SFL004C	X17       0       (14)       ≹     自動定義       R.SFL004SF.SFLF00T       RSFL004SF.SFLF00TRSFLM	<ul> <li>母号を進択する</li> <li>1. 社員賃</li> <li>2. 社員賃</li> <li>適合賃</li> <li>通合賃</li> <li>4. 社員賃</li> <li>5. 社員貸</li> <li>7. 創計電話書サ</li> <li>9 ~ 7</li> </ul>	てください.
PSLSYSMain 別名:	(ID = 14)		
】 記述:		F1= ヘルブ F3= 終了 F12= 取消し F14=MSG	

この目的を達成するには、以下のステップを完了してください。

- 5250画面とは?
- RAMP-TS015 ステップ1. 同じ名前を2つの画面バリエーションに割 り当てる
- RAMP-TS015 ステップ2. 同じ署名を持つ異なる画面を処理する
- RAMP-TS015 ステップ3. 画面バリエーションを作成する
- RAMP-TS015 ステップ4. スクリプトで画面バリエーションを使用する
- RAMP-TS015 ステップ5. 一連の画面の作成(高度)

#### • まとめ

始める前に

このチュートリアルを完了するには、以下のチュートリアルを完了して いる必要があります。

- RAMP-TS001:刷新されたアプリケーションの基本的なプロトタイ プの作成
- RAMP-TS002:アプリケーション・セグメント全体の迅速な刷新
- RAMP-TS003:社員のデータ・フィルターの作成
- RAMP-TS004:従業員画面の命名と分類
- RAMP-TS005:デザインの検証
- RAMP-TS006:基本的な照会画面のスナップイン
- RAMP-TS007:データ入力ファンクションのスナップイン
- RAMP-TS008: 照会画面の更新モードへの変更
- RAMP-TS009:ナビゲーションの追跡
- RAMP-TS010:特別なフィールド・ハンドラーの使用
- RAMP-TS011:付属のドキュメント・コマンド・ハンドラーのス ナップイン
- RAMP-TS012:付属のNotesコマンド・ハンドラーのスナップイン
- RAMP-TS013:インスタンス・リストのデータをExcelに送信する
- RAMP-TS014:HRデモ・アプリケーションへのRAMP画面のスナッ プイン

## 5250画面とは?

「5250画面とは?」という質問は、主観的な質問です。 この5250画面は...

🤞 Telephone Search						
部門電話番号の検索	Enter					
社員姓	パーを表示					
	トドキュメンテーション					
	トレース オン					
	スタリーンの調査					

#### この5250画面と同じですか?

💰 Telepho	one Search				1
	部	門電話番号の権	Enter		
社員姓		В		火ューバーを表示	
				トドキュメンテーション	
社員姓	社員名		社員 Na	トレースオン	
BLAKE BLOGGS	FRED JOHN	ALAN	A1031 A0090	スタリーンの調査	
BROWN	FREDDY		A3564		
BROWN	VERONICA		A0070		

外観的には、「いいえ、違います。見たらわかるでしょう」と答えるか もしれません。

ただし、表示されているRPGプログラムの機能を理解したら、「これらの画面は同じ画面です」と答えるかもしれません。プログラム内には、5250画面を実際に表示している1つのポイントのみがあるため、これらの
#### 画面は論理的に同じでなければなりません。

#### 画面の署名

RAMP-TSは、画面に表示されるレコード様式に基づいて、どの画面にも 署名を割り当てます。RAMPツールの使用時に署名を確認できます。

Keymaps AutoGUI Screens	K Session Display Help
一致の候補	
ID 名前 7.17	41 22 49
18 TelephoneSearch 0 🔨	在月灶
<新しい定義>	
最有力候補:TelephoneSearch (18)	
<b>画面の定義</b> 保存 キャンセル <b>削除</b> 自動定義	
書名: A.@SNAME.MSG@SFLC.P@FOOTER.P@HD00	
A@SNAME.MSG@SFLC.P@FOOTER.P@HD0014.R@	F00001
TelephoneSearch (ID = 18)	
別名:	

2番目の画面には、サブファイル制御レコードといくつかのサブファイ ル・レコードが表示されているため、上記のサンプル画面は異なる署名 を持っています。つまり、RAMP-TSはこれらの画面を別の画面と考えま す。

サブファイル以外の他の画面では、同じ種類の異なる署名という状況が 発生することがあります。

例えば、郵送の詳細を示すADDINFOなどのレコード様式が受注の詳細に 表示されることがありますが、これは配達住所が郵送住所が異なる場合 のみ表示されます。つまり、OrderDetailsなどの1つの画面名には実際は2 つのバリエーションがある(2つの異なる署名を持っている)と考えてくだ さい。

このチュートリアルの重要なポイントは、5250画面には異なる署名があること、およびこれらを使用して種々の状況を処理する方法を理解することです。

異なる署名で表示されている同じ画面を処理する

RAMPで、実際には同じ画面であると考える必要がある2つの画面に異なる署名を与えた場合、両方の画面に同じ画面名を付けてこれを容易に解決できます。

つまり、両方の画面バリエーションを処理する1つの画面スクリプトがあ

ります。

異なる画面署名ごとにバリアント名を適用する場合もあります。これに より、1つの画面スクリプトでどの画面を実際に処理しているかを指定で きます。

この状況の処理方法については、「RAMP-TS015 ステップ1.同じ名前を 2つの画面バリエーションに割り当てる」を参照してください。

同じ署名で表示されている異なる画面を処理する

異なる画面と見なしても同じ署名を持っている場合があります。

通常、これはi5/OSシステム・コマンド画面に当てはまります(すべて署 名QDUI132.USRRCDを持っています)。

画面のIDフィールドなどの追加の詳細を選択して、これらの画面を一意 に識別することができます。例えば、画面のタイトルがあります。これ を行うと、各画面に異なる画面名が割り当てられ、画面はそれ自体の一 意の画面処理スクリプトを持ちます。

「RAMP-TS015 ステップ2. 同じ署名を持つ異なる画面を処理する」を参照してください。

異なる画面をグループまたは一連の画面として処理する

一連の異なる画面全体がよく似た動作特性を持っていることがあります (コード・テーブル・メンテナンス・プログラムなど)。

各画面は異なる署名を持っていますが、すべてに同じ画面名を割り当て た場合、すべての画面を管理する1つの画面スクリプトを持つことができ ます。

1つの画面スクリプトで多数の異なる画面スクリプトを処理できるため、これは生産的です。

一般に、異なる画面ごとに別のバリアント名を割り当てます。これにより、1つの画面スクリプトでどの画面を実際に処理しているかを指定できます。

「RAMP-TS015 ステップ5. 一連の画面の作成(高度)」を参照してください。

RAMP-TS015 ステップ1. 同じ名前を2つの画面バリエーション に割り当てる

このステップでは、異なる署名を持つ画面の2つのバリエーションに同じ 名前を割り当てます。1つは空のサブファイルであり、もう1つはエント リーがあるサブファイルです。

2つのバリエーションを同じ画面として定義することで、1つのスクリプ トを使用して画面間のナビゲーションを制御できます。

このステップで使用するファンクションは人事システムのTelephone Searchファンクションです。このファンクションは画面を表示する1つの REQUESTステートメントを含んでいます。

画面が初めて表示されると、レコード様式が異なるため、従業員と電話 番号のリストを表示する時期に対する異なる署名を持っています。

- RAMPツールで、RAMP-TS 5250エミュレータ・セッションを開始します。
- 2. 社員管理メイン・メニューに移動します。

LANSA run pslsys partition(dem)

- 3. オプション7部門電話番号の検索を選択します。
- 4. 画面にTelephoneSearchという名前を付けて、[Screens} タブの[保存]ボ

#### タンをクリックします。

Keymaps AutoGUI Screens	K Session Display Help	🔎 - Auto - 💌 🔍 🥘
-致の鉄橋 10 名前 7.17	御門電話番号の後素	
《新しい定義》	11 R M	-
────────────────────────────────────		
「画面の定義		
保存 キャンセル 削除 自動定義		
₩4: A.@SNAME.MSG@SFLC.P@FOOTER.P@HD00		
名前:		
TelephoneSearch		
<u> </u>		
2述:		
	F1= ヘルブ F3= 終了 F4=PROMPT F12= 取用し I	14=#SG
*7*7>4A:		
開始 林了 名前 行/エット 見出し いうしん ワンド・ 行		
×		
74-AL*: Dock Screen		
名計 行列 Sfl		
1 20 の検索		
		a 🔤 🚬 📲 T

- 5. [社員姓]フィールドに文字を入力して、Enterを押します。画面にサ ブファイルと別の署名が表示され、RAMPは画面を[Unknown Form]と して表示します。
- 6. この画面に同じ名前TelephoneSearchを付けて、名前を保存します。

Ka 217 前 217 所以定第> ▲:	社員胜 · · · · · · · · · · · · · · · · · · ·	쉐門電話콜号. <u>8</u> 社長名	の検索	40 P1	TelephoneSearch ↓ Enter
前 スコフ 所し、定論> 画:	社員姓 社員姓 BLAKE	· · · · · · · · · · 5 社員名		<del>2</del> 0 P1	↓ Enter
新U./定碗>	社員姓 BLAKE	社員名		40 P9	↓ Enter
<u>نه:</u>	社員姓 BLAKE	<b>社員名</b>	**		
·····	BL AKE		11 = 101	電話番号	Unknown Form
	BLOGGS	JOHN FRED JOHN ALAN	A1031 A0090	(02) 9922 5588 654 6475 X432	
**	BROWN	FREDDY	A3564	(02) 456-5678	
キャンセル 削除 自動定義	BR.OWN	VERONICA	A0070	(02) 9647 2788	
neSearch					
~	F1= へルプ F	3= 我了 F4=PRONPT F12= 1	取描し F14=	NSG	
~					

- 7. TelephoneSearch画面をDestinationとして定義します。
- 8. 画面をキャンセルし、オプション7をもう一度選択します。
- 9. [社員姓]フィールドに文字を入力して、Enterを押します。両方の TelephoneSearch画面がDestinationとして定義され、画面とスクリプト のリストに1つの画面として表示されます。



- 10. RAMP定義の部分的な保存を実行します。
- 次に、TelephoneSearch画面を関連付けるビジネス・オブジェクトを作成 する必要があります。
- 11. [フレームワーク]ウィンドウで、インスタント プロトタイピング アシスタントを使用して、新しいビジネス・オブジェクトTelephone Searchを作成し、これに[詳細]コマンドを追加して人事アプリケーショ

ンに関連付けます。

- 12. Telephone Searchビジネス・オブジェクトが作成されたら、次の手順を実行します。
  - プロパティを開きます。
  - デフォルトで作成されている新しいフィルターを削除します。
  - [使用可能コマンド]タブで、このコマンドをビジネス・オブジェ クト・コマンドにします。
  - フレームワークの再起動を確認するメッセージで[閉じる]ボタン をクリックします。
  - プロパティを閉じます。
- 13. [RAMPツール]ウィンドウを表示します。
- 14. 画面とスクリプトのリストでTelephoneSearch Destination画面を選択し、その詳細を表示します。
- 15. [関連するコマンドハンドラー]リストを更新します。
- 16. Telephone Searchビジネス・オブジェクトの詳細コマンド・ハンド ラーをコマンド・ハンドラーとして選択します。

関連するコマンドハンド	<del>7</del>	
コマント (タブ)へ	わジェ外内	2 ^
厕新しいヴループ	2-ザーと権限	۷
🔲 プロフィール削ル涂	2-ザーと権限	۷
■ 縮小	実行時に VLF を修正	M
📰 言羊糸田	<del>7</del> A	M
	-7 Β	M
■メモを使用	テキスト エディター例集	T
	Code Tables	5
📰 副羊糸田	Employees	F
■ 新規	Employees	F
ト*キュメント	Employees	F
A*21	Employees	F.,
■ 11 11 11 11 11 11 11 11 11 11 11 11 11	Telephone Search	6
		~
	再表示	

17. Enterを除くすべてのファンクション・キーとボタンを無効にしま

す。

- 18. [フレームワーク]ウィンドウに切り替えて、[フレームワーク]メ ニューから[保存して再起動]オプションを選択します。
- 19. [フレームワーク]ウィンドウが再起動したら、人事アプリケーショ ンとTelephone Searchビジネス・オブジェクトを選択します。
- 20. コマンド・ハンドラーをテストします。

Telephone Search					
ファイル(E) 編集(E) 表示(V)	^ルプ( <u>H</u> ) ウィンドウ( <u>M</u>	( フレームワーク ) (管理)			
⑨作成 (1) 印 []	M 🛆 🔊 🗖	📮 🖾 () 🔕 🗠 🍓 🗿 📾	510	勿検索	
Telephone Search					
X					X
⊞ 🥵 HR デモアブリケーション ⊯ 🔶 お気に入り	🤞 Telephone Se	arch			
🖃 🚳 人事		部門電話番号	の検索	Enter	
Employees	社員姓	B		メニューバーを表示	
Telephone Search				ト・キュメンテーション	
🖿 🚳 管理	社員姓	社員名	社員N	トレース オン	
	BLOGGS	FRED JOHN ALAN	A103 A009	スクリーンの調査	
	BROWN	FREDDY	A356		
	BROWN	VERONICA	A007		
		24/# 40 (400 TO 1		(07.45 4004 C	
	メッセージ	準1/2 AS/400 TS JPN V	ZUPGMLIB 10	1/07/15 18:24	

TelephoneSearch Destinationの任意のフィールドに名前を付ける場合は、 両方の画面バリエーションで行う必要があります。

# RAMP-TS015 ステップ2. 同じ署名を持つ異なる画面を処理する

このステップでは、同じ署名(すべてのi5/OSシステム・コマンド画面は 同じ署名を持つ)、System i Main Menu、およびWork with Active Jobsを持 つ2つのi5/OSシステム・コマンド画面に異なる名前を付けます。

画面の追加の詳細をIDフィールドとして選択し、これらの画面を一意に 識別します。

- 1. RAMP-TS 5250エミュレータ・セッションのSystem i Main Menu画面 に移動します。
- 画面にはすでにi5OSMainMenuという名前を付けていますが、この画面 は他のi5/OS画面と同じ署名を持っているため、その定義にさらに情報 を追加して一意に識別する必要があります。
- 2. 定義の前にあるチェック・ボックスにチェックを入れて、定義に画 面のタイトルを追加します。



3. [Screens]タブの[保存]ボタンをクリックします。

次に、Work with Active Jobs画面に名前を付けます。

4. RAMP-TS 5250エミュレータ・セッションのWork with Active Jobs (WRKACTJOB)画面に移動します。

- 5. [Screens]タブにあるタイトルのチェック・ボックスにチェックを入れて、画面定義にタイトル部分を追加します。
- 6. 画面名にWrkActJobと入力します。

Keymaps AutoGUI Screens	K Session Display Help	🗩 - Auto - 💌 🔍 🧔
RA: A.QDUI132.USRRCD	▲	ASI002 10/07/15 19:08:24 號 : 252
名前: WrkActJob	オブションを入力して、実行キーを押してください。 2= 変更 3= 保留 4= 終了 5= 処理 6= 解放 23	=x14-ご の表示
別名:	B=スポーモ、カ羽t の処理 13= 切所 ・・・ 現行	
	OPT ナボニスmm/ブュボ マーナ゙- jbボ CPU % 機能 AXES QSYS S8S .0	状況 DEQW
	AXES AXES BCH . 0 PGN-AXESW3 AXES AXES BCI . 0 PGM-LUAS	SELW TIDW
H121_/8 -	AXES AXES BCI .0 PGM-AXESDE AXES AXES BCI .0 PGM-AXESTS AXES AXES BCI .0 PGM-AXESTS	T DW S IGW
77 774	CCSP GNL IS QSYS S85 .0 LISTENER CCSP GNL IS ASJ .0 CHD -LANSA	DEQW TINW
<u></u>	TP 00000022 CCSP GML IB BCH . 0 PGM - LCOTP	DEQW \$\$\$ <
	パラメーターまたはコマンド ===>	
フィールド: Lock Screen 名前 行 제 Sfl	F3= 終了 F5= 最新表示 F7= 検索 F10= 叙計の再 F11= 谷通 m <sup>*</sup> -jの表示 F12= 取り消し F23=e4 <sup>*</sup> -1] 親	81動 き F24= キーの親き
活動ジョブの処理		
		上 🗵 🗶 🚬

- 7. [保存]ボタンをクリックします。
- 8. WrkActJob画面をDestinationとして分類します。
- 以上で、スクリプトが関連付けられた独自の定義を持つ2つの画面が得 られました。



- 9. 次に、Work with Active Jobs画面とのナビゲーションを追跡します。
- 10. RAMP定義の部分的な保存を実行します。

RAMP-TS015 ステップ3. 画面バリエーションを作成する

このステップでは、Work with Active Jobs画面のさまざまなビューをバリ エーションとして識別し、表示されているビューをスクリプトで処理で きるようにします。

- 1. 最初のビューにバリアント名Statusを入力します。
  - 2. ステータス列見出しを画面定義に追加します。

Keymaps AutoGUI Screens	K Session Display Help 🔑 - Auto - 🔽 🔍 🧔
WrkActJob (ID = 20)	店動ジョブの発理         AS IO02           10/07/16         09:18:38           CPU %:         ・0<           経過映間:         00:00:00           活動ジョブ放:         244
Status 記述:	オブションを入力して、実行キーを押してください。 2= 変更 3= 保留 4= 終了 5= 処理 6= 解放 7=xx1-1ごの表示 8=x4「=5,4796その処理 1.3= 切断
	現行 OPT ナ#'ニカmw/ニ'ュ#' マーナ'- jb#' CPU X 機能 メモニ AXES AXES SS 0 DECM AXES AXES BCH 0 PCM-AXESW3 SEI W
サ <b>フラファイル:</b> 開始 称了 名前   行/エント 見出し リ∽ 行	AXES AXES BCI .0 PGM-LUAS TINN AXES AXES BCI .0 PGM-AXESDE TINN AXES AXES BCI .0 PGM-AXESTS SIGN
< × ×	AXES         AXES         BCI         .0         PGM-AXESS250         DEQA           CCSP GAL IB         QS YS         S85         .0         DEQM           LISTENER         CCSP GAL IB         ASJ         .0         OHD-LANSA         THM           T TR00000023         CCSP GAL IB         ASJ         .0         BCH-LOTE         DECM
고~	- 170000022 CC3704213 SCH 10 P08-2C01P 0504 ポイ・・・
名前 行 列 Sfl	> F3= 终了 F5= 最新表示 F7= 彼东 F10= 統計 の再始動 F11= 裕迅 =「-」の表示 F12= 取り消し F23=e4「-」」 穂を F24= キーの穂を
9 47	
ダ         9         63           状況	

- 3. [画面]タブの[保存]ボタンをクリックします。
- 4. Work with Active Jobs画面のF11を押して、次のビューを表示します。
- 5. 画面のタイトルとElapsed列見出しを画面定義に追加します。

Key	maps AutoGUI Scr	eens	1		~	Session Di	isplay Hel	p			,₽ -	Auto - 💌 🔍 🤯
77	9 <b>11:</b>	-	(T.J. 8 W)		~			ŝ	舌動ジ	ョブの処理	10/	ASI002 07/16 09:18:38
6697	5 ¥4 J - 2 M	11	/137 鬼西し - 竹	2	CPI	%: .0	経過時	· []	00:00	0:00 活動	ジョブ数 : 2	44
				~	オ	プションを入け	カして,実	行キー	を押し	てください。		
				~		= 変更 3=	保留	4= 終了	5=	処理 6= 解	放 7=x3ネーニ*	の表示
			Lock:	Screen		=:(@*~t,@3Dt	0) 処 理	13= QJI	UT		程淵	
74-1	11 <sup>11</sup>	E	Di off		OP1		°ı‡° jb‡°	<b>4°</b> ~₹	PTY	CPU INT	RSP AUXIO	CPU %
	5=9012	17	<b>%</b> 5fi		_	AXES	S8 S		0	.0	0	.0
		6	16	~	_	AXES	BCH	2	20	1.7	0	.0
		0	40		_	AXES	BCI	2	20	.0	0	.0
	6= 解放				_	AXES	BCI		20	.0	0	.0
		6	56		<b>—</b>	AXES	SCI	4	20	.6	0	.0
					_	CC SPOULTR	SS C	2	20	. /	0	.0
	7=>:aネーニ゙の表示		-	_		I TETENE	124 0		20	.0	0	.0
		7	4			TP00000	177 RCH		20	.0	0	.0
	8=ヌポーモ、ホヲbその処理								20			続く
Г		7	26		- 253	ラメーターま) ◇	たはコマン	۴				
	13=切断				E F3	終了 F5=	最新表示	F7=	奥索	F10= 統	計の再始動	
<b>V</b>		8	48		F11	=203 m² , m² ~j	の表示	F12=	取り道	il F23≕e‡°	[11] 読き F24:	= キーの読き
	経過	2										
		9	2									

- 6. 画面の名前(WrkActJob)とバリアント名Elapsedを入力します。
- 7. [Screens]タブの[保存]ボタンをクリックします。
- 8. Work with Active Jobs画面のF11を押して、次のビューを表示します。
- 9. タイトル---- スレッド ----に対応するフィールドにチェックを入れて スレッド列見出しを画面定義に追加します。
- 10. バリアント名Threadsを入力します。
- 11. [画面]タブの[保存]ボタンをクリックします。
- 12. F11を押して、種々のビューがバリアント名を持っていることを確認します。
- 13. RAMP定義の部分的な保存を実行します。

次に、Work with Active Jobs画面をフレームワークにスナップインすると きに使用できるアプリケーションとビジネス・オブジェクトを作成しま す。

14. [フレームワーク]ウィンドウで、インスタント プロトタイピング アシスタントを使用して新しいアプリケーションSystem i Server:を作 成します。

- ビジネス・オブジェクトActive Jobsを作成します。
- Active Jobsを[詳細]コマンド・ハンドラーに関連付けます。
- アプリケーションSystem i Serverを作成します。
- 新しいアプリケーションにActive Jobsを追加します。
- 15. Active Jobsのフィルターを削除し、[詳細]をビジネス・オブジェクト・コマンドにします。
- 16. フレームワークを保存して再起動し、RAMPツールを起動します。
- 17. [RAMPツール]ウィンドウで、WrkActJob画面をActive Jobsビジネ ス・オブジェクトの[詳細]コマンド・ハンドラーと関連付けます。
- 18. キャンセル・キーとボタンを無効にします。
- 19. セッションオプション、SESSION\_Dを選択し、別のセッションで Destination画面を実行します。
- 20. RAMP定義の部分的な保存を実行します。

### RAMP-TS015 ステップ4. スクリプトで画面バリエーションを 使用する

このステップでは、画面バリエーションを使用して、表示されたビュー に従ってF11キーに対応するボタンのキャプションを設定します。

- 1. [フレームワーク]ウィンドウで、System i Serverアプリケーションと Active Jobsビジネス・オブジェクトを選択して、Work with Active Jobs 画面を表示します。
- 2. F11を押して画面の別のビューを表示します。
- 3. F1キーに対応するボタンのキャプションは、スタティック・キャプ ションであるDisplay Elapsed Dataを持っていることがわかります。

表示されているビューに応じてボタンの適切なキャプションを設定する には、表示されているバリエーションを確認し、

OVERRIDE\_KEY\_CAPTION\_SCREENファンクションを使用してボタン のキャプションを設定する必要があります。

- 4. [RAMPツール]ウィンドウに切り替えて、WrkActJob画面のスクリプトを表示します。
- 5. vHandle\_ARRIVEファンクションで、bReturn変数宣言の後にIF... ELSE IF... ELSEステートメントを追加し、画面バリエーションが表示 されているかどうかを確認し、ボタンのキャプションを設定します。

```
if (this.vLatestVariant == "Status")
{
    OVERRIDE_KEY_CAPTION_SCREEN("WrkActJob", KeyF11, "Show E
}
```

else if (this.vLatestVariant == "Elapsed")

OVERRIDE\_KEY\_CAPTION\_SCREEN("WrkActJob", KeyF11, "Show T }

else

OVERRIDE\_KEY\_CAPTION\_SCREEN("WrkActJob", KeyF11, "Show Si

#### スクリプトは以下のようになります。

{ 14 14 #/ ===== vHandle\_ARRIVE =========== /# \_\_\_\_\_ #1 /\* Handle arrival at this Destination /\* oPayload: The payload supplied by the event initiator /\* oPreviousForm: Reference to previous Form object vHandle\_ARRIVE: function(oPayload, oPreviousForm) ł var bReturn = true; if (this.vLatestVariant == "Status") OVERRIDE\_KEY\_CAPTION\_SCREEN("WrkActJob", KeyF11, "Show Elapsed Data"); else if (this.vLatestVariant == "Elapsed") OVERRIDE\_KEY\_CAPTION\_SCREEN("WrkActJob", KeyF11, "Show Thread Data"); else OVERRIDE\_KEY\_CAPTION\_SCREEN("WrkActJob", KeyF11, "Show Status"); SHOW\_CURRENT\_FORM(true); /\* Show the form in the framework and show VLF buttons /\* Hide any 5250 style buttons displayed HIDE\_5250\_BUTTONS(); GET\_FORM\_MESSAGE(22);

/\* Extract messages and hide the message line /\* Last thing done - turn off the busy state SETBUSY(false);

#1

/\* <ARRIVE /> - Do not remove or alter this line \*/

#### 変更をコミットし、RAMP定義の部分的な保存を行います。 6.

7. [フレームワーク]ウィンドウに切り替えて、変更をテストします。 表示されている画面バリエーションに応じてボタンのキャプションが 変わります。

着 Active Jobs	
ファイル(E) 編集(E) 表示(V) ヘルブ	<u>H</u> ) ウィンドウ(W) (フレームワーク) (管理)
	🔊 🖸 📕 🕼 🔿 🧶 🧏 😹 📄 Spooled Files 🔞 終了 🛛 🕅 外分検索
Active Jobs	
× ■ ♣ HR デモアプリケーション ■ ♣ System i Server ♣ Active Jobs ■ ★ お気に入り ■ ♠ 人事 ■ � プログテミングテウニック ■ ♣ 養理	★ Active Jobs 活動ジョブの処理 10 Enter 10 CPU %: 97.8 経道時間: 00:02:17 活動ジョブ数: 日報表示 オブションを入力して,実行キーを押してください。 2=変更 3= 保留 4= 終了 5= 処理 6= 解放 7=xaネージ 統計の再始動 8=ヌホ*→モ、ホヲbモ の処理 13= 切断 現行 Show Elapsed Data
	OPT       チャーン(1*3本*)       マーナ*・       うらホ*、CPU %       機能       eホ*(13) 続き         AXES       QSYS       SBS       .0       eホ*(13) 続き         AXES       AXES       BCH       .0       PGM-AXESW3       キーの続き         AXES       AXES       BCI       .0       PGM-AXESW3       キーの続き         AXES       AXES       BCI       .0       PGM-AXESDE       メニューパ*ーを表示         AXES       AXES       BCI       .0       PGM-AXESDE       メニューパ*ーを表示         AXES       AXES       BCI       .0       PGM-AXESTS       ド*ロッパ*ーを表示         AXES       AXES       BCI       .0       PGM-AXES5250       ド*ロッパ*ーを表示         CCSPGMLIB       QSYS       SBS       .0       トレースオン         LISTENER       CCSPGMLIB       ASJ       .0       CMD-LANSA       アリーンの調査         TP00000022       CCSPGMLIB       BCH       .0       PGM-LCOTP       スリーンの調査
📧 🖽 📑 F3= 終了F5= 最新表示F7	バラメーターまたはコマンド ===> = 検索F10= 統語 メッセージ 準備 AS/400 TS JPN V20PGMLIB 10/07/16 11:58 🥥

#### RAMP-TS015 ステップ5. 一連の画面の作成(高度)

このチュートリアルでは、各画面が異なる署名を持っている場合でも、 同じ画面名を付けてよく似た動作特性を持つ一連の画面を作成します。 この方法では、1つの画面定義とスクリプトを使用してすべての画面を管 理します。

このチュートリアルで使用されている画面は、「RAMP-TS002:アプリ ケーション・セグメント全体の迅速な刷新」のフレームワークの人事ア プリケーションにある以下のCode Tablesビジネス・オブジェクトからア クセスします。

- Review/Maintain/Print Department Table
- Review/Maintain/Print Section Table
- Review/Maintain/Print Skill Table

これらの画面をまずご覧ください。

- 1. [RAMPツール]ウィンドウで、[ナビゲーション スクリプトの自動更新]オプ ションが選択されていることを確認します。
- 2. RAMP-TS 5250エミュレータを使用して、テーブル保守メニュー画面 に移動します。
- 3. オプション1. [Review/Maintain/Print Department Table]を選択します。
- 4. Review/Maintain/Print Department Table画面に*ReviewMaintPrint*と名前を付けます。
- 5. この特定の画面をスクリプトで処理したい場合は、バリアント 名DepartmentTableを付けます(これはオプションであり、通常は必要な ことがわかっている場合のみバリアント名を付けます)。

Keymaps AutoGUI Screens	K Session Display Help	🗩 - Auto - 💌 🔍 🤯
<ul> <li>一致の候補</li> <li>10 名前</li> <li>スコア</li> <li>&lt;本</li> <li>≪</li> </ul>	新田コード・テーブル 新田門 SEL コード 前門名称 ADM ADMINISTRATOR DEPT AUD INTERNAL AUDITING	
最有力候補: <b>画面の定著</b> 保存 キャンセル 肖耶奈 自動定義 <b>暑名:</b> A.©MDEPTAB.MSG©SFLC.P©FOOTER.P©HD 名前: ReviewMaintPrint <b>別名:</b> DepartmentTable	FLT FLEET ADMINISTRATION GAC GROUP ACCOUNTS INF INF ORMATION SERVICES LEG LEGAL DEPARTMENT NIS MANAGENNT INFORMATIO MKT MARKETING DEPARTMENT SD SALES & DISTRIBUTION TRVL TRAVEL DEPARTMENT	
22述:	<b>処理する御門を指定するか、「差加」キーで新しい御</b> F1= ヘルブ F3= 終了 F4=PROMPT F6= 進加 F12 F23= 印刷	<b>村を差加してください。</b> = 歌順し F14=#SG
開始		 ★

- 6. 画面定義を保存し、画面をキャンセルします。
- 7. オプション2. [Review/Maintain/Print Section Table]を選択します。
- 8. 画面にReviewMaintPrintという名前を付けます。
- 9. オプションとして、バリアント名SectionTableを付けます。
- 10. 画面定義を保存し、キャンセルしてメイン・メニューに戻ります。
- 11. オプション3. [Review/Maintain/Print Skill Table]を選択します。
- 12. ここでも、画面に*ReviewMaintPrintと*いう名前を付けて、オプションとしてバリアント名*SkillTable*を付けます。
- 13. 画面定義を保存します。
- 14. トラッキング情報エリアで、3つの画面すべてに同じ名前が付けられていることを確認します。
- 15. 画面をDestinationとして分類します。

画面とスクリプトのリストで、ReviewMaintPrintの1つの画面定義とそれ を制御する1つのスクリプトがあることを確認します。



共通の画面定義があるため、これを変更すると、3つの画面すべてに影響 があります。

16. 終了キーとボタンを無効にします。

ReviewMaintPrint	H I I						
🗉 💽 Scripts		<b>+</b> -	Caption	利用可能	利用可	FKEY a	
🕀 🥪 Special (2)		Enter	Enter	<ul> <li>Image: A set of the set of the</li></ul>	<ul> <li>Image: A set of the set of the</li></ul>	1	
표 🔤 スクリプト (12)		F1	F1			2	
		F2	F2			3	
		F3	終了			4	-
		F4	PROMPT		~	5	
		F5	F5			6	
		F6	追加	<b>~</b>	<b>~</b>	7	
		F7	F7			8	
		F8	F8			9	
		F9	F9			10	
		F10	F10			11	
		F11	F11			12 🔍	2
>			Win Sale a			10	9

- 17. RAMP定義の部分的な保存を実行します。
- 18. フレームワークを再起動します。
- 19. [フレームワーク]ウィンドウに切り替えて、前のチュートリアルで 作成したCode Tablesビジネス・オブジェクトを選択します。
- 20. 確認/保守/印刷の各オプションを選択して、3つの画面すべてで終了 キーとボタンが無効になっていることを確認します。

di Code Tables									
ファイル(E) 編集(E) 表示(V) ヘルプ(	<u>ዘ</u> )  ሳሪኑን	₩) (7V-47	ーク)(管理)						
	③作成 🗇 ြ 🔟 🛆 🕥 🗖 🦉 🖾 🖓 🥹 🖉 🕹 🖉 🧐 🖓 🏷 🌾 😽								
Code Tables									
X						X			
⊞ 🤹 HR デモアブリケーション	🍓 Code	Tables				114			
■ System i Server ■ A お気に入り			噐	コード・テーブル	Enter				
		<b>≠</b> ℝ 月月	≣∎		PROMPT				
Telephone Search	SEL	コード	- F	課名称	追加				
⊞ 🎯 J*aþ*ラミンþ* テウニック □ 🎽 毎冊		ADM	01	INTERNAL ADMIN SR	取消し				
п 🕫 ен		ADM	02	PURCHASING SECTIO	MCC				
		ADM ADM	03	SALES & MARKETING	MDG				
	-	ADM	05	MAINTENANCE	ED刷				
	I H	ADM	06	PERSONNEL SECTION	メニューバーを表示				
		ADM	09	VEHICLE MAINTENAN					
		AUD	01	ADMINISTRATION	トキュメリテーション				
		AUD	02	PURCHASING	トレースオン				
		AUD	03	ACCOUNTING	ったい、小の調査				
	-	FLI	02	PUPCHASTNG	入99年20月前五				
	-	FLT	03	ACCOUNTING					
		GAC	FC	FINANCIAL CONTROL					
E 📰 🔳		メッセージ	準備 AS/4	00 TS JPN V20PGMLIB 1	0/07/16 15:40 🥥				

#### まとめ

重要なポイント

- RAMPは、画面のレコード様式に基づいて画面に署名を割り当てます。
- 異なる署名を持つ2つ以上の画面に同じ名前を付けることができます。これを実行するときは、これらの画面が1つの画面定義で定義され、画面を処理する1つのスクリプトのみがある必要があります。
- スクリプトで別な方法で画面のバリエーションを処理したい場合 は、画面の異なるバリエーションを識別するオプションを使用できま す。
- 作成および保守する画面定義とスクリプトの数を最小限に抑えることは非常に生産的です。
- 反対に、同じ署名を持つ画面に別の名前を付けることもできます。
   この場合、画面を一意に識別する画面要素を画面定義に追加する必要があります。これは、すべてが同じ署名を持っているi5/OSシステム・コマンド画面で一般的に行われています。

覚えるべき事柄

- 画面署名とは何か
- 異なる署名を持つ画面に同じ名前を付ける方法
- 別の画面と署名を共有している画面の識別方法
- 画面名バリエーションを作成および使用する方法

高度なチュートリアル

以下の高度なチュートリアルでは、RAMP-TSの概念と技術をデモンスト レーションします。

RAMP-TSAD01: ボタンを使ったアプリケーションへの価値の付加

RAMP-TSAD02: RAMP-TSのイベント処理の基本

RAMP-TSAD03:特別なフィールド受け渡し

RAMP-TSAD04: aXesを使用して画面を再設計する

RAMP-TSAD05:SHAREDプロパティおよびファンクションを使用する

RAMP-TSAD06: 複数のタブで複数の画面を処理する

RAMP-TSAD07:1つのタブで複数の画面を扱う

RAMP-TSAD08:画面ラッパーの基本

RAMP-TSAD09: サブファイル付きの画面ラッパー

RAMP-TSAD01:ボタンを使ったアプリケーションへの価値の 付加

このチュートリアルでは、フレームワークのボタンを使用して、刷新された5250アプリケーションに便利な機能を追加する方法を示します。 目的

- ボタンを使って刷新された画面に機能を追加する方法を学習する。
- 刷新された画面のボタンをクリックする(またはファンクション・ キーを押す)操作には、5250アプリケーションとの対話処理が不要で あることを理解する。クライアント上でボタンやファンクション・ キーを取り込んで機能を追加できます。
- RAMPコマンド・ハンドラーからWindowsクリップボードにデータ をコピーする方法を学習する。



この目的を達成するには、以下のステップを完了してください。 参照先:ボタンについて RAMP-TSAD01 ステップ1.フレームワークのボタンの有効化 RAMP-TSAD01 ステップ2. DisplayEmployee画面でコピーするフィールド

に名前を付ける

RAMP-TSAD01 ステップ3. DisplayEmployee画面のスクリプトにファンク ションを追加する

RAMP-TSAD01 ステップ4. ButtonClickファンクションでのファンクションの呼び出し

RAMP-TSAD01 ステップ5. ボタンのテスト

まとめ

始める前に

このチュートリアルを完了するには、コア・チュートリアルRAMP-TS001〜RAMP-TS015が完了している必要があります。

## ボタンについて

RAMPアプリケーションには、フレームワーク・ボタンと5250ボタンの2 種類のボタンがあります。

フレームワーク・ボタン

フレームワーク・ボタンは、画面のRAMP定義のDestination画面用に設定 されます。

<b>‡</b> -	Caption	利用可能な52	利用可能なYL	> FKEY se 🔺	^
Enter	Enter	<b>&gt;</b>	<ul> <li>Image: A start of the start of</li></ul>	1	
F1	F1			2	
F2	F2	<b>V</b>		3	
F3	終了			4	
F4	PROMPT			5	
F5	F5			6	
F6	Сору			7	_
F7	Retrieve		~	8	
F8	F8			9	
F9	F9			10	
F10	F10			11	
F11	F11			12	
F12	取消し			13	
F13	F13			14	
F14	MSG			15	
F15	F15			16	
F10	E10			17	~

資格情報の検索/保守													Enter						
社員	Na.																:	A0070	Сору
社員	姓																	BROWN	
社員	名																	VERONICA	Retrieve
住所	1																	12 Railway	
住所	2																	Baulkham H	
住所	з																	NSW Austra	
自宅	電話	푭	亏															(02) 9609	
응 [ 1]		۴																INF +	
眼コ	- 1	-																DV +	
7 2+	日付	1	DD	MA	in	n												90/01/28 +	
退職	HIT	2	DD	MN	in	n		0					0			0		0/00/00 +	
-						1	1	-	-	÷.	0	-	<u>_</u>	-	-	2	<u> </u>	0,00,00	
TD 2县					22	F #2	z												
CDDMM	~				-		.,	-			12	- 15		1.53					
CODMIN	06/0	-	/24		-							10		+				unt 1 Mate	
	96/0	0	/ 51	1	AD	T IN	1	_	_		- di			S C I	a		20	Met 2	
98/03/01 ADMIN2 Administratn											Adr	n	Pa	irt 2					
	95 /1	2	/11	DI.	ΔD	VP	GN		_		Ad	var	1Ce	ed	Pr	.00	ara	ummi na	

すべてのDestination画面でフレームワーク・ボタンを使用する必要があ ります。Junction画面および特殊な画面のボタン表示は無関係です。これ らの画面は表示されません。

フレームワークをデザイナとして実行するときは、開発タスクを容易に するために設計時ボタンも表示されます。



5250ボタン

5250ボタンは、刷新された5250アプリケーションの一部であるボタンです。

🌼 Employee : Details (A2007-BERESFORD BRUCE C)
Details Documents A Events
Employee Number
Date Skl         Skill         Skill           Acquired         Code         Description         Comment         Grade           0/00/00         0/00/00         0         0         0         0           0/00/00         0/00/00         0         0         0         0
Help Exit Prompt Cancel Messages 5250ボタン

デフォルトでは、すべてのDestination画面で5250ボタンは非表示です。 5250ボタンの表示は、SHOW\_5250\_BUTTONS()および HIDE\_5250\_BUTTONS()ファンクションで制御されます。

i /* vHandle_ARRIVE */ /* vHandle_ARRIVE */ /* Handle arrival at this Destination */ /* oPayload: The payload supplied by the event initiator */ /* oPreviousForm: Reference to previous object Form*/	
<pre>vHandle_ARRIVE: function(oPayload, oPreviousForm) {     var bReturn = true;     if (CHECK_FIELD_EXISTS("txtSurname"))</pre>	
i SHOW_CURRENT_FORM(true); /* Show the form in the framework a <u>GET_FORM_MESSAGE(22); /* Extract messages and hide the me</u> <u>HIDE_5250_BUTTONS(); /* Hide any 5250 style buttons disp</u> SETBUSY(false); /* Last thing done - turn off the b } else { SENDKEY(KeyF21); }	and show <u>essage</u> 1 <u>plaved</u> pusy sta

デフォルトでは、5250ボタンは不明な画面または未定義の画面で表示されます。この動作は、特殊な不明なフォームの定義で制御されます。

**RAMP-TSAD01 ステップ1. フレームワークのボタンの有効化** このステップでは、[クリップボードにコピー]ボタンをDisplayEmployee 画面に追加します。

- 1. [RAMPツール]ウィンドウで、DisplayEmployee Destination画面の詳細 を表示します。
- 2. ファンクション・キーF6の[利用可能なボタン]列選択します(未使用の任意のキー/ボタンを使用できます)。
- 3. キャプションを[クリップボードにコピー]に変更します。

<b>+</b> -	Caption	利用可能な5250	利用可能な¥Lp F	KEY sec
Enter	Enter		Image: A start and a start	1
F1	F1			2
F2	F2	<b>~</b>		3
F3	終了			4
F4	PROMPT			5
F5	F5			6
F6	クリップボードにコピー			7
F7	F7	100 C		8
F8	F8			9
F9	F9			10
F10	F10			11
F11	F11			12
F12	取消し		Ē	13
F13	F13			14
F14	MSG			15
F15	F15			16
F16	F16		Ē	17
F17	F17			18
F18	F18		Ē	19
F19	F19			20
F20	F20			21

このキャプションのボタンがDestination画面に表示されますが、キーが 有効になっていないため、F6のキーストロークは5250アプリケーション に送信されません。ここでF6キーが有効になっていても、5250画面に存 在しない場合、5250アプリケーションに影響はありません。 RAMP-TSAD01 ステップ2. DisplayEmployee画面でコピーする フィールドに名前を付ける

このステップでは、DisplayEmployee画面でコピーするフィールドに名前 を付けます。

- 1. RAMPツールでRAMP-TSエミュレータ・セッションを開始します。
- 2. [北゙ゲーション スクリプトの自動更新]チェック・ボックスがオフになってい ることを確認します。
- 3. DisplayEmployee画面に移動し、F21キーを押して編集モードにします。
- (従業員番号)および[名前]フィールドに、それぞれtxtEmpnoと txtGivenameという名前を付けます。[姓]フィールドには、前のチュー トリアルでtxtSurnameという名前が付けられています。



フィールドの名前は大文字と小文字が区別されることに注意してくださ い。

5. [Screens]タブの[保存]ボタンをクリックします。

RAMP-TSAD01 ステップ3. DisplayEmployee画面のスクリプト にファンクションを追加する

このステップでは、DisplayEmployeeのスクリプトにファンクションを追加して、[社員]、[姓]、[名]フィールドの内容をWindowsクリップボードにコピーします。

- 1. DisplayEmployee Destination画面のスクリプトを見つけます。
- 以下のファンクションをNavigateToファンクションの 後、//<SYSINFO>ブロックの前のスクリプトにコピーして貼り付けま す。

```
uCopyEmpDetails : function () {
```

```
/* Get details from 5250 screen */
var TAB_Char = "\x09" ;
var End_Of_Line_Char = "\x0D\x0A" ;
```

```
var strEmpno = GETVALUE("txtEmpno");
var strGName = GETVALUE("txtGivename");
var strSName = GETVALUE("txtSurname");
```

```
/* Write details to clipboard */
var MyString = "";
```

```
MyString = strEmpno + TAB_Char + strGName + TAB_Char + strSName + E
COPYTOCLIPBOARD(MyString);
```

```
/* Issue a message */
MESSAGE("Details for employee ", strEmpno," sent to the clipboard");
```

```
},
```

3. [変更をコミット]ボタンを使用して、スクリプトの変更をコミットしま す。 RAMP-TSAD01 ステップ4. ButtonClickファンクションでの ファンクションの呼び出し

このステップでは、クリップボードにコピーするファンクションを ButtonClickファンクションから呼び出すコードを追加します。

- 1. DisplayEmployeeスクリプトのButtonClickファンクションのSwitchコマンドにF6キー(KeyF6)のcaseステートメントを追加します。
- 2. F6キーのステートメントで、uCopyEmpDetailsファンクションの呼び 出しを追加します。

case KeyF6:
/\* Call copy function \*/
this.uCopyEmpDetails();
break;

3. [変更をコミット]ボタンを使用して、スクリプトの変更をコミットしま す。その後、[保存]を実行します。

#### RAMP-TSAD01 ステップ5. ボタンのテスト

このステップでは、[クリップボードにコピー]ボタンをテストします。

- 1. [フレームワーク]ウィンドウで、Personnel Applicationを見つけます。
- Employeesビジネス・オブジェクトを選択し、フィルターを使用して インスタンス・リストに入力します。
- 3. 従業員を選択し、その詳細を表示します。
- 4. [クリップボードにコピー]ボタンをクリックします。

🔲 詳細 🔥 イヘシ	Nxe#1 🔄 1		
	Enter		
社員Na		: A0090	クリップボードにコピー
社員姓		BLOGGS	とう かん 大東二
社員名		· · · · · · · FRED JOHN ALAN	入口一八一名武示
住所1		70 MAIN STREET	ドキュメンテーション
住所2		NEWTOWN NSW	
住所3		AUSTRALIA	トレースオン
自宅電話番号			フカリーンの調査
部門コード・		FLT +	X// /Valeid
課コード		03 +	
入社日付 (DDM	MYY)		
退職日付(DDM	MYY)	0/00/00 +	
取得日	資格		
(DDMMYY)	リード	資格記述 コメント	
98/03/25	ADMIN1	Administratn Part 1 Met requirement	
98/05/01	ADMIN2	Administratn Part 2 G	
98/05/04	COM	Communications Degre	

従業員の詳細情報がクリップボードにコピーされたことを示すメッセー ジが表示されます。

😇 📰 🖪 Details for employee 🛛 A0090 sent to the clipboard

- 5. クリップボードの内容を貼り付ける別のアプリケーションを起動し ます(MS WordやExcelなど)。
- 6. 従業員の詳細情報を貼り付けます(Ctrl + V)。

- ■ X											
	*-4	挿入 ペ	ージ レイアウト	要权主	¢ э	データ	校閲	表示	0 -	•	×
に 貼り クリッフ	★ 付け  グボード □	MS Pゴシック B Z <u>U</u> - 一 マー フォント		副置	% 数値	▲ スタイル	セル	Σ + ● * 0 ② *	₩ -		
	A1	<b>•</b>	fx f	A009	90						≯
4	A	В	С	D		E		F	(	G .	
1	A0090	FRED JOH	BLOGGS								
2				ß							=
3											
4											
5											
6											
7											
8											-
14	► ► Shee	et1 / Sheet2	Sheet3 /			4	III			•	ī
עדב	15		データの	個数:3		0 🛄 10	00% 🕞		,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	-+	

#### まとめ

重要なポイント

- [利用可能なVLF]チェックボックスが選択され、[利用可能な5250] チェックボックスが選択されてないファンクション・キーは、フレー ムワーク専用ボタンとして機能します。フレームワーク・ボタンの処 理はクライアントですべて処理され、キーストロークは5250アプリ ケーションに送信されません。「Function Key Enablement」を参照し てください。
- ファンクション・キーのcaseステートメントは、キーボードのキー が使用されたり、ボタンがクリックされた場合の動作を制御します。
- SETKEYENABLEDファンクションを使用して、ボタンとファンクション・キーを動的に有効または無効にすることができます。
- OVERRIDE\_KEY\_CAPTION\_SCREENファンクションまたは OVERRIDE\_KEY\_CAPTION\_ALLファンクションを使用して、フレー ムワークのボタンのテキストを動的に変更することができます。ま た、OVERRIDE\_KEY\_CAPTION\_ALLファンクションを使用して、す べてのファンクション・キーのキャプションを多言語アプリケーショ ンの別の言語に設定することもできます。

#### 覚えるべき事柄

- フレームワークのボタンを使用して、刷新された5250アプリケーションに価値を付加する方法
- フレームワークのボタンを有効にし、ButtonClickファンクションを 変更して、ボタンがクリックされたときの処理を提供する方法
- フォーム・スクリプト内の任意の場所から使用可能なファンクションをフォーム・スクリプトに追加する方法
#### RAMP-TSAD02: RAMP-TSのイベント処理の基本

このチュートリアルでは、RAMPスクリプトがフレームワーク・コン ポーネントに、またその逆方向にイベントを通知する方法を示します。 目的

- RAMP画面がフレームワークにイベントを通知する方法を学習する
- フレームワークがRAMPのシグナルをリッスンする方法を学習する
- フレームワークがRAMP画面にイベントを通知する方法を学習する
- RAMP画面がフレームワークのシグナルをリッスンする方法を学習 する



この目的を達成するには、以下のステップを完了してください。 RAMP-TSAD02 ステップ1. [名前]フィルターへのシグナル・ボタンの追 加

RAMP-TSAD02 ステップ2.5250画面で通知をリッスンするように設定する

RAMP-TSAD02 ステップ3. フィルターからRAMP画面への通知のテスト RAMP-TSAD02 ステップ4. RAMP画面へのシグナル・ボタンの追加

RAMP-TSAD02 ステップ5. 通知をリッスンするようにフィルターを設定する

RAMP-TSAD02 ステップ6. RAMPスクリプトからVLFコンポーネントに 通知する

まとめ

始める前に

このチュートリアルを完了するには、コア・チュートリアルRAMP-TS001〜RAMP-TS015が完了している必要があります。 RAMP-TSAD02 ステップ1. [名前]フィルターへのシグナル・ボ タンの追加

このステップでは、「RAMP-TS003:社員のデータ・フィルターの作 成」で作成したフィルターを変更します。これにより、RAMPで処理した5250画面にシグナルを送信するボタンを追加します。

- 1. フィルター再利用可能パーツiiiRMP01を開きます。
- 2. [一般制御]タブからボタンをフィルターにドラッグし、キャプショ ンを[RAMPスクリーンへシグナルを送る]にします。

EOMRMP01 - By Name Filter	×
🥅 リストの消去	検索
社員姓	
ABCDEFGHIJKLMNOPQRST	
Send Signal to RAMP Screen	
<	1

2. 以下のコードをフィルター・ソースに追加して、ボタンのクリック・イベントを処理します。

\* Handle the signal 1 button by broadcasting FILTER\_SIGNAL\_1 with 5 alpha and 5 numeric payload items

EVTROUTINE HANDLING(#PHBN\_1.Click)

Invoke #avListManager.GetCurrentInstance AKey1(#vf\_elxak1) AKey2(#vf\_elxak2) AKey3(#vf\_elxak3) AKey4(#vf\_elxak4) AKey5(#vf\_elxak5) NKey1(#vf\_elxnk1) NKey2(#vf\_elxnk2) NKey3(#vf\_elxnk3) NKey4(#vf\_elxnk4) NKey5(#vf\_elxnk4) Found(#vf\_elBool) VisualId1(#VF\_ELXVI1) VisualId2(#VF\_ELXVI2) BusinessObjectType(#vf\_elidn)

Invoke #Com\_Owner.**avSignalEvent** withId(FILTER\_SIGNAL\_1) To(FRAMEWORK) SendAInfo1(#com\_Owner.avObjectType) SendAInfo2(#vf\_elxak1) SendAInfo3(#VF\_ELXVI2) SendAInfo4("text1") SendAInfo5("text2") SendNInfo1(1.1) SendNInfo2(2.2) SendNInfo3(3.3) SendNInfo4(4.4) SendNInfo5(5.5)

ENDROUTINE

#### コードは以下のようになります。

* Handle the signal 1 button by broadcasting FILTER_SIGNAL_1 with 5 alpha and 5 numeric payload items
-EVTROUTINE HANDLING(#PHBN_1.Click)
<pre>Invoke #avListManager.GetCurrentInstance AKey1(#vf_elxak1) AKey2(#vf_elxak2) AKey3(#vf_elxak3) AKey4(#vf_elxak4) AKey5(#vf_elxak5) NKey1(#vf_elxnk1) NKey2(#vf_elxnk2) NKey3(#vf_elxnk3) NKey4(#vf_elxnk4) NKey5(#vf_elxnk4) Found(#vf_elBool) VisualId1(#VF_ELXVI1) VisualId2(#VF_ELXVI2) BusinessObjectType(#vf_elidn)</pre>
<pre>Invoke #Com_Owner.avSignalEvent withId(FILTER_SIGNAL_1) To(FRAMEWORK) SendAInfo1(#com_Owner.avObjectType) SendAInfo2(#vf_elxak1) SendAInfo3(#VF_ELXVI2) SendAInfo4("text1") SendAInfo5("text2") SendNInfo1(1.1) SendNInfo2(2.2) SendNInfo3(3.3) SendNInfo4(4.4) SendNInfo5(5.5)</pre>
ENDROUTINE

L<sub>End\_Com</sub>

GetCurrentInstanceステートメントは、インスタンス・リストの現在選択 しているエントリーの情報を取得します。次に、avSignalEventはこの情 報をフレームワークに送信します。

avSignalEventにより渡される一部の値は単にスタティック・テキストまたは数値です。このコードは、単にコマンドで5つの文字列と5つの数値を渡すことができることを示すためのものです。

3. フィルターをコンパイルします(コンパイルが失敗した場合、フレー ムワークでフィルターが使用されていることが原因である可能性があ ります。この場合は、フレームワークを再起動します)。

# RAMP-TSAD02 ステップ2.5250画面で通知をリッスンするように設定する

このステップでは、FILTER\_SIGNAL\_1をリッスンするように、また通知 を受け取ったときにメッセージを表示するように、DisplayEmployee画面 に関連付けられているスクリプトを変更します。

他のコンポーネントが出力するイベントをリッスンする vHandle\_AVEVENTファンクションを使用します。

- 1. フレームワークが実行されていない場合は、起動します。
- 2. RAMPツールを起動し、DisplayEmployee画面に関連付けられている スクリプトを見つけます。
- 3. このコードをSYSINFOブロックの上に追加します。

```
______
*/
=============
/* ======
*/
vHandle AVEVENT: function(WithId,Sender,WithAInfo1,WithAInfo2,WithAIn
var sText = "";
if (WithId == "FILTER SIGNAL 1")
sText += "RAMP script received signal " + WithId;
sText += "\r Sender = " + Sender;
sText += "\r WithAInfo1 = " + WithAInfo1;
sText += "\r WithAInfo2 = " + WithAInfo2;
sText += "\r WithAInfo3 = " + WithAInfo3;
sText += "\r WithAInfo4 = " + WithAInfo4;
sText += "\r WithAInfo5 = " + WithAInfo5;
sText += "\r WithNInfo1 = " + WithNInfo1.toString();
sText += "\r WithNInfo2 = " + WithNInfo2.toString();
sText += "\r WithNInfo3 = " + WithNInfo3.toString();
sText += "\r WithNInfo4 = " + WithNInfo4.toString();
sText += "\r WithNInfo5 = " + WithNInfo5.toString();
alert(sText);
```

### コードは以下のようになります。

/\* Show details from clipboard \*/ ALERT\_MESSAGE("Employee Details from the Clipboard are: ", "Employee Number: ", strEmpno,"Name: ", strGNam

},

/₩ ₩/
//# #/
// ///
VHandle AVEVENT: function(withId Sender withAInfol withAInfo2 withAInfo3 withAInfo4 withAInfo5 withNInfo1 wi
Var SText = "":
Sicke - ,
if (withId == "FILTER SIGNAL 1")
SText += "RAMP script received signal " + WithId:
sText += "\r Sender = " + Sender;
sText += "\r withAInfol = " + withAInfol;
sText += "\r withAInfo2 = " + withAInfo2;
sText += "\r withAInfo3 = " + withAInfo3;
sText += "\r WithAInfo4 = " + WithAInfo4;
sText += "\r WithAIntos = " + WithAIntos;
sText += "\r withNintol = " + withNintol.tostring();
siext += \r withNinto2 = + withNinto2.toString();
stext += \r withNinfos = + withNinfos.tostring();
$s_1 = x_1 + y_1 $
$s_1 = x_1 + s_1$ with $s_1 = s_1 + w_1(s_1) + s_2(s_2)$
J return(true).
l e con necho de contra de cont
19 Anno 19
// <sysinfo></sysinfo>
vName : "DisplayEmployee",
vguid : "63005ae2681e45b4a04c616b4da0d5ee",

4. 変更をコミットし、RAMP定義の部分的な保存を行います。

RAMP-TSAD02 ステップ3.フィルターからRAMP画面への通知のテスト

このステップでは、フィルターに追加したボタンをテストします。

- 1. [フレームワーク]ウィンドウに切り替えます。
- Personnelアプリケーションで、Employeesビジネス・オブジェクトを 選択します。
- 3. [名前]フィルターで、[RAMPスクリーンへシグナルを送る]ボタンをクリック します。

何も起こりません。これは、通知を受け取るRAMP画面が表示されてい ないためです。

- フィルターを使用してインスタンス・リストを生成し、リストの従 業員を選択して、資格情報の検索/保守画面が表示されるのを待ちま す。
- 5. [RAMPスクリーンへシグナルを送る]ボタンをもう一度クリックします。

フィルターは、フレームワーク全体の通知を出力します。 DisplayEmployee画面のvHandle\_AVEVENTファンクションは、この通知 を受け取り、通知の受信とペイロードの内容を示すメッセージ・ボック スを表示します。



#### RAMP-TSAD02 ステップ4. RAMP画面へのシグナル・ボタン の追加

このステップでは、DisplayEmployee画面を変更して、フレームワークに シグナルを通知するボタンを追加します。

- 1. [RAMPツール]ウィンドウで、DisplayEmployee画面の詳細を表示し ます。
- 2. [ファンクション キー使用可能]セクションで、F17ボタンを有効にし、その キャプションをフィルターへシグナルを送るにします。

<b>+</b> -	Caption	利用可能な5:	利用可	KEY 🔺	^
F7	F7			8	
F8	F8			9	
F9	F9			10	
F10	F10			11	
F11	F11			12	
F12	取消し			13	
F13	F13			14	
F14	MSG			15	
F15	F15			16	
F16	F16			17	Ξ
F17	Send Signal to Filter		>	18	
F18	F18			19	
F19	F19			20	
F20	F20			21	
F21	変更			22	_
F22	肖·JB余			23	
F23	F23			24	
F24	F24			25	
ClearSo	c Clear			26	
PageUp	Page Up			27	
PageDov	w Page Down			28	~

次に、ボタン・スクリプトにシグナル・コードを追加する必要がありま す。

3. vHandle\_BUTTONCLICKファンクションで、F17キーの以下のCaseス テートメントを追加します。

case KeyF17: /\* Send RAMP\_SIGNAL\_1 \*/
objGlobal.txtEmpno = GETVALUE("txtEmpno");
objGlobal.txtGivename = GETVALUE("txtGivename");
objGlobal.txtSurname = GETVALUE("txtSurname");

### AVSIGNALEVENT("RAMP\_SIGNAL\_1","FRAMEWORK",objGlo break;

#### コードは以下のようになります。

	case KeyF14: SENDKEY(KevF14):
	break;
	<pre>case KeyF17: /* Send RAMP_SIGNAL_1 */     objGlobal.txtEmpno = GETVALUE("txtEmpno");     objGlobal.txtGivename = GETVALUE("txtGivename");     objGlobal.txtSurname = GETVALUE("txtSurname");     AVSIGNALEVENT("RAMP_SIGNAL_1", "FRAMEWORK", objGlobal.txtEmpno, objGlobal.txtGivename, objGlob     break;</pre>
}	default: SENDKEY(sButton); break;

このコードは、画面のtxtEmpno、txtGivename、txtSurnameフィールド(前のチュートリアルでこれらのフィールドに名前を付けている)の値を取得し、このペイロードのシグナルをフレームワークに送信します。

また、AVSIGNALEVENTファンクションが5つの文字列と5つの数値をペ イロードで渡すことができることを示すために、いくつかのスタティッ ク・テキストおよび数値文字列も送信します。

4. 変更をコミットし、RAMP定義の部分的な保存を行います。

5. フレームワークを保存して再起動します。

#### RAMP-TSAD02 ステップ5. 通知をリッスンするようにフィル ターを設定する

このステップでは、フィルターでRAMP\_SIGNAL\_1をリッスンし、通知のペイロードを示すメッセージを表示するコードを追加します。

- 1. Visual LANSAエディターで、再使用可能なパーツiiiRMP01を開きま す。
- 2. EvtRoutine #Com\_owner.avEventイベント・ルーチンを見つけます。
- 以下のコードをEndcaseステートメントの前に追加し、 RAMP\_SIGNAL\_1をリッスンして、通知のペイロードのメッセージを 発行します。

```
when '= RAMP_SIGNAL_1'
```

```
Use message_box_add ('VLF Filter received signal ' #EventId.Value)
```

- Use message\_box\_add (' WithAInfo1 =' #AInfo1)
- Use message\_box\_add (' WithAInfo2 =' #AInfo2)
- Use message\_box\_add (' WithAInfo3 =' #AInfo3)
- Use message\_box\_add (' WithAInfo4 =' #AInfo4)
- Use message\_box\_add (' WithAInfo5 =' #AInfo5)
- Use message\_box\_add (' WithNInfo1 =' #NInfo1) Use message\_box\_add (' WithNInfo2 =' #NInfo2)
- Use message\_box\_add ('WithNInfo3 =' #NInfo3)
- Use message\_box\_add ('WithNInfo4 =' #NInfo4)
- Use message\_box\_add ('WithNInfo5 =' #NInfo5)

Use MESSAGE\_BOX\_SHOW

コードは以下のようになります。

```
when '= Delete_List_Entry'
Invoke Method(#avListManager.BeginListUpdate)
* Remove instance details from the instance list
Invoke #avListManager.RemoveFromList AKey1(#EMPNO)
Invoke Method(#avListManager.EndListUpdate)
when '= RAMP_SIGNAL_1'
Use message_box_add ('VLF Filter received signal ' #EventId.Value)
Use message_box_add (' WithAInfo1 =' #AInfo1)
Use message_box_add (' WithAInfo2 =' #AInfo2)
Use message_box_add (' WithAInfo3 =' #AInfo3)
Use message_box_add (' WithAInfo5 =' #AInfo5)
Use message_box_add (' WithAInfo5 =' #AInfo5)
Use message_box_add (' WithNInfo1 =' #NInfo1)
Use message_box_add (' WithNInfo2 =' #NInfo2)
Use message_box_add (' WithNInfo3 =' #Info3)
Use message_box_add (' WithNInfo3 =' #Info3)
Use message_box_add (' WithNInfo5 =' #NInfo3)
Use MESSAGE_BOX_SHOW
```

4. フィルターをコンパイルします。

RAMP-TSAD02 ステップ6. RAMPスクリプトからVLFコン ポーネントに通知する

このステップでは、RAMPスクリプトからフィルターへの通知をテスト します。

- 1. フレームワークで、インスタンス・リストから社員を選択し、新し いボタンが追加されているDisplayEmployee画面を再表示します。
- 2. [フィルターへシグナルを送る]ボタンをクリックします。

🎽 Employees:詳編	(A0090-BLOGGS FRED JOHN ALAN)	
🗆 詳細 🛛 🛕 イベント 👘	📄 k'4axUk	
	資格情報の検索/保守	Enter
社負Na	A0090	フリップボードにコピー
社員姓	BLOGGS	
社員名	FRED JOHN ALAN	Send Signal to Filter
住所1	70 MAIN STREET	イントバーを表示
住所2	NEWTOWN NSW	111 /1 630/N
住所3	AUSTRALIA	ドキュメンテーション
自宅電話番号 .		11
部門コード	FLT +	10-242
課コード	03 +	スターンの調査
入社日付 (DDMMYY	)	
退職日付 (DDMMYY	) 0/00/00 +	
En 48 CI 14e	40	

RAMPスクリプトはフレームワーク全体にシグナルを出力します。フィルターは、そのEVTROUTINE HANDLING(#Com\_Owner.avEvent)ルーチンでこの通知を受け取り、通知の受信とペイロードの内容を示すメッセージ・ボックスを表示します。

Employees     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ▼     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■ □ ■     ■     ■ □ ■     ■     ■     ■ □ ■     ■     ■     ■				
		a 🕫 🛛 🦻	/妙検索	
Employees				
× ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	■ 名前 ■ By Date of Birth ■ By Salary マリストの消去 社員姓 検索 B By Name Filter	Employees A0070 A0090 A1031 A3564	来 記述 BROWN VERONICA BLOGGS FRED J BLAKE JOHN BROWN FREDDY	
■ 🥇 管理	VLF Filter received signal RAMP_SIGNAL_1 WithAInfo1 = A0090 WithAInfo2 = FRED JOHN ALAN WithAInfo3 = BLOGGS WithAInfo4 = RS1 Info4 WithAInfo5 = RS1 Info5 WithNInfo1 = 111.100000000 WithNInfo2 = 122.100000000 WithNInfo3 = 133.10000000 WithNInfo5 = 155.10000000 WithNInfo5 = 155.10000000 WithNInfo5 = 155.10000000 WithNInfo5 = 155.10000000 WithNInfo5 = 155.10000000	RED JOHN ALAN) 報の検索/保守 BLOGGS BLOGGS FRED JOHI NEWTOWN 1 AUSTRALI BLT + BLT +	× Enter クリップボードにコピー Send Signal to Filter メニューバーを表示 ドキュメンテーション トレース オン スグリーンの調査	
	メッセージ 準備 AS/400 TS	S JPN V20PGMLIB 1	10/07/21 9:24 🅥	

#### まとめ

重要なポイント

- シグナルは非同期です。シグナルの発行、受け取りと処理は同期イベントではありません。同期的な場合もありますが、そうでない場合もあります。アプリケーションのコードは、シグナルが非同期であるとして常に記述する必要があります。
- シグナルの識別子の命名規則を作成し、使用されるすべてのシグナルと関連するペイロードをドキュメント化する必要があります。
- シグナルは相対的に高価な要求です。これらは、MOUSE\_MOVED などの低レベルのものではなく、EMPLOYEE\_UPDATEDや LIST\_DATA\_CLEAREDなどの高レベルのものを処理する必要があり ます。
- RAMP Destination画面のvHandle\_AVEVENTファンクションは、現 在の5250画面が表示されているときのみシグナルをリッスンします (つまり、シグナルをリッスンできるのは現在のRAMP画面だけで す)。
- vHandle\_AVEVENTファンクションに渡される2番目のパラメータ (Sender)は、WINアプリケーションのみで使用できます。WEBアプリ ケーションや.NETアプリケーションでは、このパラメータは使用不 可能で、常に空の文字列として渡されます。
- RAMP Destination画面は、独特なvHandle\_AVEVENTファンクションを含んでいることがあります。

覚えるべき事柄

 フィルターとRAMPコマンド・ハンドラー間でイベントを通知する 方法 RAMP-TSAD03:特別なフィールド受け渡し

自動プロンプトの機能を示す

RAMPの特別なフィールド受け渡し(プロンプト)を使用すると、既存の 5250画面に機能や価値を付加することができます。特別なフィールド・ ハンドラーは、PC上で実行されるVisual LANSAフォームであり、5250ア プリケーションでは不可能な機能を作成できるようにします。

一般に、1つの定義と命名規則により多様な5250画面に機能を追加できます。

目的

🔲 詳細 🔥 イベント 📄 ドキュメント **睡**報の検索/保守 Enter このチュートリアルでは、5250画面の . : A0090 クリップボードにコピー 特別なフィールド受け渡しの詳細に BLOGGS ついて学習します。 Send Signal to Filter FRED JOHN ALAN 70 MAIN STREET バューバーを表示 また、使用可能な付属のフィールド・ NEWTOWN NSW ドキュメンテーション ハンドラー(独自のフィールド・ハンドラー . AUSTRALIA . 344-2234454545 を作成することもできる)、一般的な トレースオン FLT + フィールド/ハンドラーの関連付け、 フィールド・ハンドラーを動的に 03 の調査 210710 ++ + 7月2010 . .. 割り当てる方法を示しています。 0/00/ 月火水木金土日 28 29 30 1 2 3 4 5 6 7 8 9 10 11 取得日 資格 12 13 14 15 16 17 18 コード (DDMMYY) 資格記述 19 20 21 22 23 24 25 26 27 28 29 30 31 1 Administratn Part 1 98/03/25 ADMIN1 2 3 4 5 6 7 8 98/05/01 ADMIN2 Administratn Part 2 98/05/04 COM Communications Degre 閉じる OK Computer Science Deg 98/05/05 CS

この目的を達成するには、以下のステップを完了してください。 RAMP-TSAD03 ステップ1. プロンプタを表示する状況を理解する RAMP-TSAD03 ステップ2. HANDLE\_PROMPT()の高機能な使い方 RAMP-TSAD03 ステップ3. ハンドラーのスタイル RAMP-TSAD03 ステップ4. 汎用ハンドラーの関連付け RAMP-TSAD03 ステップ5. 日付ピッカーへの日付フィールドの汎用的な 関連付け RAMP-TSAD03 ステップ6. ハンドラーの動的な関連付け RAMP-TSAD03 ステップ7. ハンドラーの動的な関連付け

#### RAMP-TSAD03 ステップ8. 機能しない場合に実行する内容 まとめ

始める前に

このチュートリアルを完了するには、コア・チュートリアルRAMP-TS001〜RAMP-TS015が完了している必要があります。

#### RAMP-TSAD03 ステップ1. プロンプタを表示する状況を理解 する

- 「RAMP-TS010:特別なフィールド・ハンドラーの使用」で学習したように、プロンプトはRAMP-TSセッションに関連付けられているテーブルで設定されます。
- 1. RAMPツールを開き、左側の画面とスクリプトのリストで[セッション]ノードをクリックします。
- セッションの詳細の[特別なフィールド受け渡し]テーブルを見つけます。テーブルでは、フィールドutxtDepartmentが5250画面にあり、ユーザーがF2キーを押した場合に特別なフィールド・ハンドラーDF\_PRM04を呼び出すことが定義されています。

🗖 RAMP ウール - デウォルト セッション (RAMP-TS)				
テラオルト セッション	RAMP-TS 5250 セッション 単筆細			
保存 -	- セッション - デブオルト セッション キャプジョン デブオルト セッション フーザー オコジパークト タイフ			
名前 7	- RAMPの既定値のレイアウト サイズ - RAMPの既定値のレイアウト サイズ			
□ toya) □ □ Junction (4)	高さ 380 幅 700 上 左 Top マスクの高さ 25 Bottom マスクの幅			
∎ manneogrin 2	RAMP スクリーン レイアウト スタイル           ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・            ・ </th			
	-スクロールパーー			
	▼ 不明な52507ォームの表示時にフレームワークをロックする ■ 既存のコネケル・ハワーナ・/ハ・マワート、茨面利用する			
	セッションを常にユーザー オフジェクト 名/タイフ を持ったサーバーヘリンクする			
🛨 🕋 ኢንሃን°ኑ (9)	RAMP-TS 最大ログオフ待機時間(秒)         10           特別はフィールド受け渡し         10			
	52507ィートト・名 ファンクションキー YL <b>ハント・ラー (クラス YF_AC017 オブ・ジ・ェクト)</b>			
	1 utxtDepartment F2 DF_PRM04			
	2 F4			
	3 F4			
	4 r4 5 F4			
	6 F4			

DF\_PRM04は、Visual LANSAのコンポーネントです。これは、ソース・ コードとともにVL環境にあります。

この場合、選択したファンクション・キーF2は5250画面で使用できませ

ん。ユーザーがこのキーを押すと、要求は遮断され、クライアントPCで 完全に処理されます。

ただし、5250画面で許可されているキー/ボタンを使用し、フィールドの 特別なハンドラーをこのキーに関連付けた場合でも、要求は遮断され、 クライアントPCで処理されます。これを理解するには、以下の手順を実 行します。

- 3. DisplayEmployee画面に関連付けられているスクリプトを開きます。
- 4. vHandle\_BUTTONCLICKファンクションを見つけます。

/\* ----- BUTTONCLICK \*/ /\* ----- BUTTONCLICK ------ \*/ /\* ----- \*/ /\* sButton: The button that was clicked \*/ vHandle\_BUTTONCLICK: function(sButton)

HANDLE\_PROMPT()ファンクション呼び出しは、RAMPで提供されるファンクションを呼び出します。

使用されている5250画面のフォーカス・フィールドの名前とファンク ション・キーを使って、このファンクションは特別なフィールド・ハン ドラーを呼び出す必要があるかどうかを判定します。

- 呼び出す必要がある場合、HANDLE\_PROMPT()はtrueを返し、要求 を処理します。ボタン・スクリプトが戻し処理を介して即座に終了す るのはこのためです。
- HANDLE\_PROMPT()は、特別なフィールド・ハンドラーを呼び出 さなかった場合、falseを返します。ボタン・スクリプトは独自のロ ジックを使用してボタン・クリックを引き続き処理します。

ほとんどの場合、プロンプト要求はサーバーによる処理のために5250画 面に送信されます。その理由は特別なフィールド・ハンドラーがないた めです。

5. これをテストするには、DisplayEmployee画面でF2キーを無効にし、 F4キーを有効にします。

var bReturn = true;

‡-	Caption	利用可	利用可	FKEY & 🔨
Enter	Enter	<b>~</b>	<ul> <li>Image: A set of the set of the</li></ul>	1
F1	F1			2
F2	F2			3
F3	終了			4
F4	PROMPT	>		5
F5	F5			6

- 6. 変更を保存し、フレームワークを再起動します。
- 7. 人事アプリケーションでEmployeesビジネス・オブジェクトを見つけ て、社員の詳細を表示します。
- 8. [部門コード]フィールドのF4を押します。5250プロンプタが表示されます。
- 9. RAMPツールを起動し、F4を使用するように[utxtDepartment]フィールドの特別なフィールド受け渡しを変更します。

C	_特別な7ィール・受け渡し			
		52507ィール*名	ファンクションキー	VL ハンドラー(クラス VF_AC017 オブジェクト)
	1	utxtDepartment	F4	DF_PRM04
	2		F4	
	3		F4	
	4		F4	
	5		F4	
	6		F4	×

- 10. RAMP定義を保存し、フレームワークを再起動します。
- 11. 人事アプリケーションで社員の詳細を表示し、F4を押します。特別 なフィールド・ハンドラーが表示されます。

2倍格格の検索/保守       Enter         社員兆       340300         社員名       340300         住所1       70 MAIN STREET         住所2       NeWTOWN NSW         全部番号       344-2234454545         部門コード       71 MAIN STREET         2       1000000000000000000000000000000000000	🔲 詳細 🛛 🔬 イベント 📄 トッキュメ	ント	
社員 Ma 社員 Ma 社員 Ma 社員 Ad 社員 Ad Lag Ad Adustration 中 大主日 付(DDMMYY) 工 ー ド 変格 (DDMMYY) 工 ー ド 愛格 (DDMMYY) 工 ー ド 愛格記述 Adustration Place 本記述 Adustration Place 本記述 Adustration Place 本記述 Adustration Place 本記述 Adustration Place 本記述 Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Place Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration Administration		資格情報の検索/保守	Enter
社員経 社員名 社員名 「RED JOHN ALAN Send Signal to Filter 住所1 住所2 小ーを表示 住所3 AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA	社員 No	: A0090	クリップボードにコピー
住所1 住所2 住所3 小小-を表示 住所3 AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA AUSTRALIA FLT INTERNAL AUDITING FLEET ADMINISTRATION PURCHASING ACCOUNTING B GROUP ACCOUNTS INFORMATION SERVICES INFORMATION SERVICES INF	社員姓		Send Signal to Filter
住所 2	住所1	70 MAIN STREET	
住所3 自宅電話番号	住所2	· · · · · · · NEWTOWN NSW	メニューバーを表示
当宅電話番号 · · · · · · · · · · · · · · · · · · ·	住所3	AUSTRALIA	トドキュメンテーション
部門コート · · · · · · · · · · · · · · · · · · ·	自宅電話番号		
課コート	部門コート	••••••••••••••••••••••••••••••••••••••	DITING へトレースオン
X 荘 日 村 (DDMMYY)		••••••••••••••••••••••••••••••••••••••	ISTRATION bu-'/D調査
取得日 资格 (DDMMYY) コード 资格記述 98/03/25 ADMIN1 Administrath Part 1 98/05/01 ADMIN2 Administrath Part 2 98/05/04 COM Communications Degre		ADMINIST	RATION
取得日 資格 (DDMMYY) コード 資格記述 98/03/25 ADMIN1 Administratn Part 1 98/05/01 ADMIN2 Administratn Part 2 98/05/04 COM Communications Degre	AZ MAKELINI (DEMINITY)	PURCHASI	NG =
取得日 資格 (DDMMYY) コード 資格記述 98/03/25 ADMIN1 Administratn Part 1 98/05/01 ADMIN2 Administratn Part 2 98/05/04 COM Communications Degre		ACCOUNT:	ING
は 19日 9月1日 (DDMMYY) コード 没格記述 98/03/25 ADMIN1 Administratn Part 1 98/05/01 ADMIN2 Administratn Part 2 98/05/04 COM Communications Degre	<b>肺得口 次投</b>	GROUP ACCO     GROUP     GROUP ACCO     GROUP     GROUP	UNTS
ODMMYY)     I     P     R M III       98/03/25     ADMIN1     Administrath Part 1       98/05/01     ADMIN2     Administrath Part 2       98/05/04     COM       Communications Degre		INFORMATION Ⅲ INFORMATION	N SERVICES
98/03/25 ADMIN1 Administrath Part 1 98/05/01 ADMIN2 Administrath Part 2 98/05/04 COM Communications Degre	(DDMMYY) J = P	DE THE SE 24	TMENT 🐱
98/05/01 ADMIN2 Administratn Part 2 G 98/05/04 COM Communications Degre	98/03/25 ADMIN1	Administrath Part 1	>
98/05/04 COM Communications Degre	98/05/01 ADMIN2	Administratn Part 2	
	98/05/04 COM	Communications Degre	
98/05/05 CS Computer Science Deg	98/05/05 CS	Computer Science Deg	

## RAMP-TSAD03 ステップ2. HANDLE\_PROMPT()の高機能な使い方

前述のサンプルHANDLE\_PROMPT()は、ボタン・クリックの開始と同時 に実行されていました。

ただし、この5250画面のすべてのプロンプトがF4を使用して実行されて いることを理解している場合は、次のようにロジックを高速で高機能に することができます。

```
vHandle_BUTTONCLICK: function(sButton)
{
    var bReturn = true;
    /* <BUTTONCLICK /> - Do not remove or alter this line */
    /* Handle function keys and buttons */
    switch (sButton)
    {
        case KeyF4:
        if (!HANDLE_PROMPT()) ALERT_MESSAGE("Position the curso:
        break;
```

```
case KeyEnter:
   SENDKEY(KeyEnter);
```

etc, etc, etc

注:このステップは単にテクニックを示しています。この変更を自分のス クリプトで実行しないでください。

ここでは、HANDLE\_PROMPT()要求が必要なのはF4が使用されたときの みであることを理解しているため、F4が使用されたときのみこの要求が 呼び出されるように、要求をボタン・クリック・ファンクションの先頭 からKeyF4 caseステートメントに移動しました。

また、HANDLE\_PROMPT()が要求(ファンクション呼び出しの前の"!"に 注意してください)を処理しなかった場合は、"F4を使用するときはプロ ンプト可能なフィールドにカーソルを配置してください"と通知するメッ セージ・ボックスが表示されるように、ロジックも追加しました。 これを実行できるのは、F4がクライアント上でのみ処理され、サーバー 上の5250 RPGプログラムで処理されないことを理解しているからです。 これは、処理されないF4要求を5250アプリケーションに渡して、処理さ れるかどうかを確認する前述のステップのデフォルト動作とは対照的で す。

#### RAMP-TSAD03 ステップ3. ハンドラーのスタイル

開発者は、Visual LANSAフォームとしてコードを記述し、必要なことを 実行します。このため、特別なフィールド・ハンドラーの表示や振る舞 いはすべて開発者が決定します。

フレームワークには、サンプル・ハンドラーDF\_PRM01〜DF\_PRM07が 組み込まれています。これらは、独自のハンドラーの作成の基盤として 使用できるさまざまなハンドラーの動作を示します。

ALABAMA - AL ALASKA - AK AMERICAN SAMOA - AS ARIZONA - AZ ARKANSAS - AR CALIFORNIA - CA	<u>DF_PRM01</u> 状態コードのf ト。 従来の"コード
Item Sizes Small - SML Medium - MED Item - M	<u>DF_PRM02</u> ラジオ・ボタ: から衣料品サ・ 択。 従来の"コード
	DF_PRM03従業員番号のネコンボ・ボッ'オ・ボタンをi数のデータがiきに使う従来(ケーター(例:顧客番号)。多くの場合、i(例、名前、電便番号、説明、よる検索をサ;



DF_PRM06 - Prompting Example 6	ノンドラー内
Value to send back to 5250 screen (press Enter to send) DATA	すべての情報(
Prompting Information	
The field being prompted is named SALARY	小しよう。
The value of the field is .00	作成中の問題フ
The field is not in a subfile.	特別なフィー
The prompt key used was F4	
Handler information block 1 contains	ドラーの代わり
Handler information block 2 contains	て、想定とお
Handler Information block 3 contains	レドラーに入り
Field ADDRESS1 was also passed with value	スことを確認
Field ADDRESS2 was also passed with value	ることを確認
Field ADDRESS2 was also passed with value	<u> </u>
Field BUSPHONE was also passed with value	アプリケーショ
Field DEPTMENT was also passed with value ADM	
Field EMPNO was also passed with value	
Field GIVENAME was also passed with value	用します。
Field HOMEPHONE was also passed with value	
Field POSTCODE was also passed with value 0	
Field SALARY was also passed with value .00	
Field SECTION was also passed with value ADMIN2	
Field STARTDATE was also passed with value 0/00/00	
Field SURNAME was also passed with value	
Held TERMDATE was also passed with value 0/00/00	
	DE PRM07
	<u>レー((())</u> 従来の日付ピ
++ + February 2009 → ++	び <u>11000</u> 従来の日付ピ す。このサン
	び 従来の日付ピ す。このサン 前がDATE_nnt
Image: Mon Tue Wed Thu         Fri         Sat         Sun         Sun         Sat         Sun         Sun         Sat         Sun	び 従来の日付ピ す。このサン 前がDATE_nni
Image: Weight of the second system       Image: Weight of the second system       Image: Weight of the second system         Mon Tue Weight of the second system       Image: Weight of the second system       Image: Weight of the second system         Mon Tue Weight of the second system       Image: Weight of the second system       Image: Weight of the second system         Mon Tue Weight of the second system       Image: Weight of the second system       Image: Weight of the second system         26       27       28       29       30       31       1         2       3       4       5       6       7       8	びていてい 従来の日付ピ す。このサン 前がDATE_nni のような5250
Mon Tue Wed Thu       Fri       Sat       Sun         26       27       28       29       30       31       1         2       3       4       5       6       7       8         9       10       11       12       13       14       15	びていて、 従来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。
Image: Weak of the second system       February 2009       > >>         Mon Tue Weak Thu       Fri       Sat       Sun         26       27       28       29       30       31       1         2       3       4       5       6       7       8         9       10       11       12       13       14       15         16       17       18       19       20       21       22	びていた 従来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。 されています。
Hon Tue Wed Thu       Fri       Sat       Sun         26       27       28       29       30       31       1         2       3       4       5       6       7       8         9       10       11       12       13       14       15         16       17       18       19       20       21       22         23       24       25       26       27       28       1	従来の日付ピ す。このサン: 前がDATE_nni のような5250: にリンクする。 されています。
Image: Weight of the system       February 2009       Image: Weight of the system         Mon Tue Weight of the system       Fri       Sat       Sun         26       27       28       29       30       31       1         26       27       28       29       30       31       1         2       3       4       5       6       7       8         9       10       11       12       13       14       15         16       17       18       19       20       21       22         23       25       26       27       28       1         2       3       4       5       6       7       8	従来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。 されています。
Image: Weight of the system       February 2009       > >>         Mon Tue Weight Thu       Fri       Sat       Sun         26       27       28       29       30       31       1         2       3       4       5       6       7       8         9       10       11       12       13       14       15         16       17       18       19       20       21       22         23       24       25       26       27       28       1         2       3       4       5       6       7       8	び 従来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。 されています。 nnnnはフィー fffffffは日付が
Image: Weight of the symbol       February 2009       Image: Weight of the symbol         Mon       Tue       Weight of the symbol       Fri       Sat       Sun         26       27       28       29       30       31       1         2       3       4       5       6       7       8         9       10       11       12       13       14       15         16       17       18       19       20       21       22         23       24       25       26       27       28       1         2       3       4       5       6       7       8         2       3       4       5       6       7       8         2       3       4       5       6       7       8         2       3       4       5       6       7       8         C       Today:       Z4/UZ/ZUU       ZUU       ZUU       ZUU	び 従来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。 されています。 nnnnはフィー fffffffは日付が 形式です。
Image: Weight of the state	従来の日付ピ す。このサン: 前がDATE_nni のような5250: にリンクする。 されています。 nnnnはフィー fffffffは日付が 形式です。
Image: Weight of the second symplectic symplecter symplecter symplecter symplecti symplectis symplecti symplecti	び来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。 されています。 nnnnはフィー fffffffは日付が 形式です。 出荷時は、以
Image: Weight of the state	従来の日付ピ す。このサン: 前がDATE_nnr のような5250: にリンクする。 されています。 nnnnはフィー fffffffは日付が 形式です。 出荷時は、以 <sup>-</sup> 式をサポート
Image: Weight of the second symplectic symplecter symplecter symplecter symplecti symplectis symplecti symplecti	び来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。 されています。 nnnnはフィー ffffffは日付が 形式です。 出荷時は、以「 式をサポート」
Image: Weight of the state	び来の日付ピ す。このサン 前がDATE_nni のような5250 にリンクする。 されています。 nnnnnはフィー ffffffは日付が 形式です。 出荷時は、以 式をサポート」 す。
Image: Weight of the state of the stat	び来の日付ピ す。このサン: 前がDATE_nni のような5250: にリンクする。 されています。 nnnnはフィー fffffffは日付が 形式です。 出荷時は、以 <sup>-</sup> 式をサポート」 す。 CCYYsDDsMN

CCYYsMMsDl
CCYYDDMM
CCYYMM
CCYYMMDD
DDsMMsCCY
DDsMMsYY
DDMMCCYY
DDMMYY
MMsDDsCCY
MMsDDsYY
MMCCYY
MMDDCCYY
MMDDYY
MMYY
SysFmt6
SysFmt8
YYsMMsDD
YYMM
YYMMDD
操作できるサ注 フィールド名
DATE_Start_D
DATE_ORDER です。

ハンドラーには、任意の外観や操作性を適用できることに注意してくだ さい。これらは、始めるときに役立つ付属のサンプルです。外観や動作 に満足できない場合は変更してください。

#### RAMP-TSAD03 ステップ4. 汎用ハンドラーの関連付け

このチュートリアルの最初のステップでは、以下のように5250フィールドと特別なフィールド・ハンドラーDF\_PRM04の特別な関連付けを作成しました。

	52507ィート・名	ファンクションキー	VL バントッラー (クラス VF_AC017 オフィシュェクト)	^
1	utxtDepartment	F4	DF_PRM04	
2		F4		
3		F4		
4		F4		
5	j.	F4		
6		F4		~

つまり、この名前のフィールドを含んでいる5250画面でハンドラー DF\_PRM04が呼び出されます。

したがって、命名規則を使用することで、この名前のフィールドがある すべての5250画面にプロンプトを即座に追加できます。

この種の固有の関連付けは、キー・フィールド(CustomerNumber、 ProductNumberなど)およびコード・フィールド(StateCode、CurrencyCode など)で最も役に立ちます。キー・フィールドでは、多くの場合、ユー ザーが多様な方法で顧客や製品を検索できるハンドラーを関連付けるこ とができます。コード・フィールドでは、多くの場合、ユーザーが使用 したいコードを選択できるコード/デコード・テーブルを表示することが できます。

また、以下のような汎用の関連付けも作成できます。

特別	りなフィール・受け渡し			
	52507ィート・名	ファンクションキー	¥L ለント*ラー (クラス ¥F_AC017 オブジェウ	^
1	utxtDepartment	F4	DF_PRM04	
2	DATE_*	F4	HANDLER1	
3	CODE_*	F4	HANDLER2	
4	CUST_*	F4	HANDLER3	
5		F4		
6		F4		×

ここで、

- 名前がDATE\_で始まるフィールドがある場合、F4が使用されたときにHANDLER1が呼び出されます。
- 名前がCODE\_で始まるフィールドがある場合、F4が使用されたときにHANDLER2が呼び出されます。
- 名前がCUST\_で始まるフィールドがある場合、F4が使用されたときにHANDLER3が呼び出されます。

HANDLER1が日付ピッカーの場合は、フィールドの命名規則を使用して HANDLER1と5250画面の日付フィールドを即座に関連付けることができ ます

(また、aXes eXtensionsを使用して、日付ハンドラー、ドロップ・ダウン なども作成できることに注意してください)。aXesの概要については、 「RAMP-TSAD04:aXesを使用して画面を再設計する」を参照してくだ さい。

同様に、プロンプトされたフィールド名の残りの部分(CODE\_STATE、 CODE\_CURRENCY、またはCODE\_AIRPORT)をHANDLER2でサブスト リング化して、選択に対してどのコード・テーブル(州、通貨、または空 港)を表示する必要があるかを決定できます。この種の"スーパー・プロ ンプタ"は、データ・ソース(つまり、表示するコード・テーブル)が変 わっても、コードの表示と選択の方法が通常同じであるため、一般的に 使用されています。

最後の例で、CUST\_\*は何らかの特別な内容を実行する(おそらく"CUST" (顧客)を使用)、特別なフィールド・ハンドラーの汎用的なタイプを示し ています。

実行できる内容には、名前の検索から、詳細の印刷、クリップボードへの画面の値の貼り付け、MS-Excelドキュメントへの顧客のマッピング、 顧客への電子メールの準備までにわたる可能性があります。特別な フィールド受け渡しが簡素なF4プロンプトよりも多くの機能を備えてい る理由はここにあります。

特別なフィールド受け渡しとは、新しい動作を既存の5250画面に添付す ることです。特別なフィールド受け渡しテーブルを賢明な命名規則と組 み合わせることで、新しい動作を多数の5250画面にコーディングなしで 添付できます。

これらの例では、選択した命名規則を使用して、意図と情報をハンド ラーに通知します。また、意図と情報をハンドラーに通知する正確な方 法もあります。これらの方法については、このチュートリアルの後のス テップで説明します。

RAMP-TSAD03 ステップ5. 日付ピッカーへの日付フィールドの汎用的な関連付け

このステップでは、組み込みの日付ピッカーDF\_PRM07に日付フィールドの汎用的な関連付けを行います。これをテストするには、

DisplayEmployee画面の[入社日付]フィールドおよび[退職日付]フィールド にDate\*で始まる名前を付けます。

- 1. [RAMPツール]ウィンドウでRAMP-TS 5250エミュレータ・セッショ ンを開始します。
- 2. Maintain/Browse Employee and Skill Files画面に移動し、F21キーを押します。
- 3. [入社日付]フィールドにDate\_Start\_DDMMYYという名前を付けて保 存します。
- 4. [退職日付]フィールドにDate\_Termn\_DDMMYYという名前を付けて 保存します。



5. セッションの詳細を表示し、フィールドDate\*を日付ピッカー・コン ポーネントDF\_PRM07に関連付けます。

-特別なフィールド受け渡し				
	5250フィールド名	ファンクションキー	VL //ンドラー (クラス VF_AC017 オブジェウト)	^
1	utxtDepartment	F4	DF_PRM04	
2	DATE*	F4	DF_PRM07	
3		F4		1
4		F4		
5		F4		
6		F4		~

- 6. RAMP定義を保存し、フレームワークを再起動します。
- 7. 従業員の詳細および[入社日付]フィールドと[退職日付]フィールドの プロンプトを表示します。両方のフィールド(および Date\_nnnnn\_ffffffffという名前の他のフィールド)に特別なフィール ド・ハンドラーが表示されます。

🔲 詳細 🛕 イベツト 📄 ドキュメン	•	
	資格情報の検索/保守	Enter
社員Na	: A0090	クリップボードにコピー
社員姓	BLOGGS	Send Signal to Filter
住所1	ZO MAIN STREET	
住所2	NEWTOWN NSW	パーコーハーを表示
住所3	AUSTRALIA	トドキュメンテーション
自宅電話番号		トレースオン
部門그는ド	FLT +	
ネコート	92/08/ 44 4 7 H 2010	
退職日付 (DDMMYY)	·····	+ 8
	28 29 30 1 2	3 4
	5 6 7 8 9	10 11
取得日 資格 (DDMMYY) コード	資格記述 12 13 14 15 16	17 18
98/03/25 ADMIN1	Administratn Part 1 26 27 28 29 30	31 1
98/05/01 ADMIN2	Administratn Part 2 2 3 4 5 6	7 8
98/05/04 COM	Communications Degre 閉じる	ок
98/05/05 CS	Computer Science Deg	

#### RAMP-TSAD03 ステップ6. ハンドラーの動的な関連付け

今までのステップでは、RAMPツールを使用して特別なフィールド・ハンドラーを持続的に定義する方法を学習しました。

また、RAMPスクリプトで特別なハンドラーを動的に定義、変更、および削除することもできます。通常、これはログオン画面スクリプトで行われるため、1回のみ発生します。ただし、この機能は特別な目的のために個々の画面スクリプトで使用することができます。

特別なフィールド・ハンドラーを動的に定義または再定義するには、 SET\_SPECIAL\_FIELD\_HANDLER()ファンクションを使用します。特別 なフィールド・ハンドラーを動的に削除するには、

DROP\_SPECIAL\_FIELD\_HANDLER();を使用します。

ハンドラーの動的な削除は、ハンドラーが現在表示されている場合、そ れに影響を及ぼしません。再表示できなくなるだけです。

- 1. RAMPツールを起動し、DisplayEmployee画面のスクリプトを見つけ ます。
- 2. 画面のvHandle\_ARRIVEファンクション内で、以下のように新しい ハンドラーをutxtDepartmentフィールドに動的に添付します。

SET\_SPECIAL\_FIELD\_HANDLER("utxtDepartment",KeyEnter,"DF\_PRM06"

コードは以下のようになります。

vHandle\_ARRIVE: function(oPayload, oPreviousForm)

var bReturn = true;

if (CHECK\_FIELD\_EXISTS("txtSurname"))

SECIAL\_FIELD\_HANDLER("utxtDepartment",KevEnter,"DF\_PRM06"); /\* Attach an Enter Key handler \*/
SHOW\_CURRENT\_FORM(true); /\* Show the form in the framework and show VLF buttons \*/
GET\_FORM\_MESSAGE(22); /\* Extract messages and hide the message line \*/
HIDE\_5250\_BUTTONS(); /\* Hide any 5250 style buttons displayed \*/
SETBUSY(false); /\* Last thing done - turn off the busy state \*/
else
{
 SENDKEY(KeyF21);
 }
 /\* <ARRIVE /> - Do not remove or alter this line \*/

Enterキーが押されると、DF PRM06ハンドラーが呼び出されます。

#### 3. 変更をコミットし、RAMP定義の部分的な保存を行います。

- 4. 人事アプリケーションで従業員の詳細を表示し、変更したスクリプ トを使用して画面を表示します。
- 5. [部門コード]フィールドでF4を押し、フィールドからDF\_PRM04八 ンドラーを呼び出します。
- 6. [部門コード]フィールドでEnterを押し、フィールドからDF\_PRM06 ハンドラーを呼び出します。

詳細 <u>A</u> イベント   ドキュメント	
資格情報の検索/保守	Enter
社員Na	クリップボードにコピー
社員姓 BLAKE 社員名	Send Signal to Filter
住所1	メニューバーを表示
11±川2 Winston Hills 住所3 NSW Australia 2100	ドキュメンテーション
自宅電話番号	
課コード	5
入社日付(DDMM YY)	^
取得日 資格 (DDMMYY) コード 資格記述 <u>95/12/10</u> ADMIN1 Administratn Part <u>96/06/30</u> ADVPGM Advanced Programm <u>90/06/05</u> COM Communications Der Field DATE_START_DDMMYY was also passed with value OF Field DATE_START_DMY was also passed with value OF	5/05/18 0/00/00
メッセージ 準備 AS/400 TS JPN V20PGMLIB	10/07/21 11:12

つまり、使用するファンクション・キーによって区別した、複数のハン ドラーを同じフィールドに添付できます。

7. 次に、以下のコードをvHandle\_ARRIVEファンクションに追加し て、フィールドに関連付けられているF4ハンドラーをドロップしま す。

DROP\_SPECIAL\_FIELD\_HANDLER("utxtDepartment",KeyF4);

コードは以下のようになります。

	van preturn = true;	
if (	(CHECK_FIELD_EXISTS("txtSurname"))	
	<pre>{     SET_SPECIAL_FIELD_HANDLER("utxtDepartment",KeyEnter,"DF_PRM06"); /* Attach an Enter Key handler */     DROP_SPECIAL_FIELD_HANDLER("utxtDepartment",KeyF4); /* Drop the F4 handler     *     SHOw_CURRENT_FORM(true); /* Show the form in the framework and show VLF buttons */     GET_FORM_MESSAGE(22); /* Extract messages and hide the message line */     HIDE_5250_BUTTONS(); /* Hide any 5250 style buttons displayed */     SETBUSY(false); /* Last thing done - turn off the busy state */ }</pre>	1 */
eise	{ SENDKEY(KeyF21); }	

- 8. リストから別の従業員を選択し、[部門コード]フィールドでF4を使用しても特別なコマンド・ハンドラーDF\_PRM04が表示されないことを確認します(代わりに、5250プロンプタ画面が表示されます)。
- 9. Enterを使用してDF\_PRM06ハンドラーをアクティブにできることを 確認します。

Destination到着スクリプトでは、このような操作は普通ではない(つまり、特別な)ことに注意してください。ほとんどの動的な添付はログオン・スクリプトで1回のみ行われ、そのセッション全体で持続します。

#### RAMP-TSAD03 ステップ7. ハンドラーとの通信

ハンドラーに追加の目的や情報を伝達するには、2つの方法があります。 最初の方法では、HANDLE\_PROMPT()要求で情報を渡しま す。

- 1. RAMPツールを起動します。
- DisplayEmployee画面のスクリプトで、ボタン・スクリプトの HANDLE\_PROMPT()ファンクション呼び出しを見つけて、その呼び出 しに文字列パラメータを以下のように追加します。

HANDLE\_PROMPT("My Parm1","My Parm2","My Parm3")

コードは以下のようになります。

```
vHandle_BUTTONCLICK: function(sButton)
{
```

var bReturn = true;

if (HANDLE\_PROMPT("My Parm1", "My Parm2", "My Parm3")) return(bReturn); /\* If th

/\* <BUTTONCLICK /> - Do not remove or alter this line \*/

- 3. 変更をコミットし、RAMP定義の部分的な保存を行います。
- 4. [フレームワーク]ウィンドウで、従業員の詳細を表示し、画面の変 更されたスクリプトを実行します。
- 5. Enterを押して[部門コード]フィールドをプロンプトし、テスト/デ バッグ・ハンドラーDF\_PRM06を表示します。すると次のように表示 されます。


HANDLE\_PROMPT()ファンクションに渡される引数は、シンボル名 (UARG1、UARG2、UARG3など)でハンドラーに渡されます。Visual LANSAハンドラーは コード内で次のようなメソッド呼び出しを使用し てこれらの値を取得できます。

Invoke Method(#Com\_Owner.uGet5250Field) Name(UARG1) Value(#Std\_Text

このテクニックは、開発者が対話しているハンドラーを理解していると きや一般的にすべてのハンドラーと対話しているときに適切に機能しま す。また、このテクニックは、UARG1、UARG2などの使用目的につい ての規則が恐らく必要になることも指しています。 特定のハンドラーと通信するには、次のテクニックを使用します。 2番目の通信手段では、SET\_SPECIAL\_FIELD\_HANDLER() ファンクションを使用してハンドラーに情報を添付します。

これは、ファンクション呼び出しの最後で3つのオプション・パラメータ を使用して行われます。

6. 前のステップで作成したSET\_SPECIAL\_FIELD\_HANDLERステート メントを以下のように変更します。

SET\_SPECIAL\_FIELD\_HANDLER("utxtDepartment",KeyEnter,"DF\_PF "Other 1","Other 2","Other3");

7. 変更をコミットし、[保存]を実行します。次に、従業員を選択し、

## 画面のスクリプトをもう一度実行します。

8. VF\_PRM06ハンドラーを呼び出します。次のように表示されます。



これら3つの情報ブロックを使用して、適切なハンドラーに情報を伝達す ることができます。以下のサンプルのように、Visual LANSAハンドラー はこれらの値をプロパティとして取得できます。

#Product := #Com\_Owner.uHandlerInfo1
#Customer := #Com\_Owner.uHandlerInfo2
#ZipCode = #Com\_Owner.uHandlerInfo1.toNumber()

RAMP-TSAD03 ステップ8. 機能しない場合に実行する内容

特別なフィールド・ハンドラーで問題がある場合は、デバッグする必要 があります。

ALERT\_MESSAGE()、alert()、MESSAGE\_BOX\_SHOW操作でこれらをデ バッグしようとすると、混乱が生じる可能性があります。

これには2つの理由があります。

- まず、ALERT\_MESSAGEは非同期です。このため、表示されると 想定しているときに表示されないことがあります。
- 次に、これらの処理を使用すると、Windowsメッセージ・ボックス が表示されます。通常、これらはフォーカスを取得します。ほとんど のハンドラーはフォーカスを失うと、それ自体を隠すか、非アクティ ブになります。したがって、一般にメッセージ・ボックスを表示する たびにハンドラーは消えます。これは、VLコード・デバッガーの使 用にも当てはまることがあります。デバッグ・フォーカスやアクティ ブ化に敏感なアプリケーションは技巧的である可能性があります。

最適な解決策は、フレームワーク・アプリケーション・レベルのトレー ス機能を使用し、フレームワーク・ウィンドウとオーバーラップしない ように配置することです。これをオンにした場合、特別なフィールド受 け渡しの大量のトレース情報が表示されます。

これで問題が解決しない場合は、TRACE()呼び出しをRAMPスクリプト に追加し、VLコードに#AvFrameworkManager.AvRecordTraceを追加しま す。

別の役に立つデバッグ機能として、ハンドラーの代わりに付属の DF\_PRM06ハンドラーをプラグインすることができます。これを呼び出 すと、ハンドラーに渡された情報に関する多数の情報が報告され、問題 の解決に役立ちます。

#### まとめ

重要なポイント

- 特別なフィールド・ハンドラーはF4タイプのプロンプトを非常に簡単に実行できます。通常、結果は表示が良好で、高速に動作し、5250 サーバーに対する負荷も少なくなります。
- 多くのレベルで多様な方法により5250フォームのフィールドに動作 を添付することができるため、特別なフィールド・ハンドラーはF4プ ロンプトよりも多くのことを実行できます。
- アプリケーション・トレースは、ハンドラーの問題のデバッグに使用するツールです。
- 5250アプリケーションのすべてのキー・フィールドとコード・ フィールドのリストを記述し、賢明なフィールド命名規則を使用する と、すべての5250画面にプロンプト機能を自動的に添付できます。
- 「高度なプロンプト」も参照してください。

#### 覚えるべき事柄

 特別なフィールド受け渡しを使用して5250アプリケーションに価値 を付加する方法

## RAMP-TSAD04: aXesを使用して画面を再設計する

aXes Designerを使用して、5250画面を自由に再設計することができま す。コンテンツの移動や非表示、ラベルの変更、スタイルの適用、線や グループ・ボックスなどの表示エレメントの使用などを行うことができ ます。

aXes eXtensionsを使用すると、コンテンツをドロップ・ダウン、カレン ダー・ドロップ・ダウン、チェック・ボックス、ラジオ・ボタンとして 表示したり、新しいエレメントをボタン、イメージ、ツールチップ、ハ イパーリンク、Googleマップとして追加することもできます。

aXes eXtensionsで実行できる内容は、特別なフィールド・ハンドラーな どで実行できる内容と部分的にオーバーラップしていることがわかりま す。どの手法を採用して画面を拡張するかは自分で決定します。

目的

• aXes Designerを使用して画面を再設計する方法を学習する



この目的を達成するには、以下のステップを完了してください。 RAMP-TSAD04 ステップ1. aXes Designerの開始 RAMP-TSAD04 ステップ2. スタイルのセットアップ RAMP-TSAD04 ステップ3. 頻繁に出てくる情報を非表示にする RAMP-TSAD04 ステップ4. ツールチップの追加 RAMP-TSAD04 ステップ5. ドロップ・ダウンの追加 RAMP-TSAD04 ステップ6. グループ・ボックス内のフィールドの整理 RAMP-TSAD04 ステップ7. サブファイルへの上へボタンおよび下へボタ ンの追加 RAMP-TSAD04 ステップ8. ファンクション・キーを非表示にしてピク チャーを追加する

RAMP-TSAD04 ステップ9. ハイパーリンクの追加

RAMP-TSAD04 ステップ10. 再設計した画面のテスト

RAMP-TSAD04 ステップ11. 画面のカスタマイズの削除

#### 始める前に

このチュートリアルを完了するには、コア・チュートリアルRAMP-TS001〜RAMP-TS015を終えている必要があります。

## RAMP-TSAD04 ステップ1. aXes Designerの開始

このステップでは、[aXesデザイナー]ウィンドウを使用して画面の再設 計を開始する方法を学びます。また、画面上の要素のプロパティを設定 する方法についても学びます。

- 1. RAMPツールで、DisplayEmployee画面の詳細を表示します。
- 2. [RAMPの規定値のレイアウト サイズ]セクションで、[Topマスクの高さ]が0 またはブランクに設定されていることを確認します。

RAMP-TS 5250 セッション 詳細		
DisplayEmployee		
-		
\$++7 <sup>(1</sup> /2 <sup>1</sup> )		
カッシュン カッルーフ <sup>®</sup>		
-RAMPの既定値のレイアウト サイン	χ*	
高さ 380 幅	700 上 2	左
Top マスクの高さ	Bottom マスクの幅	
-RAMP スクリーン レイアウト スタイル-		
<ul> <li>セッション</li> </ul>	○ 固定レイアウト 〇	) 7ローレイアウト

- トップ・マスクは、RAMPで画面タイトルを非表示にするために使用されます。このチュートリアルでは、aXesを使用してタイトルを非表示にする方法を学ぶため、タイトルを表示します。
- 3. RAMP-TS 5250エミュレータ・セッションを起動します。
- 4. DisplayEmployee画面に移動し、変更モードで表示します。
- 5. [aXesデザイナー]ウィンドウで、[この画面のカスタマイズを開始し ます。]をクリックします。



#### 画面は、5250エミュレータ・スタイルなしで表示されます。5250画面の 境界が細い赤の線で示されていることがわかります。

拄負的.										•						:	A1234
计员性																	JACKSON
计具名					•				•	•		•					STEPHEN
主所 1												•		•			6 Meliasa Place
1116日1月11日																	West Pennant Hills
主所3				•	•	•	•	•		•	•		•	•			NSW Australia 2125
182	ΞŦ	号															(02) 9871 7773
1111 ⊐	- 1		•			•				•			-	•	•		SD +
<b>R</b> – –	۲	•	•	•			•	•	•	•	•			•			FS +
入社日	Ħ	(D1	MIN	III	)	-	•	•		•	•	•	•	•			96/08/14 +
入社 日イ 波職 日イ	村 村	(DI		YY	)			•	•	•		:	•	•	•	•	98/08/14 + 0/00/00 +
した 日本 記載日本	Ħ Ħ	(DI (DI	MM	YY	)				•					•		•	98/08/14 + 0/00/00 +
、社日/ 2購日/ 27月日	村 村	(DI (DI		IYY IYY J		•	•		•	•		•					95/09/14 + 0/00/00 +
社日	村 村	(DI (DI		YY		- K	•	•	•	ji			· ·	•			<u>98/09/14</u> + 0.00/00 + コメント 資格
社日 開日 R R R 保日 D M M Y S	17 17 17 17 17 17 17	(DI (DI		YYI YYI Ĩ		- F	1	•		) Ad	t fe						98/08/14 + 0.00/00 + コメント 資格 azt 1 Metrequirement 「
、社日イ 2購日イ 2時日 20MMY3 回	17 17 17 17 17 17 17 17 17 17 17 17 17 1	(DI (DI (25		YYI YYI IA IA	) ) 1	- F	1	•		) Ad		ni ni	· · ·		tn.		<u>98/08/14</u> + 0.00/00 + art 1 <u>Metrenuirement</u> [ art 2
	17 17 17 17 17 17 17 17 17 17 17 17 17 1	(DI (DI (0) (0) (0) (0)				- K	1 2	•	•	Ad Ad Ec	Ini ini ion	ni ni om	· · · ·	ra:	tn De	P. P. gr	<u>98/08/14</u> + 0.00/00 + art 1 <u>Metrenuirement</u> 「 art 2 」

赤の線を移動して、画面の境界を変更できます。方法については、aXes のチュートリアルを参照してください。

- 6. 画面のタイトルをクリックして選択します。[aXesデザイナー]ウィ ンドウに画面のタイトルのプロパティが表示されます。
- 7. タイトルの[Visible]プロパティを[False]に設定します。

🔊 aXesデザイナ	Web ページダイアログ  🛛
💾 保存 😭 取消	i 🕕
この画面はカスタマ	イズされています。
選択されたオブジェ	つトの皆報
タイプ	出力
17	1
列	20
プロパティー	
4 基本	
style	
tabIndex	0
visible	False 🥒
enabled	True 🥖
tooltip	/

8. [aXesデザイナー]ウィンドウの上部にある[保存]ボタンをクリックし ます。



画面のカスタマイズが保管されたことを示すメッセージが表示されま す。



# タイトルは表示されません。

Sessi	on	1	Dis	pla	У	н	elp		 	 	 		
社員No.												:	A1234
t員姓									-			-	JACKSON
計昌名													STEDHEN

#### RAMP-TSAD04 ステップ2. スタイルのセットアップ

このステップでは、すべての画面のフォント・スタイルをセットアップ します。

スタイルを適用すると、共通の外観と操作性が得られ、制限のないスタ イルの発展が解消されるため、アプリケーションでは(手動による個別の 画面や要素のフォーマットとは対照的に)スタイルを使用する必要があり ます。また、スタイルによりシングル・ポイントでの変更ができるよう になるため、例えば、スタイルを変更してすべての画面のフォントを変 更することができます。

aXesを使用して自由にカスタマイズできますが、アプリケーションの外 観や操作性は標準のフレームワーク・テーマにより制御する必要がある こと、カスタマイズはこれらのテーマに適合する必要があることに注意 してください。

例えば、フレームワークがすべてのRAMP画面の背景色を自動的に設定 するため、画面の背景色のセットアップにaXesスタイルを使用してはい けません。

 [aXesデザイナー]ウィンドウの下部にある[アプリケーションのプロ パティーを見ます。]をクリックして、アプリケーションのプロパティ を表示します。



 プロパティを変更するには、[aXesデザイナー]ウィンドウの[アプリ ケーションのプロパティーを編集します。]をクリックします。



# 3. [スタイル]グループの[styles]をクリックします。

spaceBetweenColumns	0	
🕕 781h		
styles	0 items	~
エクステンション		

Stylesエディターが表示されます。



- 4. [追加]ボタンをクリックしてスタイルを追加します。
- スタイルにBasicFontという名前を付けます。
- [StyleFor]プロパティの値に[全て]を選択します。
- [htmlTag]プロパティはブランクのままにします。

選択された項目のプロパティー						
name	BasicFont	^				
styleFor	全て (1)					
htmlTag						
style						
theme						

- 5. [style]プロパティをクリックします。すべてのスタイル・プロパティ を示すウィンドウが表示されます。
- 6. [font-family]属性を見つけて、Verdanaを指定します。
- 7. [font-size]属性を見つけて、9ptを指定します。

🥫 スタイル・エディター	- style Web ページ ダイアログ	
ベース・スタイル		
このスタイルの基となるスタ	イルを選択します。	
[ペース・スタイルなし]		•
属性		
「 使用された属性のみま	示	
cursor		^
direction		
display		
empty-cells		
float		
font		
font-family	Verdana	
font-size	9pt	
font-size-adjust		
font-stretch		
font-style		
font-variant		
font-weight		
height		
imo-modo		

8. aXesデザイナーで[styles]をダブルクリックし、追加したスタイルを 表示します。

🔊 ሀスト・エディター	- styles Web ページ ダイアログ	X
😚 追加 💊 挿入	T and a	
項目		
BasicFont		<u>^</u>
		~
選択された項目のプロ	コパティー	
name	BasicFont	~
styleFor	<b>全</b> て	
htmlTag		
style	font-family:Verdana;font-size:9pt;	
theme		
for5250Attributes	0 items	0
		~

フォント・サイズを設定するときは、RAMPアプリケーションに表示される画面が使用可能なスペースに合わせて自動的にサイズが設定されることに注意してください。つまり、表示されるフォント・サイズは指定したサイズよりも小さくなることがあります。

10. [style]プロパティとは何かを理解するには、[aXesデザイナー]ウィ ンドウの[スタイル]見出しの前にある[i]アイコンをクリックします。

スタイル

11. [StyleFor]プロパティの説明を読みます。



BasicFontスタイルの[StyleFor]プロパティの値を[全て]に設定すると、 aXesで定義されているすべての画面のすべてのテキストにスタイルが適 用されます(aXes Designerで画面を編集して保管すると、aXesのscreens フォルダにscreen\_xxxxx.jsファイルが作成されます)。

12. [aXesデザイナー]ウィンドウの上部にある[保存]ボタンをクリック して、作成したスタイルを保管します。

🔊 aXesデザイ	(ナー Web ページ ダー	לם 🛛
💾 保存	取消	•
選択されたオブ	ジェクトの皆報	
タイプ	アプリケーション	
ブロパティー		
🤨 基本		<u>^</u>
defaultTheme	b	1
onApplication	Start	

# 以上で、画面のフォントがVerdana 9ptに設定されました。



RAMP-TSAD04 ステップ3. 頻繁に出てくる情報を非表示にする

このステップでは、従業員番号がフレームワークのインスタンス・リストおよびインスタンス・バーに表示され、この画面で変更できないため、画面の[従業員番号]フィールドを非表示にします。 5250画面を刷新するときは、以下の点を考慮する必要があります。

- 5250画面の識別子およびタイトルを非表示にします。代わりに、フレームワークのタブとヒントを使用します。これらの情報が5250画面に残ったままになると、基盤の画面が5250画面であることが明確になります。情報に含まれる値が何であるか常に確認してください。何もないか、少量の値しかない場合は、非表示にします。
- 画面の日時を非表示にします。[フレームワーク]ウィンドウには、
   日時が表示されます。
- インスタンス・バー、インスタンス・リスト、その他の場所にすで に表示されている繰り返しの情報やキー情報を非表示します。この場 合も、大規模なフレームワーク・コンテキストで頻繁に表示される情 報の値を確認します。
- アプリケーション・バージョンの詳細情報を非表示にします。[へ ルプ]メニューの[バージョン情報]オプションは、Windowsの場合と同 様にバージョン情報を確認できます。
- aXes画面のタイトル・バーやストライプを使用しません。これらの 場所には、ユーザーにとって多くの場合冗長で付加価値のない情報が 表示されます。
- 1. [aXesデザイナー]の上部にある[この画面のカスタマイズを開始しま す。]をクリックします。



2. 従業員番号ラベルを選択します。

■注肩約	A1234
社員姓	JACKSON
社員名	STEPHEN
住所1	6 Melissa Place

- 3. [Visible]プロパティを[False]に設定します。
- 4. 次に、フォント変更によりラベルを区切っている、ラベルの後の ドットを選択し、[Visible]プロパティを[False]に設定します。
- 5. 最後に、[従業員番号]フィールドの[Visible]プロパティを[False]に設 定します。
- 6. 画面のカスタマイズを保管します。

以上で、従業員番号ラベルとフィールドは画面に表示されなくなりま す。

#### RAMP-TSAD04 ステップ4. ツールチップの追加

このステップでは、[部門コード]フィールドのツールチップを追加します。ツールチップを使用すると、長いラベルやコンテキスト依存ヘルプ を画面に表示することができます。

- 1. [aXesデザイナー]ウィンドウで、[この画面のカスタマイズを開始し ます。]をクリックします。
- 2. [部門コード]フィールドを選択します。
- 3. [ツールチップ]プロパティのフィールドを見つけて次のテキストを コピーして貼り付けます。

このコードは、社員が現在勤務している部門のコードです。多くの場合、これは"DC"コードと呼ばれています。社員のIDバッジには、常にこのコードが表示されている必要があります。そうでない場合、会社構内への入構が拒否されることがあります。

選択されたオブジェクトの皆載							
タイプ	入力						
名前	utxtDe	epartment					
fī	10						
列	43						
プロパティー							
tabIndex		0		^			
visible		True	0				
enabled		True	0				
tooltip		このコードは、社員が現在勤活	0				
🌒 デフォルト	1						

- 4. 画面のカスタマイズを保管します。
- 5. [部門コード]フィールドにカーソルを置いて、ツールチップをテス トします。



# RAMP-TSAD04 ステップ5. ドロップ・ダウンの追加

このステップでは、米国の州を表示する[State and Country]フィールドと いうドロップ・ダウンを作成します。ドロップ・ダウンのデータは、 aXes eXtensionsに付属の静的テーブルから取得されます。

- 1. [aXesデザイナー]ウィンドウで、[この画面のカスタマイズを開始し ます。]をクリックします。
- 2. DisplayEmployee画面の[State and Country]フィールドを選択します。
- 3. aXes Designerでは、拡張がデフォルトのビジュアライゼーションが ドロップ・ダウンに変更されました。

エクステンション						
□ タイマ-	^					
□ チェックボックス						
▶ ドロップダウン						
□ ハイパーリンク						
フッシュボタン						
ラジオボタン						
□ 日付						
	*					

- 4. ドロップ・ダウンのdataSourceTypeを[静的テーブル]に変更します。
- 5. tableNameにUSStateを指定します。



- 6. 画面のカスタマイズを保管します。
- 7. ドロップ・ダウンをテストします。



固定値を入力するか、動的テーブルを使用して、ドロップ・ダウンに表示される値を設定することもできます。aXesチュートリアルでは、これを実行する方法の詳細を説明しています。

# RAMP-TSAD04 ステップ6. グループ・ボックス内のフィール ドの整理

このステップでは、フィールドをグループ・ボックスに入れることにより画面上のフィールドを論理的に整理します。

- 1. [aXesデザイナー]ウィンドウで、[この画面のカスタマイズを開始し ます。]をクリックします。
- 2. ドットを含んでいる画面上のすべての要素で[removeCUADots]プロ パティを[True]に設定します。



- 3. この変更を保管し、aXesデザイナーを編集モードに戻します。
- 2. 新しい要素を画面に追加します。



3. これを[グループ・ボックス]の拡張にします。

エクステンション		
「 デフォルト	^	
Frame		
□ Google <del>J</del> r-ト		
T HTML		
「 イメージ		
<u> クイックビック・メニュー</u>		
ダループ ボックス	~	

## 4. グループ・ボックスのキャプションを識別子にします。

🕛 グループ ボックス		
caption	識別子	Ø
style		0

## 5. サイズを以下のように初期設定します。

- 識別子	
社員姓	JACKSON
社員名	STEPHEN
住所1	6 Melissa Place

次に、フィールドのサイズを変更し、以下のようにグループ・ボックス内に移動します(グループ・ボックスの上部には少しスペースを残します)。

識別子		
社員姓	JACKSON	
社員名	STEDUEN	

フィールドを整列するときは、画面グラフ用紙を使用できます。

- 7. 画面上の空いているスペースをクリックします。
- 8. [aXesデザイナー]ウィンドウで[グラフ紙・線(開発者モードのみ)]オ プションを選択します。

エクステンション
<ul> <li>□ Auto Zoom Screen Size</li> <li>□ Subfile enhancer</li> <li>□ グラフ紙・線(開発者モードのみ)</li> </ul>
アクション
💱 新しいエレンC小を追加します。

# 次のように表示されます。



- 9. [グラフ紙・線(開発者モードのみ)]オプションをオフにします。
- 10. グループ・ボックスの[look]プロパティを[モダン]に変更します。

caption	識別子	0
style		0
ook	王文 <sup>N</sup> J	· 🥖 🗸
識別子		
識別子 社員姓	JACKSON	
識別子 社員姓 社員名	JACKSON STEPHEN	
<ul> <li>識別子</li> <li>社員姓</li> <li>社員名</li> <li>1注所1</li> </ul>	JACKSON STEPHEN	6 Melissa Place

- 11. この変更を保管し、aXes Designerを編集モードに戻します。
- 12. キャプションが[*Dates*]で、[look]プロパティが[モダン]である別のグ ループ・ボックスを追加します。
- 13. [識別子]グループ・ボックスの横に追加します。
- 14. 以下のように、グループ・ボックスに[入社日付]と[退職日付]を配 置します。

識別子		Dates
社員姓 社員名	JACKSON STEPHEN	入社日付(DDMMYY)         96/08/14         +           返職日付(DDMMYY)         0/00/00         +
住所1 住所2 住所3 自宅電話番号 卸門コード 課コード		6 Melissa Place West Pennant Hills (02) 9871 7773 SD + ES +

15. キャプションが[Contact and Location]で、[look]プロパティが[モダン] である別のグループ・ボックスを追加します(ここでも、画面の背景色 で囲まれるように、グループ・ボックスの周囲にいくらかスペースを 残します)。

# 16. [識別子]グループ・ボックスの下に配置し、資格の詳細フィールドを除く残りのフィールドを配置します。

	iepiaj iieip				<b>a</b> 1 7 14	
制子			Dates			
社員姓	JACKSON		入社日付 (四	DDMMYY)	96/08/14	+
社員名	STEPHEN		退職日付 (0	DDMMYY)	0/00/00	+
Contact and Loca	ation					
住所1 61	Melissa Place					
住所2 🛛 🛶	est Pennant Hi	lls				
住所3		× 2	2125			
自宅電話番号 (02	2) 9871 7773					
部門コード の	+	_				
	+					
BRUTE ES						
	1997 F.B					
〒「ES 取得日 (DDMMYY)	資格 コード	资格記述		JXXF		资格
〒1000000000000000000000000000000000000	資格 コード ADMIN1	资格記述 Administratn Pa	rt 1	コメント Met requirer	nent	资格
取得日 (DDMMYY) <u>98/03/25</u> 98/05/03	資格 コード ADMIN1 ADMIN2	資格記述 Administratn Pa Administratn Pa	rt 1 rt 2	コメント Met requirer	ment	
取得日 (DDMMYY) 98/03/25 98/05/03 98/05/05	資格 コード ADMIN1 ECD	資格記述 Administratn Pa Administratn Pa Economics Degr	rt 1 rt 2 ee	그メント Met requirer	nent	资格 ]

17. 最後に、キャプションが[資格]であるスキル情報のグループ・ボッ クスを追加し、その周りにいくらかのスペースを残します。

			Dates				
社員姓	JACKSON		入社日付	(DDMMYY)	96/08/14	+	
社員名	STEPHEN		波龍日付	(DDMMM)	0/00/00	+	
Contact and L	ocation						
住所1	6 Melissa Place		]				
住所2	Nest Pennant H	-ills	]				
住所3		~	2125				
自宅電話番号	02) 987 1 7773						
新門コード	5D +						
411円コード 第コード	5D + Eg +						
111円コード 第3ード	5D + Eg +						
111円コード () () () () () () () () () () () () ()	5D +						_
御門コード 課コード 総格 取得日	5D + 國 + 資格						_
御門コード 課コード (DDMMYY)	5D + 1日 1日 1日 1日 1日 1日 1日 1日 1日 1日 1日 1日 1日	夏格記述		144		業格	-
141円コード ( 取得日 (DDMMY) 98/03/	5D + E雪 + 夏午 コード 25 ADMIN1	資格記述 Administrat	n Part 1	コメント Mat require	ment	¥18-	
御門コード 課コード 取得日 (DDMMYY) <u>98/03/</u> 98/05/	250 + 25 + 25 ADMIN1 23 ADMIN2	資格記述 Administrat Administrat	n Part 1 n Part 2	コメント Met require	ement	¥18-	_
第1門コード 第コード (DDMIMY) 98/03/ 98/05/ 98/05/	\$50         +           \$25         +           \$26         +           \$27         K           \$27         K           \$28         +           \$27         K           \$27         K           \$28         +           \$27         K           \$28         +           \$27         K           \$28         +           \$27         K           \$27         K           \$28         +           \$27         K           \$28         +	資格記述 Administrat Administrat Economics	n Part 1 n Part 2 Degree	コメント Met require	em∉nt	¥.46	
第11円コード 第コード (DDMIMV) <u>98/03/</u> 98/05/ 98/05/ 98/05/ 98/05/	5D         +           5S         +           25         +           25         ADMIN1           03         ADMIN2           05         ECD           05         ENG	資格記述 Administrat Administrat Economics I English Deg	n Part 1 n Part 2 Degree pree	コメント Met require With disting	ement ction		+

# 18. 画面のカスタマイズを保管します。

RAMP-TSAD04 ステップ7.サブファイルへの上へボタンおよび 下へボタンの追加

このステップでは、サブファイルに複数のエントリーが存在することを 示すプラス記号を上へキーおよび下へキーに置き換えます。

1. サブファイルのプラス記号を選択します。



2. [Screens] タブで、moreindicatorという名前を付けます。

名:						
Chan	geMod	e				
3 <b>4</b> :						
						~
						Y
プファ	4L:	17.36	-	-		
16	144 I	21	17. 1/-	121	見出し	
8	21	R@F00003	1			~
						V
				-		
r-1	P*:		_	1	LOCKS	creen
_	216		17	70	Sfl	
-	27542					-
	Alle		18	7	V	-
	-			-		_
			18	16	Y	
			18	28	Y	
	Admir	nistratn Part 1				
			18	49	Y	
			18	73	٧	-
	-	- Contract	-	70		-
	Imoreir	noicator	41	79	Y	
	+			-		
	1		23	2		
	F1= /	Vルプ F3=終了	F4=PR	OMP	PT F12=	= R 🞽

3. 画面を保管します。



4. aXes Designerでは、+記号のデフォルトのビジュアライゼーションは オフです。つまり、表示されません。



5. 新しい要素を追加し、サブファイル・スクローラ・タイプにしま す。



6. *markerFieldName*にmoreindicatorを指定します。

🚺 サブファイル・スクローラー			
markerFieldName	moreindicator	0	
scrollerStyle	height:17px;width:17	px;	

- 7. サブファイル・スクローラの位置とサイズを設定します。
- 8. 画面のカスタマイズを保管します。
- 9. RAMP定義の部分的な保存を実行します。
- 10. サブファイルで上へボタンと下へボタンをテストします。

取得日	資格			
(DDMMYY)	3-8	資格記述	コメント	資格
98/05/0	2 HIS	History Degree	EXECELLENT	F
98/02/0	5 INTRO	Company Introduction	Met requirement	
98/02/0	5 KEY	Keyboard Skills	Met requirement	
98/03/1	5 MANAGE1	Management Course 1	Met requirement	

RAMP-TSAD04 ステップ8.ファンクション・キーを非表示に してピクチャーを追加する

このステップでは、画面の下部に表示されるファンクション・キー・テ キストを非表示にし、従業員のピクチャーを追加します。

aXesデザイナーで画面を編集するときは、RAMPにより前に非表示に設定されていた場合でも、画面上のファンクション・キーのテキスト行が表示されます。この行を非表示にするには、以下の手順を実行します。

- 1. aXesデザイナーで、[この画面のカスタマイズを開始します。]をク リックします。
- 2. 画面の下部にあるファンクション・キーのテキスト行を選択しま す。
- 3. [Visible]プロパティを[False]に設定します。

#### 次に、イメージを画面に追加します。

4. キャプションが[*Photo*]で、[look]プロパティが[モダン]であるグルー プ・ボックスを追加します。

識別子		Date	35		
社員姓	JACKSON	入社	B († (DD MMYY)	96/08/14	+
社員名	STEPHEN	退職	B († (DD MMYY)	0/00/00	] +
Contact and Lo	cation		Phot	0	
住所1 6	Melissa Place				
住所2 1	Vest Pennant H	ills			
住所3		2125			
自宅電話書号 🕜	02198717773		1		-
## <b>3</b> - K	D +				
課コード E	+				
資格					
取得日	資格				
(DDMMYY)	3-4	資格記述	コメント		資格
98/03/2	5 AD MIN 1	Administratn Part 1	Met require	ment	
98/05/0	3 AD MIN 2	Administratn Part 2			
98/05/0	5 EC D	Economics Degree			48
94/06/0	5 ENG	English Degree	With disting	tion	

5. 次に、新しい要素を追加して、グループ・ボックス内のイメージ拡

F1= ヘルプ F3= 桜了 F4=PROMPT F12= 取消し F14=MSG

## 張にします。

851T			Dates			
社員姓	JACKSON		入社日付 ()	DDMMYY)	96/08/1	4 +
社員名	STEPHEN		退職日付 ()	DDMMYY)	0/00/0	0 +
ontact and Lo	cation			Photo		
生所1 6	Melissa Place				~	
生所2 🗸	est Pennant H	tills				
主所3		♥ 2	1 25			
						-
自宅電話書号 🛛	121 9871 7773				T.	1.1
白宅電話者号 🕜 制用コード 🔤	2)98717773					
自宅電話番号 (( 利用コード s 舞コード p	2198717773 D + J+					
自宅電話書号 () 第日コード S 集コード E	02)98717773 D + g+					
自宅電話番号 (() 部門コード (s) 乗コード (g) 税	02)98717773 D + g+					
自宅電話番号 (1) 第1日コード (5) 乗コード (5) 橋 校得日	12198717773 D + g+ 資格					
自宅電話音号 (() 第二コード S 東コード E 橋 校祥日 DDMMYY)	12198717773 D + g+ g+ 資格 コード	資格記述		1/2/2		<b>資格</b>
自宅電話 중국 (( ) 배리 그 ~ ド ( ) 및 그 ~ ド ( ) 및 및 드 ~ ド ( )	12195717773 □ + g+ <b>資格</b> □ - ド 5. ADMIN1	資格記述 Administrato Pau	rt 1	コメント Mat require	ment	jan and a state of the state o
自宅電話 중국 (初 期一 그 - ド (日 東 그 - ド (日 東 그 - ド (日 朝 御 (明 日 (日) (日) (日) (日) (日) (日) (日) (日) (日) (	2195717773 D + g+ g+ <b>資格</b> □-K 5 <u>ADMIN1</u> 3 <u>ADMIN2</u>	資格記述 Administrath Par Administrath Par	rt 1 rt 2	그것가 Mat require	ment	jetă
自宅電話書号 (() 第一コード () 第コード () 第コード () 第二、ド () 第二、ド () 98/03/2 98/05/0 98/05/0	2195717773 D + 子 学 本 ADMIN1 3 ADMIN2 5 ECD	資格記述 Administrath Par Administrath Par Economics Degri	rt 1 rt 2	אלא ב Met require	ment	je te

6. imagePathを/ts/skins/images/に、imageNameをexamplephoto.gifに変更します。

🕕 イメージ		
style		
imagePath	/ts/skins/images/	
imageName	examplephoto.gif	
onClick		

- 7. [保存]をクリックします。イメージがフル・サイズで表示されま す。
- 8. 縮小するには、イメージ拡張の[Style]プロパティをクリックし、ス タイルの高さと幅を100%に変更します([イメージ]グループの[Style]プ ロパティを慎重に選択します)。

選択されたオブ	ジェクトの情報	
タイプ	入力	
プロパティー		
<ol> <li>基本</li> </ol>		^
name		
style	left:470px;top:110px;wi	dth:1
type	入力	
tabIndex	0	=
visible	True	0
enabled	True	0
tooltip		1
🕕 イメージ		
style	width:100%;height:1009	%; 🥖
imagePath	/ts/skins/images/	0

# このとき、画面は次のようになります。

0.001			Dates			
社員姓	JACKSON		入社日	+ (DDMMYY)	96/08/1	4 +
社員名	STEPHEN		速驟日(	+ (DD MMYY)	0/00/0	0 +
Contact and	Location			Phot		
住所1	6 Melissa Place		]		100	ALC: NOT THE OWNER OF
住所2	West Pennant H	lills			1 100	1000
住所3		~	2125			6 X
					1	2/202
自己有效不可	(02)9871 7773	1				
白毛電話書号	(02) 9871 7773					-
	(02) 9871 7773 SD + Ed +					
自宅電話査국 웨門그~ド 屛그~ド	(02) 9871 7773 SD + ES +					
自宅電話香号 斜門コード 緑コード 御福	(02)98717773 SD + ES +					
自宅電話者号 斜門コード 課コード 武橋 泉澤日	(102) 9571 7773 SD + ES +					
自宅電話書号 新門コード 康コード 御橋 東得日 (DDMMYY)	(102)95717773 SD + ES + 資格 コード	資格記述		אנאב		je Ka
日宅電話名号 第一コード 展コード (DDMMYY) 98/03	(02)95717773 SD + ES + 資格 コード /25 ADMIN1	<b>資格犯過</b> Administrati	n Part 1	אלאב Mat require	ment	
日宅電話名号 第日コード 森コード 資格 泉澤日 (DDMMYY) <u>98/03</u> 98/05	(02)95717773 SD + ES +	<b>資格犯述</b> Administratu Administratu	n Part 1 n Part 2	عريد Mat require	ment	
そる林家 珍白 メービ 同時 メービ 同時 メービ 同時 メービ 同時 第二 ド ジード 第二 ド ジード 第二 ド ジード (DDMMYY) 95/03 95/05 95/05	(02) 9571 7773 SD + ES +	<b>資格犯述</b> Administratu Administratu Economics 1	n Part 1 n Part 2 Degree	عريد Mat repuire	ment	jan

# 9. 画面のカスタマイズを保管します。
### RAMP-TSAD04 ステップ9. ハイパーリンクの追加

このステップでは、サーバーにあるPDFドキュメントを開くハイパーリ ンクを画面に追加します。

- 1. aXesデザイナーで[この画面のカスタマイズを開始します。]をク リックし、画面を編集モードにします。
- 2. 新しい要素を画面に追加します。

アクション			
💋 新しい3	ロレメントを追加しま	₫.	
[ LAN	VF_T	RAM	🚺 koma

- 3. これをハイパーリンク拡張にします。
- 4. 以下のようにサイズと位置を設定します。

nex Hea	JACKSON	2	社日付 (DDMMYY)	96/08/14 +
ALE O	STEPHEN		State 14. (DDHHHLL)	0/00/00 +
Contact and I	ocation		Phote	0
住所1	6 Melissa Place			and the second second
住所2	West Pennant H	Hills		( March
住所3		212	25	000
自宅電話書号	(02) 9871 7773			
				and the second se
웨티그-K	SD +			
第 <b>日</b> コード 第コード	SD +	<u>55</u>		
#Fコード ほコード		<u></u>		
新用コード 舞コード 観櫓 取得日	SD + ES + 資格	<u></u>		
第一コード 第コード 取得日 (DDMMYY)	sp + Es + 資格 3~ド	以上		řě.
#月コード 課コード (DDMMYY) 98/03	SD + ES + 直一下 (25 ADMIN1	検格化述 Administratn Part : Administratn Part :		ÉR.
第日コード 第コード 武裕 (DDMMYY) 95/03 98/05 98/05	SD + ES + D - K 25 ADMIN1 03 ADMIN2	<b>検疫12点</b> Administratn Part : Administratn Part : Fenomic Deares	1 Matrauire	
第日コード 東コード 取得日 (DDMMYY) <u>98/03</u> 98/05 94/05	SD + ES + D - K 2- K 25 ADMIN1 03 ADMIN2 05 ECD 05 FING	検格12.2 Administratn Part 1 Administratn Part 1 Economics Degree English Degree	1 Met require	

5. キャプションをEmployment Contractに設定します。

6. onClickプロパティを以下のように変更します。

window.open('/ts/skins/images/examplecontract.pdf', '\_blank');

- 7. 画面のカスタマイズを保管します。
- 8. 画面の[Employment Contract]ハイパーリンクをクリックします。

ブラウザ・ウィンドウが表示され、サーバーにある契約書PDFが表示されます。



RAMP-TSAD04 ステップ10. 再設計した画面のテスト

このステップでは、再設計した画面をテストし、フレームワークのテー マが機能することを確認します。

- 1. RAMP-TS 5250エミュレータでログオフします。
- 2. フレームワークを再起動し、aXesを起動します。
- 3. 人事アプリケーションで、従業員Veronica Brownを表示します。

詳細コマンド・ハンドラーは、以下のようになります。

🛔 Employees	::詳編(A0070-E	ROWN VERONICA)			
🔜 詳細 🔒	イベント 📄 ドキュ:	わとと			
					保存
歳別子			Dates		
社員姓	BROWN	/	、社日付 (DDMMYY)	90/01/28 +	PROMPT
社員名	VERONICA	, i	乱職日付 (DDMMYY)	0/00/00 +	クリップボードへコピー
Contact and I	Location		Photo	0	Send Signal to Filter
住所1	12 Railway Stree	t		Call Too	
住所2	Baulkham Hills			A MERIA	MSG
住所3		¥ 215	53	Core -	メニューバーを表示
自宅電話番号	(02) 9609 4627				スナップショットを表示
課コード		ployment Contract		30	トドキュメンテーション
				Not No.	トレース オン
藏格					10 647
取得日	资格				スターンの調査
(DDMMYY)	コード	资格記述	コメント	資格	
96/06	/30 ADMIN1	Administratn Part	1 Met require	ement F	
98/03	/01 ADMIN2	Administratn Part	2	F	-
95/12	10 ADVPGM	Advanced Program	nming	F	E
00/05	/04 COM	Communications E	)e are		E

 Windowsメニューの[全体のテーマ]オプションを使用して、フレーム ワークのテーマを変更し、各種のテーマによるカスタマイズの外観を 確認します。

2007 Oliveのテーマでの画面の外観は以下のとおりです。

読月子       Dates       A注目付(DDMMYY)       90/01/28 +       PROMPT         社員谷       VERONICA       別日付(DDMMYY)       0/00/00 +       別ップボードへコピー         Contact and Location       Photo       Send Signal to Filter         住所1       12 Railwav Street       2153       Photo       Send Signal to Filter         住所3 <ul> <li>2153</li> <li>9 年後話番号</li> <li>(02) 9609 4627</li> <li>第門コード</li> <li>INF +</li> <li>理コード</li> <li>D,+</li> <li>Employment Contract</li> <li>アント</li> <li>文格</li> <li>(DDMMYY)</li> <li>12 - F</li> <li>交格</li> <li>(DDMMYY)</li> <li>12 - F</li> <li>交格記述</li> <li>12 X2/F</li> <li>※2153</li> <li>22 153</li> <li>12 153</li> <li>12 153</li> <li>153</li> <li>154</li> <li></li></ul>	🗔 詳細 🛛 🛕 イベント 📄 ドキュメ	,ト				
社員姓         BROWN         入社日付 (DDMIMYY)         90/01/28 +         PROMPT           社員名         VERONICA         週間日付 (DDMIMYY)         0/00/00 +         グリックボードへコピー           Contact and Location         Photo         Send Signal to Filter           住所1         12 Railwav Street         (DDMIMYY)         0/00/00 +         グリックボードへコピー           住所3         ●         2153         ●         ●         ●           自宅電話番号         (02) 9609 4627         ②153         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●         ●	識別子	Date:	5			保存
社員名 VERONICA 退配日付(DDMMYY) 0/00/00 + の小ブボードヘコピー Contact and Location 住所1 12 Railwav Street 住所2 Baulkham Hills 住所3 ● (02) 9609 4627 部門コード INF + 課コード D + Employment Contract 文格 (DDMMYY) コード 交格記述 コメント 交格 96/06/30 ADMIN1 Administrath Part 1 Met requirement f 98/03/01 ADMIN2 Administrath Part 2 「 C C C C C C C C C C C C C C C C C C	社員姓 BROWN	入社日	付 (DDMMYY)	90/01/28 +		PROMPT
Contact and Location       Photo       Send Signal to Filter         住所2       Baulkham Hills       2153       「       「       M5G         自宅電話番号       (02) 9609 4627       2153       「       「       M5G         部門コード       INF +       T       INF +       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T       T <th>社員名 VERONICA</th> <th>退職日</th> <th>付 (DDMMYY)</th> <th>0/00/00 +</th> <th></th> <th>クリップボードへコピー</th>	社員名 VERONICA	退職日	付 (DDMMYY)	0/00/00 +		クリップボードへコピー
住所1 住所2 自宅電話番号 (02) 9609 4627 部門コード INF + 課コード ①+ Employment Contract 変格 (DDMMYY) コード 変格記述 コメント 変格 96/06/30 ADMIN1 Administrath Part 1 98/03/01 ADMIN2 Administrath Part 1 98/03/01 ADMIN2 Administrath Part 2 95/12/10 ADVPGM Advanced Programming 98/05/04 COM Communications Degre	Contact and Location		Photo			Send Signal to Filter
住所2 住所3 自宅電話番号 (02) 9609 4627 部門コード INF + 課コード D\ + Employment Contract 変格 (DDMMYY) コード 変格記述 コメント 変格 96/06/30 ADMIN1 Administrath Part 1 Met requirement f 98/03/01 ADMIN2 Administrath Part 1 Met requirement f 98/03/01 ADMIN2 Administrath Part 2 95/12/10 ADVPGM Advanced Programming 98/05/04 COM Communications Degre	住所1 12 Railwav Street			ALCON.		
住所3 自宅電話番号 「02) 9609 4627 部門コード INF + 課コード D、+ Employment Contract 近格 変格 (DDMMYY) コード 変格記述 コメント 変格 96/06/30 ADMIN1 Administrath Part 1 Met requirement F 98/03/01 ADMIN2 Administrath Part 1 Met requirement F 98/03/01 ADMIN2 Administrath Part 2 F 95/12/10 ADVPGM Advanced Programming 98/05/04 COM Communications Degre	住所2 Baulkham Hills			A CONTRACTOR		MSG
自宅電話番号       (02) 9609 4627         部門コード       INF +         課コード       D) +         医からの       Employment Contract         資格       トドキュメンテーション         取得日       没格         (DDMMYY)       コード         マード       没格         96/06/30       ADMIN1         Administratn Part 1       Met requirement         98/03/01       ADMIN2         Administratn Part 2       「「」」」         95/12/10       ADVPGM         98/05/04       COM         Communications Degre       「」」」	住所3	2153		100		バーを表示
課コード D + Employment Contract 資格 取得日 没格 (DDMMYY) コード 没格記述 コメント 没格 96/06/30 ADMIN1 Administrath Part 1 Met requirement F 98/03/01 ADMIN2 Administrath Part 2 F 95/12/10 ADVPGM Advanced Programming 98/05/04 COM Communications Degre	自宅電話番号 (02) 9609 4627 部門コード TME +	]		DEL		スナップショットを表示
資格 取得日 没格 (DDMMYY) コード 没格記述 コメント 资格 96/06/30 ADMIN1 Administratn Part 1 Met requirement F 98/03/01 ADMIN2 Administratn Part 2 F 95/12/10 ADVPGM Advanced Programming F 98/05/04 COM Communications Degre [ [ ]		loyment Contract		100		ドキュメンテーション
取得日 資格 (DDMMYY) コード 资格記述 コメント 资格 96/06/30 ADMIN1 Administratn Part 1 Met requirement F 98/03/01 ADMIN2 Administratn Part 2 F 95/12/10 ADVPGM Advanced Programming F 98/05/04 COM Communications Degre [ ]	資格					トレースオン
(DDMMYY)       コード       資格記述       コメント       資格         96/06/30       ADMIN1       Administratn Part 1       Met requirement       F         98/03/01       ADMIN2       Administratn Part 2       F         95/12/10       ADVPGM       Advanced Programming       F         98/05/04       COM       Communications Degre       F	取得日 资格					スタリーンの調査
96/06/30       ADMIN1       Administratn Part 1       Met requirement       F         98/03/01       ADMIN2       Administratn Part 2       F         95/12/10       ADVPGM       Advanced Programming       F         98/05/04       COM       Communications Degre       F	(DDMMYY) J-F	资格記述	コメント	ÿ	资格	
98/03/01     ADMIN2     Administratn Part 2     F       95/12/10     ADVPGM     Advanced Programming     F       98/05/04     COM     Communications Degre     F	96/06/30 ADMIN1	Administratn Part 1	Met requirem	nent	F	
95/12/10 ADVPGM Advanced Programming F	98/03/01 ADMIN2	Administratn Part 2			E	
98/05/04 COM Communications Degre	95/12/10 ADVPGM	Advanced Programming			65	
	98/05/04 COM	Communications Degree			[ E	

aXesグループ・ボックスは、すべて青みがかった階調を使用していることに注意してください。これらは、VLテーマに従うように変更することもできます。グループ・ボックスで色の階調を設定する方法の詳細については、aXes USERENVファイルを参照してください。aXesスクリプトで現在のVLテーマを確認する方法の詳細については、「Axesチュートリアル 10」を参照してください。

## RAMP-TSAD04 ステップ11. 画面のカスタマイズの削除

このステップでは、screen\_xxxx.jsファイルの名前を変更して、aXes画面のカスタマイズを削除します。その後、ファイルの名前を元に戻して拡張を再適用することができます。

- 1. フレームワークを閉じます。
- Windowsエクスプローラで、axes\ts\screens\ディレクトリを見つける か(ユーザーはサーバーのIFSドライブにマッピングされている必要が あります)、使用している場合はプライベート定義フォルダを見つけま す。
- 4. 次に、screen\_displayemployee.jsファイルを見つけて、名前を screen\_displayemployeeX.jsに変更します。

screens								
ファイル(E) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)								
🔇 戻る 🔹 🌍 🔹 🏂 🎾 検索 💫 フォルダ 🏢・								
アドレス(D) 🛅 Z:¥axes¥ts¥screens			*					
ว <del>ร</del> มชี ×	名前 🔻	サイズ 種類	更新日時					
🕀 🛅 fogi	Screen_displayemployee Ss	12 KB JScript Script File	2010/07/23 10:59					
🗉 🧰 images	Extension_axTimer.js	6 KB JScript Script File	2009/11/18 13:08					
표 🛅 install	😹 Extension_axSubfileScroller.js	8 KB JScript Script File	2009/11/18 13:08					
🗄 🛅 logs	😹 Extension_axSubfileEnhancer	10 KB JScript Script File	2010/01/22 20:10					
표 🛅 samples	😹 Extension_axSimpleStripe.js	6 KB JScript Script File	2009/11/18 13:08					
🖃 🧰 ts	Extension_axSimpleLine.js	6 KB JScript Script File	2009/11/18 13:08					
표 🫅 admin	🕼 Extension_axSimpleBox_x.js	2 KB JScript Script File	2009/11/18 13:08					
🗉 🧰 dev	🖉 Extension_axSimpleBox.js	2 KB JScript Script File	2010/01/22 20:10					
🕀 🫅 lang	🕼 Extension_axRadioButton.js	18 KB JScript Script File	2009/11/18 13:08					
🖃 🧰 screens	Extension_axQuickPick.js	13 KB JScript Script File	2009/11/18 13:08					
🕀 🦳 axes01	😹 Extension_axMessageHandler	8 KB JScript Script File	2009/11/18 13:08					
표 🫅 demo	😹 Extension_axLabel_x.js	3 KB JScript Script File	2009/11/18 13:08					
표 🦳 eeva	😹 Extension_axLabel.js	3 KB JScript Script File	2010/01/22 20:10					
🗉 🧰 RAMP_Izumi	😹 Extension_axImage.js	3 KB JScript Script File	2009/11/18 13:08					
🗉 🦳 rts_demo	😹 Extension_axHyperLink.js	5 KB JScript Script File	2009/11/18 13:08					
🕀 🧰 zzz01	😹 Extension_axHTML.js	3 KB JScript Script File	2009/11/18 13:08					
	Extension_axGroupBox.js	5 KB JScript Script File	2009/11/18 13:08					
🗉 🧰 skins	Extension_axGraphPaper.js	6 KB JScript Script File	2009/11/18 13:08					

- 5. フレームワークを再起動します。
- 6. 画面のカスタマイズが削除されていることを確認します。

この拡張を再適用する場合は、ファイル名からXを削除します。 また、aXes Designerで[保存]ボタンをクリックするたびに、カスタマイ ズされた画面のコピーが、screensディレクトリのサブフォルダ ScreenVersionsまたはプライベート定義フォルダに Screen\_xxxxxxxxx\_YYYYMMDD\_HHMMSS\_mmmmmm.jsのような名前で保管されることにも注意してください。

必要なバージョンを見つけて、既存のScreen\_xxxxxxxxx.jsファイルを削除し、画面バージョン・ファイルを定義セット・フォルダ内にコピーすることで、以前の設計に戻すことができます。これに、

Screen\_xxxxxxxxx.jsという名前を付けます。

これを実行する前に、必ずすべてのaXes開発者セッションを終了してく ださい。

#### まとめ

aXesを使用して画面を再設計する場合は、aXesのホーム・ページからア クセスできるaXesチュートリアルを終了しておく必要があります。

ブラウザからaXesを起動するには、

URL、http://hostname:80/wba/home.htmlにアクセスします(hostnameには、 ユーザーのホストの名前を指定するか、必要によりデフォルトのポート 80に変更します)。aXesのホーム・ページの右側にある[チュートリアル]リンク をクリックし、[Tutorial 0 - はじめに]リンクをクリックします。 重要なポイント

- 画面の拡張を作成するには、代替の方法がある場合があります。例 えば、特別なフィールド・ハンドラーを使用したり、aXesで日付拡張 にしたりすることで、日付フィールドをカレンダー・ドロップ・ダウ ンとして表示することができます。
- aXesのホーム・ページからaXesを使用して、RAMPツールを使用し ないで画面に名前を付けたり、再設計したりすることができます。画 面の分類やスクリプト作成には、RAMPツールを使用する必要があり ます。

ヒント

- aXesはテーマもサポートします。これらは、VLFテーマに直接マッ ピングすることができます。つまり、aXesでキー・テキストを強調 し、太字にするロール・ベースのスタイル"KeyDetails"を定義した場 合、実際には、VLFテーマBlueの場合、テキストの色を濃い青、テー マSilverの場合、テキストの色を黒、テーマOliveの場合、テキストの 色を濃い緑としてスタイルをテーマ化することができます。
- 明るい原色や大型のフォントまたは装飾的フォントは使用しないで ください。大型のフォント、装飾的フォント、および明るい原色を使 用しても拡張された5250画面のWebページで機能することもありま す。ただし、VLFではより低品質のキー画面が生成される傾向があ り、これらの種類のフォントは不必要に目立って、多くの場合非常に 不適切になります。

### 覚えるべき事柄

 aXes eXtensionsを使用してRAMPアプリケーションの画面を拡張す る方法

## RAMP-TSAD05:SHAREDプロパティおよびファンクション を使用する

RAMPスクリプトを開始すると、ロジックでパターンや繰り返しが使用 されていることがわかります。このロジックを共有オブジェクト内に移 動することで、任意のRAMPスクリプトからロジックを呼び出すことが できます。この機能を使用すると、再利用とメンテナンス面で利点があ ります。

目的

#### 共有プロパティおよびファンクションを使用する方法を学習する



この目的を達成するには、以下のステップを完了してください。 参照:共有スクリプトについて RAMP-TSAD05 ステップ1.オプション - 共有スクリプト・ファイルの ユーザー独自のコピーを作成する RAMP-TSAD05 ステップ2.SHAREDプロパティおよびファンクションに アクセスする

RAMP-TSAD05 ステップ3. 独自のSHAREDプロパティを作成する RAMP-TSAD05 ステップ4. 独自のSHAREDファンクションを作成する まとめ

始める前に

このチュートリアルを完了するには、コア・チュートリアルRAMP-TS001〜RAMP-TS015が完了している必要があります。 共有スクリプトについて

共有スクリプト・ファイル、uf\_sy420\_rts.jsを使用して、すべての5250画 面スクリプトからアクセスできる共通のJavaScriptプロパティおよびファ ンクションを保管できます。

通常、このファイルはRAMP-TSのskinsフォルダに保管されます。ただ し、このファイルのプライベート・バージョンは、指定したプライベー ト定義フォルダにも配置できます(「RAMP-TSAD05 ステップ1.オプショ ン-共有スクリプト・ファイルのユーザー独自のコピーを作成する」を 参照してください)。

共有スクリプト・ファイルについて確認するには、以下を実行します。

- 1. Windowsエクスプローラで、\axes\ts\skinsフォルダにアクセスできる ようにマッピングされたドライブを設定します。
- 2. メモ帳またはテキスト・エディターを使用し、uf\_sy420\_rts.jsファイルを見つけて開きます。次のように表示されます。

/\* \_\_\_\_\_\_

/\* Note that this file is used when using RAMP-TS as the RAMP 5250 server \*/

/\* This file is for common JavaScript properties and functions you want to accest /\* from all your 5250 screen scripts. To provide an unlimited name space your /\* properties and functions MUST be encapsulated inside an object named SHAI /\* Typically is reside in the \axes\ts\skins folder \*/

/\* \_\_\_\_\_\*/

/\* The SHARED object contains all customer defined shared scripts and properti /\* ------ \*/

```
var SHARED =
```

/\* \_\_\_\_\_ \*/

/\* Properties defined as part of the shared object \*/

/\* \_\_\_\_\_ \*/

myProperty1 : "a",

myProperty2:42,

```
/* ______ */
/* Functions defined as part of the shared object */
/* ______*/
/* myFunction1 is a test function */
myFunction1 : function(a,b,c)
{
 alert("myFunction1 executed with parameters " + a.toString() + " " + b.toStri
 return:
}, /* <====== Note the comma ======= */
/* myFunction2 is another test function */
myFunction2 : function(a,b)
{
 var sResult = "myFunction2 was executed with parameters " + a.toString() +
 return(sResult);
}, /* <====== Note the comma ======= */
/* Dummy last property that does not have a comma, leave here. All preceedin
```

myEndProperty : true

}; /\* End of SHARED object definition \*/

このファイルの構造は以下に示すように簡素です。

- var SHARED =の行は、SHAREDという名前のJavaScriptオブジェクトの始まりを定義します(SHAREDという名前を使用する必要があります)。
- SHAREDオブジェクト内には、myProperty1とmyProperty2という2つのプロパティがあります。
- また、それぞれが3つおよび2つのパラメータを受け取る myFunction1とmyFunction2という2つのファンクションがあります。
   これらのプロパティとファンクションは、SHAREDオブジェクト内で定

義される方法を示す以外に何も機能はありません。特に、コンマを使用 してファンクションを区切ることを示しているコメントに注意してくだ さい。

このオブジェクト・フォーマットは純粋なJavaScriptです。これは、 RAMP独特のものではありません。

この技術を使用して、他と競合しないコード用に保持されたnamespaceを 作成します。 RAMP-TSAD05 ステップ1.オプション - 共有スクリプト・ファ イルのユーザー独自のコピーを作成する

通常、共有スクリプトはRAMP-TS skinsフォルダにUF\_SY420\_RTS.JSと いうファイル名で格納されます。ただし、プライベート定義フォルダに SHAREDスクリプトファイルのプライベート・バージョンを作成するこ ともできます。

注:クラスルーム設定でこれらのチュートリアルを終えると、どのユー ザーにもプライベート定義フォルダが設定されます。ただし、実際のプ ロジェクトではこれを行うべきではありません。プロジェクト・フォル ダで完了した作業を別のプロジェクト・フォルダで完了した作業とマー ジすることはできないため、プロジェクトはプロジェクトごとを基準に 設定する必要があります。複数の開発者が同じ定義セットを使用して同 じプロジェクトで作業することはよくあります。

プライベート定義フォルダに共有スクリプトファイルのプライベート・ バージョンを作成するには

- 1. Windowsエクスプローラで、\axes\ts\skinsフォルダにアクセスできる ようにマッピングされたドライブを設定します。
- 2. uf\_sy420\_rts.jsをコピーし、プライベート定義フォルダ \axes\ts\screens\MyPrivateDefinitionFolderに貼り付けます。

3. フレームワークの[サーバーの詳細]で、[RAMP-TS (ターミナル サーバー)]グ ループ・ボックスの[共有オブジェクトを含む]オプションをオンにし、 SHAREDオブジェクト・ファイルがプライベート定義フォルダにあるこ とをRAMP-TSに示します。

配布サーバーとして保存		
IP アトシス	VLFTEAM	
ホ"─卜番号	8080	
実行モート・ロート・ハウス	/ts/skins/	📄 HTTPS を使用
RAMPツールモート፣ ロート፣ハ*ス	/ts/dev/	HTTPS を使用
プライベート定義フォルダ	eeva	✓ 共有わジェ外を含む
RAMP-TS	ソールのインストールと構成をテスト	

4. フレームワークを保存します。

RAMP-TSがこのファイルを認識するためには、これが\*PUBLIC \*Rユー ザー権限のみを持っていることを確認する必要があります。これを確認 するには、

5. iSeriesで、WRKLNKコマンドを使用します。

MAIN SYSTEM I Xインメニュー		
次の 1. つを選択してください。	システム:	AS1002
1. ユーザー・タスク		
2 · オフィス・ダスジ 3 · 汎用システム・タスク		
4.ファイル,ライブラリー,およびフォルダー		
5. プログラミング 5. 通信		
2. 23 18 7. システムの定義または変更		
8. 問題処理		
9. メニューの表示 10. 協毎援助オブション		
11. SYSTEM I ACCESS のタスク		
90 # インオフ		
50, 94 9 <b>4</b> 9		
選択項目またはコマンド		
===> wrkink		
F3=終了 F4=ホ*ワ]ホ*n F9=テu]n*の複写 F12=取り消し	F13= 情報援	助
F23= 初期 xpや~ の設定		

6. プライベート定義フォルダに移動し、UF\_SY420\_RTS.JSの権限を表示します。

オ	ブジェクト・	リンクの処理			
ディレクトリー・・・・ /axe	s/ts/screens	;/eeva			
オブションを入力して,実行す 2=編集 3=コピー 4= 11=現行ディレクトリーの39	⊧ーを押して∘ 除去 5= 表 【更	ください。 示 7=名前	の変更	8= 属性の表	示
OPT e#*1*phn,x]h	jb‡°	属性	テキスト		
screens.jsn	STMF				
5 ur_syr20_rcs.js	SIMP				
					錦 わり
パラメーターまたはコマンド					
 F3= 終了 F4= ブロンプト	F5 = 最新表示	π F9=7u]n'	の複写	F12= 取り》	首し
F17= 位置指定 F22=#?~モn <sup>*</sup>	全体の表示	F23=e\$°1	1] 続き		

7. \*PUBLICユーザーのみが\*R権限を持っていることを確認します。

							権	眼のタ	⊥理		
オブ	ジェクト							/axes	/ts/s	creens/e	eva/uf s
DA.	7							STMF	,,	ereens, e	ere/er_s
所有	#							OSECO	FR		
1 2	ガループ							*NONE			
梅限	HZF.							*NONE			
142 143											
オフ	ションを	スカー	. 7.	実行	i ‡ -	一方	- #甲	してく	ださし	١.,	
1=	ユーザー	O iB	hΠ	2=	ц – т	+f	— 擁	:B尼のス	रंक	4= ユ —	サーの除
			<del>,</del> -	々	_	-	オフ	ジェク	7 ト 権[	眼	
OPT	ューザ	_	権即	₹		存す	τ Έ	MGT	変更	REF	
							-				
	*PUBLIC		*R								
	OSECOFR		*RWX			х		x	х	х	
	AXES		*RWX			х		x	x	х	
R 5	v = /2 =	± 7-1	t n v	D B							
		6. (E 1)	8 4 1								
F3=	这了 F4	4- 7	<u>n 22</u>	7 ト		F	5 - 1	昆新主	<u></u>	F9=īuln	2 山畑玉
E11_	ニーク地		E¥ SEL	 	-	E1	12_	BD LIS	ж. Н	E24- +	一の結手
LTT=	ノーン 相	E MC O)	a于 4回 1	27 <del>at</del> e 3	11	Γ.	12=	4X (D.)	H U	T24= +	- 00 mc 8

以上で、共有スクリプト・ファイルの個人用のコピーの使用を開始する準 備が整いました。 RAMP-TSAD05 ステップ2. SHAREDプロパティおよびファン クションにアクセスする

このステップでは、共有プロパティおよびファンクションにアクセスす る方法を学習します。

1. 共有スクリプト・ファイルで2つのテスト・プロパティを見つけて、 その内容を確認します。

```
/* ---- */
/* The SHARED object contains all customer defined shared scripts and properties
 */
var SHARED =
{
    /* ------ */
    /* Properties defined as part of the shared object */
    /* ------ */
    myProperty1 : "a",
    myProperty2 : 42,
```

2. 選択した5250 DestinationフォームのRAMPスクリプトを開き、以下 のコードをvHandle\_ARRIVEファンクションに追加してプロパティの 値を表示します。

alert(SHARED.myProperty1 + " " + SHARED.myProperty2);

コードは以下のようになります。

else {

window.alert(SHARED.myProperty1 + " " + SHARED.myProperty2);

SHOW\_CURRENT\_FORM(true); /\* Show the form in the framework and show VLF buttons \*/ GET\_FORM\_MESSAGE(22); /\* Extract messages and hide the message line \*/ HIDE\_5250\_BUTTONS(); /\* Hide any 5250 style buttons displayed \*/ SETBUSY(false); /\* Last thing done - turn off the busy state \*/ }

/\* <ARRIVE /> - Do not remove or alter this line \*/

変更をコミットし、[保存]を実行します。次に、従業員の詳細を表示して到着スクリプトを実行します。以下のようなメッセージ・ボックスが表示されます。

	X		
- 名前 - By Date of Birth - By Salar	У	😳 🗆 🔺 🖻 🗎	
▼ リストの消去	検索	Employees	記述
2+ 自加生		A0070	BROWN VERONICA
R		A0090	BLOGGS FRED JOHN ALAN
5		A1031	BLAKE JOHN
RAMPスクリーンヘジクナルを送る		A3564	BROWN FREDDY
<			
		Web ページからのメッ	t 💌
🔮 Employees:詳編(A0070-BROWN)	VERONICA)		
🔲 詳細 🛕 イベツト 📄 ドキュメント		A = 12	
		A 42	
		( The second sec	
			UK
		2	

このメッセージ・ボックスは、SHARED.myProperty1プロパティと SHARED.myProperty2プロパティの値を表示しています。

4. 共有スクリプト・ファイルでテスト・ファンクションFunction1を見つけて、その内容を確認します。



5. このコードを到着スクリプトに追加します。

SHARED.myFunction1("1",2,"3");

6. 変更をコミットし、[保存]を実行します。次に、別の従業員の詳細 を表示します。以下のように別のメッセージ・ボックスが表示され、 SHAREDオブジェクトでmyFunction1ファンクションを実行したことが 示されます。

	X		
- 名前 - By Date of Birth - By Salary	/	📀 🗆 🔺 🖻 🗎	
▼ リストの消去	検索	Employees	記述
計 自动生		A0070	BROWN VERONICA
D		A0090	BLOGGS FRED JOHN ALAN
D		A1031	BLAKE JOHN
RAMPスクリーンヘシクナルを送る		A3564	BROWN FREDDY
< >			~
	Web ページから	5のメッセージ	
💰 Employees : 詳細 (A0090-BLOGGS)			
	🔒 👔 my	Function1 executed wi	th parameters 1 2 3
			ОК
	_		

5. 最後に、このコードをRAMPスクリプトに追加します。

var sMessage = SHARED.myFunction2("Hello","World"); alert(sMessage);

以下のような別のメッセージ・ボックスが表示されます。



### 以上で、SHAREDオブジェクトで定義されている共有プロパティおよび 共有ロジックにアクセスする方法を理解しました。

## RAMP-TSAD05 ステップ3. 独自のSHAREDプロパティを作成 する

1. メモ帳を使用して新しいプロパティMessageLineNumberを定義し、 uf\_sy420\_rts.jsファイルを変更します。

messageLineNumber: 22,

#### コードは以下のようになります。



DisplayEmployee画面のスクリプトのvHandle\_ARRIVEファンクションで、前述のステップで追加したコードを削除し、以下のコードの行を追加します。

alert(SHARED.messageLineNumber);

3. DisplayEmployee画面を実行します。以下のようなメッセージ・ボックスが表示されます。

RONICA RED JOHN ALAN IN REDDY
RONICA RED JOHN ALAN IN REDDY
RONICA RED JOHN ALAN IN REDDY
red John Alan In Reddy
IN REDDY
REDDY
1

このようなSHAREDプロパティは、定義の集中管理に役立ち、これらの 変更を容易にします。例えば、RAMPファンクションを使用して画面の 詳細を抽出しているときは、リテラルの22などを使用するよりも、 SHARED.messageLineNumberを使用する方が適切です。

## RAMP-TSAD05 ステップ4. 独自のSHAREDファンクションを 作成する

1. 以下のように、Addというファンクションを追加して、 uf\_sy420\_rts.jsファイルを変更します。

```
/* Add adds 3 numbers together */
Add : function(a,b,c)
{
    return(a + b + c);
}, /* <====== Note the comma ======= */</pre>
```

コードは以下のようになります。



2. RAMP Destination画面スクリプトで、前述のステップのコードを削除し、以下のコードの行を追加します。

var iResult = SHARED.Add(100,200,136); alert("Result = " + iResult);

- 3. フレームワークを終了し、再起動します。
- 4. RAMP 5250 Destination画面を実行します。以下のようなメッセージ・ボックスが表示されます。

	X		
📃 名前 📃 By Date of Birth 📃 By Salar	у	😳 🗆 🔺 🖻 🗎	
▼ リストの消去	検索	Employees	記述
2+ 自加生		A0070	BROWN VERONICA
		A0090	BLOGGS FRED JOHN ALAN
D		A1031	BLAKE JOHN
RAMPスタリーンへシケナルを送る		A3564	BROWN FREDDY
۰ III ۲			
		Web ページからのメッ	t 💌
· · · · · · · · · · · · · · · · · · ·			
🤹 Employees:詳編(A3564-BROWN)	FREDDY)		
🔲 詳細 🔥 イベント 📄 トキュメント		A Decult d	PC
		Kesult = 4.	30
		_	
			ок

以上です。引数を"Add"ファンクションに渡して、その結果を受け取り ました。すべてのRAMPスクリプト間でスクリプト・ロジックを共有す るには、この方法を理解していれば問題ありません。

#### まとめ

#### 重要なポイント

- JavaScriptは、非常に強力なプログラミング言語です。その機能を 理解すればするほど、日常の操作に活用できます。
- また、AJAX WEB 2.0の世界では、JavaScriptの知識がますます不可 欠のITスキルとなっています。
- JavaScriptのトレーニングを提供している多数の無料のオンライン・コースがあります。例えば、http://www.w3schools.com/があります。

### 覚えるべき事柄

• SHAREDプロパティおよびファンクションを使用、作成する方法

#### RAMP-TSAD06: 複数のタブで複数の画面を処理する

RAMP-TS 5250 Destination画面は、フレームワークのコマンド・ハンド ラー・タブに表示されます。通常、1つのコマンド・ハンドラー・タブに 1つの画面が表示されます。ただし、複数のタブに複数の画面を分割する ことができます。このチュートリアルは、この操作に必要な概念とスキ ルを扱っています。

目的

 複数の画面に分散しているDestinationを複数のタブに添付する方法 を学習する

	<	
名前By Date of Birth By Salary	🖸 🕁 🗹 😳	
社員姓 検索	Employees	記述
В	A0070	BROWN VERONICA
	A0090	BLOGGS FRED JOHN ALAN
	A1031	BLAKE JOHN
< >	A3564	BROWN FREDDY
🥻 Employees:住所 (A1031-BLAKE JOHN)		
🔄 詳細 🛯 😕 名前 🔛 住所 🏻 🗐 HR Details 📝		
UFRT S03_D2	このチュートリアルでは	、複数の画面に分散している
	5250社員ファンクションを	を複数のタブに添付する方法に
社員 No A1031	ついて	学習します。
住所1 3 Woodbury Road		
住所2 Winston Hills	エンド・ユーザーは、1回	回のクリックでタブ間を移動し、
住所 3 N SW Australia	フィルターを使用して	て、操作する社員のリストを
	作成	できます。

この目的を達成するには、以下のステップを完了してください。 RAMP-TSAD06 ステップ1. 複数の5250画面のシナリオ RAMP-TSAD06 Step 2. 画面の命名 RAMP-TSAD06 ステップ3. 画面の分類 RAMP-TSAD06 ステップ4. ターゲット・リストの確認と理解 RAMP-TSAD06 ステップ5. 複数のコマンド・ハンドラー・タブの使用 RAMP-TSAD06 ステップ6. ボタンとファンクション・キーの確認と変更 RAMP-TSAD06 ステップ7. 5250アプリケーションに追加した価値の確認 RAMP-TSAD06 付録:ファンクションUFRTS03

### 始める前に

このチュートリアルを開始するには、以下の知識を備えていることを前提としています。必要な操作は次のとおりです。

- RAMPスクリプトの基本構造と仕組を理解している。
- Destination、Junction、またはSpecialとして分類される5250の概念を 理解している。
- フレームワークのコマンド・ハンドラー・タブにDestination画面を スナップする方法を理解している。

これらの前提知識なしにこのチュートリアルを開始した場合、理解できない可能性があります。

#### RAMP-TSAD06 ステップ1. 複数の5250画面のシナリオ

このチュートリアルでは、従来の4画面の5250 キー -> データ照会シーケンスをそのメイン・シナリオとして使用します。

ただし、データ部分は、1つの5250画面に収容するには多すぎるため、以下のように3つの5250画面に分散されています。



これらのサンプル画面には多くの情報が含まれていません。これは、このチュートリアルの目的をぼやけさせないようにするうえで適切です。 5250の各データ画面には多数の情報が組み込まれていると考えてください。通常、3つの5250画面に情報を分割する必要がある理由はこの点にあります。

従来の5250スタイルのナビゲーションを実行するためにファンクショ ン・キーEnterとF12がどのように使用されていたかに注意してくださ い。また、データ画面3で操作している場合、要求画面を経由しないで データ画面2に再び戻ることはできないことにも注意してください。小さ な設計上の不備は、この5250アプリケーションを常時使用しているユー ザーにとって大きないらだちとなる可能性があります。 このチュートリアルをクラスルーム・トレーニングの一部として 学習している場合、表示されたLANSA 5250画面のファンクション は、システムのプロセスUF\_RTS、ファンクションUFRTS03にあ ります。

独習としてこのチュートリアルを学習している場合、ソース・ コードは「RAMP-TSAD06 付録:ファンクションUFRTS03」の UFRTS03ファンクションにあることに注意してください。5250 LANSAプロセス、UF\_RTSを作成し、そのプロセス内にコードを コピーするRDMLファンクション、UFRTS03を作成することをお 勧めします。次に、プロセスとファンクションを5250サーバーに チェックインします。これにより、これらのシナリオを詳細に試 すことができます。

## RAMP-TSAD06 ステップ2. 画面の命名

このステップでは、RAMPツールを使用してプロセス・メニューと4つの 画面に名前を付けます。

この方法が理解できない場合は、このチュートリアルを中止し て、いずれかのコア・チュートリアルを終了してください。

1. System i Main Menuで以下のコマンドを入力します。

lansa run uf\_rts partition(dem)

2. 画面に名前を付けます。

画面	使用する名前
RTSTESTメ ニュー画面	UF_RTS オプションのフィールド[txtOption]にも名前を 付けます。 クラスルームの設定でこのチュートリアルを 学習している場合は、[RTSTEST]メニューのオ プション3を選択して正しい画面が表示される ようにする必要があります。 UF_RTS Ramp Testing Functions 16/10/09 Enter number of function required or place cursor on same line : 3 1. Browse - Select with option number 2. Wultiple Screen Enquiry - 1 3. Wultiple Screen Enquiry - 2 4. Wultiscreen Update 5. Display process or function HELP text 6. Exit from system
キー(社員No.)入 力画面	UFRTS03_R1 また、ナビゲーションで使用される従業員番 号フィールドに <i>txtEmpNoと</i> 名前を付けます。
UFRTS03_D1 - Employee Details	UFRTS03_D1

-1画面	
UFRTS03_D2 - Employee Details - 2 画面	UFRTS03_D2
UFRTS03_D3 Employee Details - 3 画面	UFRTS03_D3

# RAMP-TSAD06 ステップ3. 画面の分類

•

1. 必要なすべての画面を特定し、必要なフィールドに名前を付けた ら、画面を分類する必要があります。

使用した名前	この画面の分類
UF_RTS	Junction
UFRTS03_R1	Junction
UFRTS03_D1	Destination
UFRTS03_D2	Destination
UFRTS03_D3	Destination

#### RAMP-TSAD06 ステップ4. ターゲット・リストの確認と理解

今までのステップで、1つのJunction画面(UFRTS03\_R1)と3つのDestination 画面(UFRTS03\_D1、UFRTS03\_D2、UFRTS03\_D3)を定義してスクリプト を作成しました。

RAMPツールを開き、各画面定義をクリックしてターゲット・リストを 確認します。

各么	マーケ	ッ	ト・	リス	トは以	、下のよう	に表示され	れます。
	-	-	-					

画面名	画面のタイプ	ターゲット
UFRTS03_R1	Junction	UFRTS03_D1
UFRTS03_D1	Destination	UFRTS03_R1 UFRTS03_D2
UFRTS03_D2	Destination	UFRTS03_R1 UFRTS03_D3
UFRTS03_D3	Destination	UFRTS03_R1

ターゲット・リストがこのように表示されない場合は、表示されるまで RAMPツールを使用して画面ナビゲーションを手動で示す必要がありま す。

画面定義に関連付けられているターゲットは非常に重要です。これら は、特定の画面でどの画面(ターゲット)に移動できるかをRAMPナビゲー ション・プランナーに指示します。通常、ターゲットはナビゲーション を手動でRAMPツール・エディターに示すときに自動的に更新されま す。

RAMPでより多くの経験を重ねるにつれて、ターゲット・リスト を手動で更新し、適切なコードを画面のナビゲーション・ハンド ラー・ファンクションに追加することができます。実際には、こ れによりRAMPツールを使用してナビゲーションを正確に示して います。

ここで使用しているターゲット・リストは、以下のように容易に理解で きます。

- Junction UFRTS03\_R1のターゲット・リスト は、"vHandle\_NAVIGATETOファンクションはDestination UFRTS03\_D1に移動できるスクリプトを含む"と明確に示していま す。
- Destination UFRTS03\_D1のターゲット・リスト は、"vHandle\_NAVIGATETOファンクションはJunction UFRTS03\_R1ま たはDestination UFRTS03\_D2に移動できるスクリプトを含む"と明確に 示しています。
- 3つのすべてのDestinationのターゲット・リストにJunction UFRTS03\_R1が存在することには、重要な意味があります。 Destinationのターゲット・リストにある最初のJunctionは、Exit Junctionと呼ばれています。Exit Junctionは、Destinationを出て、「高 速道路や自動車道路」(つまり、Destination間をすばやく移動するため に使用される個別のJunctionのネットワーク)のJunctionに戻るために使 用されます。Destination画面が表示されている場合、Destination画面 は、それ自体を含め、別のDestinationに移動する前に、指定され た"Exit Junction"に移動するように指定されます。

RAMP-TSAD06 ステップ5. 複数のコマンド・ハンドラー・タブの使用

この種のシナリオの処理で利用可能な最初のオプションは、各 Destination画面をそれ独自のコマンド・ハンドラー・タブに配置するこ とです。

これは、ユーザーがデータ画面1と2に移動しないとデータ画面3などに即 座に移動できない5250アプリケーションに価値を付加します。これによ り、ユーザーはキャンセルして主要画面に戻ることなく、3つのデータ画 面すべての間を自由に移動できます。

- フレームワークで、Employeeビジネス・オブジェクトに3つの新しい インスタンス・レベルのコマンド、[名前]、[住所]、[HR詳細]を追加し ます。
- Nameをデフォルトのコマンドにし、コマンドのシーケンスを1、
   3に設定します。従業員に関連付けられているその他のコマンドがこれらのコマンドの後にくるようにシーケンスを再設定します。
- 3. RAMPツールで、Destination画面をこれらのコマンドにリンクしま す。

UFRTS03_D1	[名前]
UFRTS03_D2	[住所]
UFRTS03_D3	[HR詳細]

 UFRTS03\_R1のスクリプトを変更します。インスタンス・リストで 選択した従業員についてコマンドが表示されるように、SETVALUEを 変更する必要があります。

case "UFRTS03\_D1":
{
 /\* Set up data fields on form UFRTS03\_R1 \*/
 SETVALUE("EmpNo",objListManager.AKey1[0]);
 SENDKEY(KeyEnter);
 /\* Check for arrival at UFRTS03\_D1 \*/
 Q\_CHECK\_CURRENT\_FORM("UFRTS03\_D1","Unable to navigate to form
 UFRTS03\_D1");

} break;

「RAMP-TS006 ステップ2. 現在のインスタンス・リスト・エントリーを 使用するようにスクリプトを変更する」を確認してください。

5. フレームワークを保存して再起動します。

従業員を表示すると、コマンド・タブは次のようになります。

	×	·			X
□ 名前 □ By Date of Birth □	By Salary	🗄 🔺 🗎 🖂 🗖	🗿 🤠		
社員姓	検索	Employees	記述		
В		A0070	BROWN VER	ONICA	
		A0090	BLOGGS FRE	D JOHN ALAN	
V YXNU)自去		A1031	BLAKE JOHN		
3		A3564	BROWN FRE	DDY	
					X
🤞 Employees:名前(A1031-I	BLAKE JOHN)				
🔲 詳細 🛛 🎐 名前 🛛 区 住所 📄	🖐 HR Details	📄 ዞ፝፝キュメント   🛕 イベン	·ト		
UFR	TS03_D1 - Emp	oloyee Details - 1		Enter	スクリ・
社員 No A1031				PROMPT	
社員姓 BLAKE				<b>En</b> 2941	
社員名 JOHN				邦公育し	
				MSG	
				火ューバーを表示	
				トドキュメンテーション	
				トレースオン	

3つの5250画面は任意の組み合わせで表示できることに注意してください。

スクリプトをトレースして変更したり、新しいナビゲーションを示して このサンプルが正しく機能するようにしなければならないことがありま す。これは、5250画面の対話処理のスクリプトの通常の部分です。3つの 画面すべてが適切に機能するまで、この操作を続ける必要があります。 今まで終了したチュートリアルでは、スクリプトが適切に機能するまで デバッグを行うために必要なスキルが提供されています。
RAMP-TSAD06 ステップ6. ボタンとファンクション・キーの 確認と変更

すべての5250ファンクション・キーは、ナビゲーション操作に関連して います。また、すべてのナビゲーションをより適切なものに置き換えま した。5250のEnterを使用した先へ進む操作、F12を使用したキャンセル 操作は、Windowsアプリケーションでは多くの場合意味がありません。 例:従業員A0090の住所に関する照会をキャンセルしないで、次に実行 する内容(従業員A0070の名前の詳細を表示するなど)に移動します。刷新 プロジェクトでは、この処理に努める必要があります。こうすること で、アプリケーションはWindowsインターフェースを使用していた人に とってより親しみやすく、快適になります。

- 1. RAMPツールで、<u>すべての</u>ファンクション・キーまたはボタンが無 効になるように、Destination画面UFRTS03\_D1、UFRTS03\_D2、 UFRTS03\_D3を変更します。
- 2. [保存]を実行します。
- 3. フレームワークで、別の従業員を表示します。設計時のボタンのみ が表示されます。

		X
🤞 Employees:住	所(A0070-BROWN VERONICA)	
🔲 詳細   🤔 名前	- 🖂 住所 🛛  HR Details   📄 ドキュメント   🛕 イベント	
	UFRTSO3_D2 - Employee Details - 2	バーを表示
社員No	A0070	トドキュメンテーション
住所 1	12 Railway Street	
住所 2	Baulkham Hills	トレース オン
住所3	NSW Australia	スケリーンの調査

# **RAMP-TSAD06** ステップ7.5250アプリケーションに追加した 価値の確認

以下の処理のみを実行できる非常に簡素な5250の4つの画面照会を開始し ました。



これは、次のように表示されます。

X 国家 Date of Birth 国 By Salary	🗿 🕁 🖂 🕁 🗆	
社員姓 検索	Employees	記述
В	A0070	BROWN VERONICA
同時の海井	A0090	BLOGGS FRED JOHN ALAN
	A1031	BLAKE JOHN
< >	A3564	BROWN FREDDY
✓ Employees:住所(A1031-BLAKE JOHN) □詳細 ● 名前 ☑ 住所 ● HR Details ● UFRTS03_D2 - Employ	ドキュメント 🛕 イベント yee Details - 2	
社員 Na A1031		
住所1 3 Woodbury Road		
住所2 Winston Hills		
住所3 NSW Australia		

- 以上で、ユーザーは操作したい従業員のリストを作成できます。
- また、1回のクリックで従業員A0070の名前の詳細表示から[HR詳 細]に移動し、1回のクリックで戻ることができます。
- さらに、2回のクリックで従業員A0070の名前の詳細表示から従業員A1031の[HR詳細]に移動できます。

# RAMP-TSAD06 付録:ファンクションUFRTS03

FUNCTION OPTIONS(\*DIRECT) BEGIN\_LOOP

- \* Get the key details
- \* A classic junction
- \* Cancel key goes back to process menu or caller program

RQ1: DOUNTIL COND('#IO\$STS = OK') REQUEST FIELDS(#EMPNO) EXIT\_KEY(\*NO) PANEL\_ID(UFRTS03\_R1) I FETCH FIELDS(\*ALL) FROM\_FILE(pslmst) WITH\_KEY(#EMPNO) ISSUE\_ ENDUNTIL

- \* Display details screen 1
- \* Cancel key goes back to request next employee number
- \* Enter goes forward to Details screen 2

DISPLAY FIELDS(#EMPNO #SURNAME #GIVENAME) EXIT\_KEY(\*NO) ]

- \* Display details screen 2
- \* Cancel key goes back to request next employee number
- \* Enter goes forward to Details screen 3

DISPLAY FIELDS(#EMPNO #ADDRESS1 #ADDRESS2 #ADDRESS3) EXIT

- \* Display details screen 3
- \* Cancel key goes back to request next employee number
- \* Enter goes forward to request next employee number

DISPLAY FIELDS(#EMPNO #SALARY #DEPTMENT #SECTION) EXIT\_KE

\* Loop around and ask for the next employee number

END\_LOOP

## RAMP-TSAD07:1つのタブで複数の画面を扱う

RAMP-TS 5250 Destination画面は、フレームワークのコマンド・ハンド ラー・タブに表示されます。通常、1つのコマンド・ハンドラー・タブに 1つの画面が表示されます。ただし、1つのタブに複数の5250画面を表示 することができます。このチュートリアルは、この操作に必要な概念と スキルを扱っています。

目的

 複数の画面に分散しているDestinationを1つのタブに添付する方法、 および画面間を移動するボタンを有効にする方法を学習する



この目的を達成するには、以下のステップを完了してください。 RAMP-TSAD07 ステップ1. 複数の5250画面のシナリオ RAMP-TSAD07 ステップ2. 計画を立てる RAMP-TSAD07 ステップ3. 1つのタブに複数の画面を配置する RAMP-TSAD07 ステップ4. ファンクション・キー/ボタンの有効化と必要 なスクリプトの追加 RAMP-TSAD07 ステップ5. Exit JunctionとvHandle\_NAVIGATETOスクリ プトの定義 RAMP-TSAD07 ステップ6. テストとデバッグの実行 RAMP-TSAD07 ステップ7. 微調整

## 始める前に

このチュートリアルを開始するには、以下の知識を備えていることを前提としています。必要な操作は次のとおりです。

 前のミニチュートリアル「RAMP-TSAD06:複数のタブで複数の画 面を処理する」が完了している。これらの前提知識なしにこのチュー トリアルを開始した場合、理解できない可能性があります。

# RAMP-TSAD07 ステップ1. 複数の5250画面のシナリオ

このチュートリアルでは、前のチュートリアルと同様に、従来形式の同 じ4つの画面の5250キー -> データ照会ファンクションUFRTS03をシナリ オに使用します。「RAMP-TSAD06 ステップ1. 複数の5250画面のシナリ オ」を参照してください。

## RAMP-TSAD07 ステップ2. 計画を立てる

「RAMP-TSAD06:複数のタブで複数の画面を処理する」チュートリア ルでは、これらの画面を識別し、3つのコマンド・ハンドラー・タブで機 能するようにしました。

ここでは、3つのDestination画面を[全ての明細]という1つのコマンド・タ ブに表示し、ユーザーがこれらの画面間を[前へ]ボタンと[次へ]ボタンを 使用して移動できるようにします。このためには、以下のように計画を 立てる必要があります。

画面	関連付け られるコ マンド・ タブ	有効なボタン/ファンク ション・キー	ターゲット
UFRTS03_D1	[全ての詳 細]	次へ/Enter -> UFRTS03_D2	UFRTS03_R1
UFRTS03_D2	-	前へ/F12 - > UFRTS03_D1 次へ/Enter -> UFRTS03_D3	UFRTS03_R1
UFRTS03_D3	-	前へ/F12 -> UFRTS03_D2	UFRTS03_R1

#### この計画を詳細に確認して、以下の質問に答えられる必要があります。

質問	注記
UFRTS03_D1のみがコマン ド・ハンドラー・タブにリン クされているのはなぜですか?	1つのコマンド、[全ての明細]の みを使用する予定であり、ユー ザーがこのコマンドを実行する と、コマンド・タブに画面 UFRTS03_D1が表示されます。
UFRTS03_D2とUFRTS03_D2が コマンド・ハンドラー・タブ にリンクされていないのはな ぜですか?	これらは、コマンド自体にはリ ンクされていません。画面 UFRTS03_D2またはUFRTS03_D3 には、UFRTS03_D1を経由し、

	[次へ]ボタンを使用して進みま す。これらの画面に個別に直接 アクセスできるようにするに は、画面を独自のコマンド・タ ブに配置します。
画面UFRTS03_D1が表示され ると、どのファンクション・ キー/ボタンが有効になり、ど のように機能しますか?	[次へ]ボタンとEnterキーが有効 になります。 これらを使用すると、画面 UFRTS03_D1が表示されます。
画面UFRTS03_D2が表示され ると、どのファンクション・ キー/ボタンが有効になり、ど のように機能しますか?	<ul> <li>[前へ]ボタンとF12キーが有効になります。</li> <li>これらを使用すると、画面UFRTS03_D1が(再)表示されます。</li> <li>[次へ]ボタンとEnterキーが有効になります。</li> <li>これらを使用すると、画面UFRTS03_D3が表示されます。</li> </ul>
画面UFRTS03_D3が表示され ると、どのファンクション・ キー/ボタンが有効になり、ど のように機能しますか?	[前へ]ボタンとF12キーが有効に なります。 これらを使用すると、画面 UFRTS03_D2が(再)表示されま す。
3つのDestination画面すべてに Exit Junctionがあるのはなぜで すか?	表示されたDestinationは、指定 されたExit Junctionと、その Junctionに移動するための適切な コードをvHandle_NAVIGATETO スクリプトに含んでいる必要が あります。
Exit Junctionはどのように使用 されますか?	Destination画面が表示される と、別のビジネス・オブジェク トやビジネス・オブジェクト・ インスタンスなど(別の発注、製 品、または顧客など)の別の

Destination画面にいつでも置き	Destination画面にいつでも置き
換えることができます。これを	換えることができます。これを
可能にするために、Destination	可能にするために、Destination
画面は、RAMPナビゲータが	画面は、RAMPナビゲータが
Destination画面を出て、Junction	Destination画面を出て、Junction
(高速道路/自動車道路のジャン	(高速道路/自動車道路のジャン
クション)に戻り、次の	クション)に戻り、次の
Destination (目的地)への最速の経	Destination (目的地)への最速の経
路を計画できるようにするExit	路を計画できるようにするExit
Junctionを持っている必要があり	Junctionを持っている必要があり
ます。	ます。

このチュートリアルの次のいくつかのステップでは、この計画の実装およびテスト/デバッグを扱います。

RAMP-TSAD07 ステップ3.1つのタブに複数の画面を配置する このステップでは、複数の画面を1つのコマンド・ハンドラー・タブに 添付します。

- 1. フレームワークで、[全ての明細]コマンドをEmployeeビジネス・オ ブジェクトに関連付けます。
- 2. このコマンドをデフォルトにします。

💑 ቲጭቶአ አንጭታትወንግለምንና - Employees					
識別子   アイコン   ビジュアル スタイル   フィルター   使用可/不可の設定を切り替えるにコよリストかい トロッフ .キオ	フィルター設定 使用可能コマント コマントの表示 コーサ らト・ラッケ・アント・	ザ━ プロ パティ  サブタイプ  インスタンス リストの設定			
<pre> PHY/ULL 9      選択可能コマント*     AJAXEX1     AJAXEX2     AJAXEX3     Assistant Exa     Assistant Exa     Command Pane     CRUD     Details RAMP-TS     e-mail     Example 1     Example 2     Example 2     Example 3     Example 4     Example 5     Example 6     Example 8     Example 8     Example 8     Example 8     Example 8 </pre>		<ul> <li> インスタンス コマント* </li> <li> ウィント*ウのサイズ* 「幅 高さ ウィント*ウ Web フ*ラウザ* </li> <li> <b>任意引数</b> 文字引数 1: 文字引数 2: 数字引数 1: 数字引数 2: </li> </ul>			

- 3. RAMPツールで、UFRTS03\_D1画面の詳細を見つけて、[名前]コマンドとの関連付けを削除し、画面を[全ての明細]コマンドに関連付けます。
- 4. UFRTS03\_R2とUFRTS03\_R3からコマンド・ハンドラーの関連付けを 削除します。
- 5. RAMP定義の部分的な保存を実行します。
- 6. フレームワークを保存して再起動します。

以上で、[全ての明細]というキャプションのコマンド・ハンドラー・タ ブが作成されました。このタブを使用して、選択した従業員の基本的な 詳細(画面UFRTS03\_D1)を表示できます。

▲ 白 By Date of Birth By Salary 社員姓 B ッリストの消去 ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・					
社員姓 B Employees 記述 A0070 BROWN VERONICA A0090 BLOGGS FRED JOHN ALAN A1031 BLAKE JOHN A3564 BROWN FREDDY					
B ダリストの消去 A0070 BROWN VERONICA A0090 BLOGGS FRED JOHN ALAN A1031 BLAKE JOHN A3564 BROWN FREDDY					
<ul> <li>リストの消去</li> <li>A0090</li> <li>BLOGGS FRED JOHN ALAN</li> <li>A1031</li> <li>BLAKE JOHN</li> <li>A3564</li> <li>BROWN FREDDY</li> </ul>					
A1031 BLAKE JOHN A3564 BROWN FREDDY					
A3564 BROWN FREDDY					
🤞 Employees:全ての明細(A1031-BLAKE JOHN)					
🔃 全ての明細 📄 詳細   🧐 名前   🖂 住所   🗐 HR Details   📄 ドキュメント   🔔 イベント					
UFRTSO3_D1 - Employee Detaris - 1					
社員 No A1031					
社員姓 BLAKE					
社員名 JOHN					

RAMP-TSAD07 ステップ4.ファンクション・キー/ボタンの有 効化と必要なスクリプトの追加

このステップでは、画面の[次へ]および[前へ]ボタンを有効にして、エンド・ユーザーがファンクション内で画面間を移動できるようにします。 また、スクリプトにいくつかの変更も加えます。

 
 画面UFRTS03\_D1、UFRTS03\_D2、UFRTS03\_D3のプロパティを確認 し、以前計画したようにボタン/ファンクション・キーを有効にしま す。

UFRTS03\_D1は以下のようになります。

1	ーファンクション キ	-使用可能			
	<b>+</b> -	Caption	利用可能な5250	利用可能なVLF	p FKEY se 🔺 🔼
	Enter	Next		<ul> <li>Image: A set of the set of the</li></ul>	1
	F1	F1			2
	F?	F?			3

#### UFRTS03\_D2は以下のようになります。

ſ	-ファンクション キ	-使用可能			
	<b>+</b> -	Caption	利用可能な5250	利用可能な¥	FKEY s 🔺 🔥
	Enter	Next	✓	<b>~</b>	1
	F12	Previous			2

## UFRTS03\_D3は以下のようになります。

ーファンクション キ	一使用可能————				
<b>+</b> -	Caption	利用可能な5250	利用可能なVL	FKEY : 🔺	^
F12	Previous	Image: A start of the start	Image: A start of the start	1	
Enter	Enter			2	
F1	F1			3	
F3	F3			А	

2. 変更を保存し、フレームワークを再起動します。

次に、これらの各画面のvHandle\_BUTTONCLICKスクリプトを確認し、 ボタン・ナビゲーションが正しく処理されるように変更します。

3. まず、「RAMP-TSAD06 ステップ1. 複数の5250画面のシナリオ」の 5250ナビゲーションの図を参照してください。 4. このナビゲーションの図から、何を実行する必要があるか非常に簡 単にわかります。

画面	ボタン/ ファン クショ ン・ キー	結果として 表示される 画面	<b>vHandle_BUTTONCLICK</b> が実行する必要がある内 容
UFRTS03_D1	次 ヘ/Enter	UFRTS03_D2	Enterキーの送信。 デフォルトのボタン・ス クリプトがこれを処理し ます。
UFRTS03_D2	次 ヘ/Enter	UFRTS03_D3	Enterキーの送信。 デフォルトのボタン・ス クリプトがこれを処理し ます。
UFRTS03_D2 (注を参照)	前へ/F12	UFRTS03_D1	F12を送信して UFRTS03_R1 (Junction)を 取得します。 Enterを送信して UFRTS03_D1に進みま す。 デフォルトのボタン・ス クリプトはこれを処理し ません。
UFRTS03_D3 (注を参照)	前へ/F12	UFRTS03_D2	F12を送信して UFRTS03_R1 (Junction)を 取得します。 Enterを送信して UFRTS03_D1に進みま す。 Enterを送信して UFRTS03_D2に進みま す。

	デフォルトのボタン・ス クリプトはこれを処理し ません。	スし
--	------------------------------------	----

UFRTS03\_D2とUFRTS03\_D3への前へ/F12操作の追加に注意してくださ い。基盤の5250アプリケーションでは、このような直接の移動は存在し ません(つまり、1つの操作でUFRTS03\_D2からUFRTS03\_D1に実際に移動 することはできません)。

ただし、いくつかのシンプルなスクリプトを使用して、この機能が実際 に存在する場合と同様に画面をユーザーに表示することができます。こ れは、既存の5250アプリケーションに価値を付加するもう1つのシンプル な例です。

5. vHandle\_BUTTONCLICKファンクションを変更します。 UFRTS03\_D1スクリプトを変更する必要はありません。

UFRTS03\_D2で、以下のようなボタン・クリック・スイッチ文を使用します。

```
switch (sButton)
```

```
{
```

case KeyEnter: /\* Enter-Next means move forward to UFRTSD03\_03 \*/ SENDKEY(KeyEnter);

break;

case KeyF12: /\* F12-Previous means go back to UFRTSD03\_01 \*/ Q\_SENDKEY("",KeyEnter);

SENDKEY(KeyF12);

break;

default:

```
ALERT_MESSAGE("Invalid function key used");
break;
```

}

UFRTS03\_D3で、以下のようなボタン・クリック・スイッチ文を使用します。

switch (sButton)

{

case KeyF12: /\* F12-Previous means go back to UFRTSD03\_02 \*/
 Q\_SENDKEY("",KeyEnter);

```
Q_SENDKEY("",KeyEnter);
SENDKEY(KeyF12);
break;
default:
ALERT_MESSAGE("Invalid function key used");
break;
}
```

注:

- これらのRAMPスクリプトは、F12の使用の実際の意味を再解釈するために変更されました。5250画面UFRTS03\_D3が表示されているRAMPタブでは、以上の変更により、F12ファンクション・キーは、F12、Enter、5250サーバーへのEnterの送信を意味します。ユーザーがこの画面から前の画面に直接移動できるようになったため、この再解釈は基盤の5250アプリケーションに価値を付加します。多数の価値あるビジネス・プロセスの機能強化は、このように非常にシンプルな戦略によって支えられています。
- すべての無効なファンクション・キーによってアラート・メッセージが表示されるようになりました。これは、フェイル・セーフのみを目的としています。RAMPでは、何らかの方法でキーの使用を禁止する必要があります。
- Q\_SENDKEY()ファンクションの使用。RAMP-TS SENDKEY()の操作は非同期です。このため、即座に送信できるのは最初の SENDKEY()要求だけです。以降の要求を待ち行列に入れて、結果の 画面が非同期で到着したときに処理する必要があります。
- 7. スクリプトの変更を保存し、RAMP定義の部分的な保存を行いま す。

次のステップを完了するまで、変更をテストしないでください。そうで ない場合、アプリケーションのナビゲーションがスタックすることがあ り、アプリケーションのキャンセルが必要になります。

# RAMP-TSAD07 ステップ5. Exit Junctionと vHandle\_NAVIGATETOスクリプトの定義

実行する必要がある最後の内容は、3つのすべてのDestinationが指定され たExit Junctionを持っていること、またこれらのDestinationの vHandle\_NAVIGATETOファンクションが要求されたときにExit Junctionに 移動できることを確認することです。

- 1. RAMPツールを開き、3つのDestination画面、UFRTS03\_D1、 UFRTS03\_D2、UFRTS03\_D3のプロパティを確認します。
- 各Destination画面で、Junction UFRTS03\_R1の名前のみを含むように 表示されるターゲット・リストを編集します。各ターゲット・リスト に変更を加えた後、[保存]ボタンをクリックしてください。



Exit Junctionに関する技術ノート:Destination画面が表示され、別の画面 へのナビゲーションを実行する必要がある場合は、ターゲット画面への ナビゲーション・ルートを計算する前にExit Junctionに移動するように Destination画面に指示します。デフォルトでは、Destination画面のExit Junctionはターゲット・リストで定義されている最初のJunctionです。

- 3. ここで、各Destination画面に関連付けられているスクリプトを確認 します。
- 画面のvHandle\_NAVIGATETOファンクションを見つけ
   て、*UFRTS03\_R1というJunction*に移動する要求を処理できるスクリプトが含まれていることを確認します。

スクリプトには、以前のチュートリアルからこのコードがすでに含まれ ている可能性があります。ただし、ターゲット・リストを手動で編集 し、関連する実装ロジックをvHandle\_NAVIGATETOファンクションに追 加できる(実行する場合もある)ことを理解することが重要です。 最も簡潔なフォームでは、すべてのDestination画面の vHandle NAVIGATETOファンクションは次のようになります。

```
switch (sToForm)
```

{

```
/* <NAVIGATESWITCH> - Do not remove or alter this line */
case "UFRTS03_R1": /* Navigate back to the exit junction */
{
    SENDKEY(KeyF12);
    Q_CHECK_CURRENT_FORM("UFRTS03_R1","Unable to navigate to
    }
break;
default: /* Handle an invalid request */
    alert("Form " + this.vName + " cannot navigate to form " + sToForm);
    bReturn = false;
    break;
}
```

#### RAMP-TSAD07 ステップ6. テストとデバッグの実行

この最終ステップでは、コードで定義したナビゲーションの幅広いテス トを実行します。

[全ての明細]コマンド・ハンドラー・タブが以下の状況を処理できることを確認します。

- 3つの異なるDestination画面間で、種々のバリエーションと組み合わせにより、[次へ]->[次へ]->[前へ]サイクルを繰り返します。
- UFRTS03\_D1を表示しているときにインスタンス・リストで別の従 業員を選択します。
- 最初にUFRTS03\_D2を表示しているときにインスタンス・リストで 別の従業員を選択します。
- 最初にUFRTS03\_D3を表示しているときにインスタンス・リストで 別の従業員を選択します。
- UFRTS03\_D1を表示しているときに、他のビジネス・オブジェクト に移動して戻ります。
- UFRTS03\_D2を表示しているときに、他のビジネス・オブジェクト に移動して戻ります(UFRTS03\_D1に戻る必要があります)。
- UFRTS03\_D3を表示しているときに、他のビジネス・オブジェクト に移動して戻ります(UFRTS03\_D1に戻る必要があります)。

問題がある場合は、アプリケーション・レベルのトレースを使用して、 スクリプトの実行時に何が発生しているかを確認する必要があります。

## RAMP-TSAD07 ステップ7. 微調整

[次へ] -> [次へ] -> [前へ]ボタン処理のテスト中、RAMP-TSが中間画面を 通じて移動する際にフリッカーに気付くことがあります。

これは、画面UFRTS03\_D3で[前へ]ボタンを使用しているときに特に目立 つことがあります。その理由は、基盤の5250アプリケーションの設計で この動作が意図されているためです。

```
switch (sButton)
{
    case KeyF12: /* F12-Previous means go back to UFRTSD03_02 */
    Q_SENDKEY("",KeyEnter);
    Q_SENDKEY("",KeyEnter);
    SENDKEY(KeyF12);
    break;
    default:
        ALERT_MESSAGE("Invalid function key used");
        break;
```

このフリッカーは、現在の5250画面を表示してこのボタン処理シーケン スを開始したことが原因です。

スクリプトはまだ完全ではなく、微調整が必要です。これを次のように 変更します。

```
switch (sButton)
{
    case KeyF12: /* F12-Previous means go back to UFRTSD03_02 */
    HIDE_CURRENT_FORM();
    Q_SENDKEY("",KeyEnter);
    Q_SENDKEY("",KeyEnter);
    SENDKEY(KeyF12);
    break;
    default:
        ALERT_MESSAGE("Invalid function key used");
        break;
    }
UDE_CURDENT_EODMODE=>
```

HIDE\_CURRENT\_FORM()ファンクション呼び出しにより、現在の RAMP-TS画面は非表示になり、中間画面ナビゲーション(フラッシュ)が 表示されなくなります。 最終的なDestinationのvHandle\_ARRIVEスクリプトのSHOW\_FORMファン クションが実行されると、RAMP\_TS画面が再び表示されます。

## RAMP-TSAD08:画面ラッパーの基本

このチュートリアルでは、画面ラッパーを作成する基本的な手順を示します。

画面ラッパーは、Visual LANSAの再利用可能なパーツで、1つ以上の 5250画面に移動して、画面の値を取得および設定できます。ラッパー は、既存の5250アプリケーションを書き直すことなく拡張できる方法を 提供します。

画面ラッパーはユーザー・インターフェースになりますが、根底では既 存の5250アプリケーションが引き続き使用されます。

これは簡素な導入演習です。実際には、この5250画面で画面ラッパーを 作成する理由はありません。

目的

• 画面ラッパーの作成の基本を理解する



この目的を達成するには、以下のステップを完了してください。 RAMP-TSAD08 ステップ1. ラッパーで使用されるフィールドに名前を付ける RAMP-TSAD08 ステップ2. 画面ラッパーの作成とスナップ RAMP-TSAD08 ステップ3. 画面ラッパー・コードについて RAMP-TSAD08 ステップ4. ラッパーからの画面の更新のテスト まとめ 始める前に

このチュートリアルを完了するには、コア・チュートリアルRAMP-TS001〜RAMP-TS015を終えている必要があります。 RAMP-TSAD08 ステップ1. ラッパーで使用されるフィールド に名前を付ける

このステップでは、画面ラッパーで使用できるように、DisplayEmployee 画面上のすべてのフィールドに名前を付けます。

- 1. RAMPツールを起動します。
- 2. RAMP-TS 5250エミュレータを起動します。
- 3. 資格情報の検索 / 保守画面に移動します。

LANSA RUN PSLSYS PARTITION(DEM)

- 4. 社員管理メイン・メニューでオプション3を選択します。
- 5. 社員No.を指定します。例: A1234
- 6. F21キーを押して、画面を入力モードにします。
- 7. 画面上のフィールドに名前を付けます(前のチュートリアルで一部の フィールドには名前が付けられています)。フィールドには、以下のように名前を付けます。

社員No.	txtEmpno
社員姓	txtSurname
社員名	txtGivename
住所 1	txtAddress1
住所 2	txtAddress2
住所 3	txtAddress3
郵便番号	txtPostcode
自宅電話番号	txtPhone
部門コード	utxtDepartment
課コード	SECTION

入社日付 (DDMMYY)	Date_Start_DDMMYY			
退職日付 (DDMMYY)	Date_Termn_DDMMYY			

#### スキル・テーブルのサブファイル列にも名前を付ける必要があります。

8. 列の先頭ではなく、取得日(DDMMYY)見出しを選択し、*subDateacq* という名前を付けます。

74- <b>1</b> .	**:			Lock Scre	een	自宅電話番号		(02) 9871	7773	2125	
	名前	fī	列	Sfl		部門コード		SD	+		
		16	68		^	그 개미 선 생		<u>ES</u>	+		
-	資格	-				调整日付 (1		0/00/00			
	subDateacq	17	6	γ				<u> </u>			
	subSkillcode	17	15	Y		取得日 (DOMMYX)	資格フェド	次故記述	TVNA	次故	
Г	subSkilldesc	17	26	Y		98/03/2	ADMINI	_ Administratn Part 1	Met requirement	D	
	Administratn Part 1					98/05/03	ADMIN2	Administratn Part 2		<u>D</u>	
П	subComment	17	47	Y		98/05/09	ECD ENG	Economics Degree	With distinction	_ <u>D</u>	
	subGrade	17	70	Y		96/09/1	INDUCT	Company Induction		P	
		21	79	٧		F1= ヘルブ	F3= 終了	F4=PROMPT F12= 取消	L F14=MSG		
_	+	22	2								
		25	2								

9. 以下のように残りの列に名前を付けます。

資格コード	subSkillcode
資格記述	subSkilldesc
コメント	subComment
資格等級	subGrade
+(プラス記号)	moreindicator

チュートリアルの最初の部分で名前を付けたすべてのフィールドを使用 するわけではないことに注意してください。

10. 画面定義を保存します。

以上で、画面上のすべてのフィールドに名前を付けました。これで、画 面ラッパー内に配置する準備が整いました。 RAMP-TSAD08 ステップ2. 画面ラッパーの作成とスナップ このステップでは、最初の画面ラッパーのコードをコピーし、画面ラッ パーをフレームワークにスナップします。

- 1. Visual LANSAエディターで、iiiRMP03という名前の、新しい再使用 可能なパーツを作成します。DisplayEmployee Wrapperと記述します。
- 2. 再使用可能なパーツのコードを以下のコードに置き換えます。

Function Options(\*DIRECT) BEGIN\_COM ROLE(\*EXTENDS #VF\_AC010) HEIGHT(227) LAYOUTMANAGER(#MAIN\_LAYOUT) WIDTH(497) \*

\* Simple Field and Group Definitions

\*

Group By Name(#XG HEAD) Fields(#EMPNO #SURNAME #GIVENAME #ADDRESS1 #ADDRESS2 #ADDRESS3 #POSTCODE #PHONEHME **#DEPTMENT #SECTION**) \* Body and Button arrangement panels DEFINE COM CLASS(#PRIM PANL) NAME(#BUTTON PANEL) DISPLAYPOSITION(2) HEIGHT(227) HINT(\*MTXTDF\_DET1) LAYOUTMANAGER(#BUTTON FLOW) LEFT(409) PARENT(#COM OWNER) TABPOSITION(3) TABSTOP(False) TOP(0) WIDTH(88) DEFINE COM CLASS(#PRIM PANL) NAME(#BODY HEAD) DISPLAYPOSITION(1) HEIGHT(227) HINT(\*MTXTDF\_DET1) LAYOUTMANAGER(#BODY HEAD FLOW) LEFT(0) PARENT(#COM OWNER) TABPOSITION(2) TABSTOP(False) TOP(0) VERTICALSCROLL(True) WIDTH(409) \* Attachment and flow layout managers DEFINE COM CLASS(#PRIM ATLM) NAME(#MAIN LAYOUT) DEFINE COM CLASS(#PRIM FWLM) NAME(#BUTTON FLOW) DIRECTION(TopToBottom) FLOWOPERATION(Center) MARGINBOTTOM(4) MARGINLEFT(4) MARGINRIGHT(4) MARGINTOP(4) SPACING(4) SPACINGITEMS(4) DEFINE\_COM CLASS(#PRIM\_FWLM) NAME(#BODY\_HEAD\_FLOW) DIRECTION(TopToBottom) MARGINBOTTOM(4) MARGINLEFT(4) MARGINRIGHT(4) MARGINTOP(4) SPACING(4) SPACINGITEMS(4)

DEFINE COM CLASS(#PRIM FWLI) NAME(#FWLI EMPNO) MANAGE(#EMPNO) PARENT(#BODY HEAD FLOW) DEFINE\_COM CLASS(#PRIM\_FWLI) NAME(#FWLI\_SURNAME) MANAGE(#SURNAME) PARENT(#BODY HEAD FLOW) DEFINE COM CLASS(#PRIM FWLI) NAME(#FWLI GIVENAME) MANAGE(#GIVENAME) PARENT(#BODY HEAD FLOW) DEFINE\_COM CLASS(#PRIM\_FWLI) NAME(#FWLI\_ADDRESS1) MANAGE(#ADDRESS1) PARENT(#BODY\_HEAD\_FLOW) DEFINE COM CLASS(#PRIM FWLI) NAME(#FWLI ADDRESS2) MANAGE(#ADDRESS2) PARENT(#BODY\_HEAD\_FLOW) DEFINE\_COM CLASS(#PRIM\_FWLI) NAME(#FWLI\_ADDRESS3) MANAGE(#ADDRESS3) PARENT(#BODY\_HEAD\_FLOW) DEFINE\_COM CLASS(#PRIM\_FWLI) NAME(#FWLI\_POSTCODE) MANAGE(#POSTCODE) PARENT(#BODY HEAD FLOW) DEFINE COM CLASS(#PRIM FWLI) NAME(#FWLI PHONEHME) MANAGE(#PHONEHME) PARENT(#BODY HEAD FLOW) DEFINE\_COM CLASS(#PRIM\_FWLI) NAME(#FWLI\_SAVE\_BUTTON) MANAGE(#SAVE\_BUTTON) PARENT(#BUTTON\_FLOW) \* The save button

DEFINE\_COM CLASS(#PRIM\_PHBN) NAME(#SAVE\_BUTTON) CAPTION(\*MTXTDF\_SAVE) DISPLAYPOSITION(1) LEFT(4) PARENT(#BUTTON\_PANEL) TABPOSITION(1) TOP(4)

\* Collection for detail fields

DEFINE\_COM CLASS(#Prim\_ACol<#prim\_evef>) NAME(#PanelFields)
\* Fields in the head area

DEFINE\_COM CLASS(#EMPNO.Visual) DISPLAYPOSITION(1) HEIGHT(19) HINT(\*MTXTDF\_DET1) LEFT(4) PARENT(#BODY\_HEAD) READONLY(True) TABPOSITION(1) TOP(4) USEPICKLIST(False) WIDTH(209)

DEFINE\_COM CLASS(#SURNAME.Visual) DISPLAYPOSITION(2) HEIGHT(19) HINT(\*MTXTDF\_DET1) LEFT(4) PARENT(#BODY\_HEAD) TABPOSITION(2) TOP(27) USEPICKLIST(False) WIDTH(324) DEFINE\_COM CLASS(#GIVENAME.Visual) DISPLAYPOSITION(3) HEIGHT(19) HINT(\*MTXTDF\_DET1) LEFT(4) PARENT(#BODY\_HEAD) TABPOSITION(3) TOP(50) USEPICKLIST(False) WIDTH(324) DEFINE\_COM CLASS(#ADDRESS1.Visual) DISPLAYPOSITION(4) HEIGHT(19) HINT(\*MTXTDF\_DET1) LEFT(4) PARENT(#BODY\_HEAD) TABPOSITION(4) TOP(73) USEPICKLIST(False) WIDTH(363) DEFINE\_COM CLASS(#ADDRESS2.Visual) DISPLAYPOSITION(5) HEIGHT(19) HINT(\*MTXTDF DET1) LEFT(4) PARENT(#BODY HEAD) TABPOSITION(5) TOP(96) USEPICKLIST(False) WIDTH(363) DEFINE\_COM CLASS(#ADDRESS3.Visual) DISPLAYPOSITION(6) HEIGHT(19) HINT(\*MTXTDF DET1) LEFT(4) PARENT(#BODY HEAD) TABPOSITION(6) TOP(119) USEPICKLIST(False) WIDTH(363) DEFINE COM CLASS(#POSTCODE.Visual) DISPLAYPOSITION(7) HEIGHT(19) HINT(\*MTXTDF\_DET1) LEFT(4) PARENT(#BODY\_HEAD) TABPOSITION(7) TOP(142) USEPICKLIST(False) WIDTH(216) DEFINE COM CLASS(#PHONEHME.Visual) DISPLAYPOSITION(8) HEIGHT(19) HINT(\*MTXTDF\_DET1) LEFT(4) PARENT(#BODY\_HEAD) TABPOSITION(8) TOP(165) USEPICKLIST(False) WIDTH(286) DEFINE\_COM CLASS(#PRIM\_ATLM) NAME(#ATLM\_1) DEFINE\_COM CLASS(#PRIM\_ATLI) NAME(#ATLI\_1) ATTACHMENT(Center) PARENT(#ATLM 1) DEFINE COM CLASS(#PRIM ATLI) NAME(#ATLI 2) ATTACHMENT(Center) MANAGE(#BODY\_HEAD) PARENT(#MAIN\_LAYOUT) DEFINE\_COM CLASS(#PRIM\_ATLI) NAME(#ATLI\_3) ATTACHMENT(Right) MANAGE(#BUTTON\_PANEL) PARENT(#MAIN LAYOUT) \* A screen wrapper is a VL reusable part of class VF\_SY122. You must define it globally scoped as opposed to inside any type of routine. DEFINE\_COM CLASS(#vf\_sy122) NAME(#myscreen\_wrapper) DISPLAYPOSITION(3) HEIGHT(227) PARENT(#COM\_OWNER) WIDTH(409) DEFINE COM CLASS(#PRIM ATLI) NAME(#ATLI 4) ATTACHMENT(Center) PARENT(#MAIN LAYOUT) DEFINE COM CLASS(#PRIM ATLI) NAME(#ATLI 6) ATTACHMENT(Center) MANAGE(#myscreen wrapper) PARENT(#MAIN LAYOUT) \_\_\_\_\_ \* Handle Initialization Mthroutine Name(uInitialize) Options(\*REDEFINE) Define Com Class(#Prim evef) Name(#FormField) Reference(\*dynamic) Invoke Method(#Com Ancestor.uInitialize) For Each(#Control) In(#Body Head.ComponentControls) If Ref Com(#Control) Is(\*INSTANCE OF #prim evef)

Set\_Ref Com(#FormField) To(\*dynamic #Control) Invoke Method(#PanelFields.Insert) Item(#FormField) Endif Endfor \* In the command's uInitialize method routine, set the screen wrapper's uCommand property to #com\_owner Set Com(#myscreen\_wrapper) Ucommand(#com\_owner) Endroutine \* Handle Command Execution \* You may also disable the entire form to prevent any input while RAMP is navigating \* \_\_\_\_\_ Mthroutine Name(uExecute) Options(\*REDEFINE) Invoke Method(#Com\_Ancestor.uExecute) #myscreen\_wrapper.makerampTSavailable Set Com(#Save\_Button) Enabled(False) #com owner.enabled := false Endroutine \_\_\_\_\_ \* Event Handlers \_\_\_\_\_ \* RAMP has signalled it's ready. Invoke your navigation here. \* Once the navigaton starts, processing resumes in the vHandleArrive event handler. Evtroutine Handling(#myscreen wrapper.RampTSAvailable) Invoke Method(#myscreen wrapper.navigatetoscreen) Name('DisplayEmployee') Endroutine Evtroutine Handling(#myscreen wrapper.vHandleArrive) Arrivedscreen(#CurrentScreen) Previousscreen(#PreviousScreen) Arrivedpayload(#Payload) Case (#CurrentScreen)

```
When Value_Is(= 'DisplayEmployee')
```

```
Set Com(#SAVE_BUTTON) Enabled(false)
```

\* Error handling:Payloads are destroyed when the ARRIVE script finishes executing.Therefore, a payload of UPDATE\_EMPLOYEE would most likely mean there was a validation error.

If (#Payload = UPDATE\_EMPLOYEE)

Use Builtin(message\_box\_show) With\_Args(ok ok info "Validation Error" "Please correct any errors")

Else

\* Unlock the framework

```
Set Com(#avFrameworkManager) Ulocked(false)
```

#myscreen\_wrapper.getvalue From('txtEmpno') Value(#empno.value) #myscreen\_wrapper.getvalue From('txtSurname') Value(#surname.value) #myscreen\_wrapper.getvalue From('txtGivename') Value(#givename.value) #myscreen\_wrapper.getvalue From('txtAddress1') Value(#address1.value) #myscreen\_wrapper.getvalue From('txtAddress2') Value(#address2.value) #myscreen\_wrapper.getvalue From('txtAddress3') Value(#address3.value) #myscreen\_wrapper.getvalue From('txtPhone') Value(#phonehme.value) #myscreen\_wrapper.getvalue From('txtPhone') Value(#phonehme.value)

#com\_owner.enabled := true

Endif

Endcase

Endroutine

\* \_\_\_\_\_

\* Handle Save

\* \_\_\_\_\_

Mthroutine Name(Save)

\* Set the 5250 field values to the values from this panel #myscreen\_wrapper.setvalue Infield('txtSurname') Value(#surname.value) #myscreen\_wrapper.setvalue Infield('txtGivename') Value(#givename.value) #myscreen\_wrapper.setvalue Infield('txtAddress1') Value(#address1.value) #myscreen\_wrapper.setvalue Infield('txtAddress2') Value(#address2.value) #myscreen\_wrapper.setvalue Infield('txtAddress3') Value(#address3.value) #myscreen\_wrapper.setvalue Infield('txtPhone') Value(#phonehme.value) #myscreen\_wrapper.setvalue Infield('txtPostcode') Value(#POSTCODE.value) \* Send the Enter key with the payload #myscreen\_wrapper.sendkey Key(#myscreen\_wrapper.KeyEnter) Payload(UPDATE\_EMPLOYEE) Endroutine

\* Listen to messages from RAMP and the 5250 application Evtroutine Handling(#myscreen wrapper.RampMessage) Umessagetype(#MsgType) Umessagetext(#MsgText) Case (#msgtype.value) When Value\_Is('= VF\_ERROR') \* Fatal messages reported by Ramp (e.g. Navigation request failed, etc). If in design mode, show the underlying 5250 screen. Otherwise, make the error message appear in a message box on top of the command If (#usystem.iDesignMode = true) Set Com(#myscreen\_wrapper) Visible(True) Else Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value) #com\_owner.avshowmessages Endif \* Messages sent by the System i application or unknown form was encountered When Value Is('= VF INFO' '= VF UNKNOWN FORM') Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value) \* Failure to initialize RAMP.Could occur for mainly one of two reasons When Value\_Is('= VF\_INIT\_ERROR') Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value) #com\_owner.avshowmessages Otherwise Use Builtin(message\_box\_show) With\_Args(ok ok info \*Component ('Unknown message type ' + #MsgType + 'encountered')) Endcase Endroutine \* Handle changes in any of the fields on the panel \* \_\_\_\_\_ Evtroutine Handling(#PanelFields<>.Changed) \* Enable the save button Set Com(#SAVE BUTTON) Enabled(True) \* Lock the framework and set a message for the user Use Builtin(bconcat) With Args('Changes made to employee' #GiveName #Surname 'have not been saved yet."Do you want to save them before continuing?')To Get(#sysvar\$av) Set Com(#avFrameworkManager) Ulocked(USER)

Ulockedmessage(#sysvar\$av) Endroutine

\* \_\_\_\_\_ \* Enter key pressed \* \_\_\_\_\_

Evtroutine Handling(#PanelFields<>.KeyPress) **Options(\*NOCLEARMESSAGES \*NOCLEARERRORS)** Keycode(#KeyCode) If Cond('#KeyCode.Value = Enter') \* If there no changes have been made issue message and ignore enter If Cond('#SAVE\_BUTTON.Enabled \*EQ True') Invoke Method(#Com\_Owner.Save) Else \* Issue 'There are no changes to save' message Use Builtin(Message\_box\_show) With\_Args(ok ok Info \*Component \*MTXTDF NO SAVE) Endif Endif Endroutine \* \_\_\_\_\_ \_\_\_\_\_ \* Handle the save button \* \_\_\_\_\_ Evtroutine Handling(#SAVE\_BUTTON.Click) #com owner.Save Endroutine \* Handle Termination \*\_\_\_\_\_ Mthroutine Name(uTerminate) Options(\*REDEFINE) \* Clean up the colelction of fields on the panel Invoke Method(#PanelFields.RemoveAll) \* Do any termination defined in the ancestor Invoke Method(#Com Ancestor.uTerminate) Endroutine

End\_Com

3. コンポーネントの[デザイン]タブを表示し、画面ラッパーのユー ザー・インターフェースを確認します。

EOMRMP03 - DisplayEmp				
社員№	ABCDE	- 保存		
社員姓	ABCDEFGHIJKLMNOPQRST			
社員名	ABCDEFGHIJKLMNOPQRST			
住所1	aAbBcCdDeEfFgGhHiIjJkKlLm			
住所2	aAbBcCdDeEfFgGhHiIjJkKlLm			
住所3	aAbBcCdDeEfFgGhHiIjJkKlLm			
郵便番号	123456			
自宅電話番号	ABCDEFGHIJKLMNO			

- 4. 画面ラッパーをコンパイルします。
- 5. フレームワークで、Employeesビジネス・オブジェクトのプロパティ を表示します。
- 6. 新しいコマンド、[Details Wrapper]を作成し、Employeesビジネス・オ ブジェクトで有効にして、iiiRMP03画面ラッパーをこれに関連付けま す。

使用可能	┌─コマンドタイプを選択して下さい─────	
Details Wrapper	🔘 শর্ভাগ্রম রস্ট্রান্স্ রব্যসং 💿 র্বাস্র্র্যম রব্যসং	
🖖 HR Details		
<u>▲</u> (^*)ト	/ 順序 1	
		_
	常にアクティブ   省略値 ▼	
❹新規	省略値のコマント・レリンス・	
💽全ての明細	Web ブラウザ Web ブラウザ	
🦫 名前		)
	▼ Windows の使用を許可	5
	▽ ポッフアップメニューの表示   文字引数 1:	
	▼ ミニツールパー上に表示 文字引数 2:	
	他のコマントないを隠す 数字引数 1:	
	アクセス制限 数字引数 2:	
	■ 非表示コマンドとして実行	
		1
	○ サンプル - RAD-PADRADPAD_BA208A8E93814AEDBC02ED658E42DAF6.HTM	
	I Contraction and the second	

- 7. フレームワークを保存して再起動します。
- 8. 従業員を選択し、従業員の[Details Wrapper]コマンド・ハンドラーを 表示して、画面ラッパーをテストします。

dia Employees							
File Edit View Help Windows (Framework) (Administration)							
Spooled Files 💥 Sign Off 🛛 📋 Reports 🔲 Details Using SETCURSOR							
Employees							
Administration   Ajax Test   ColHdg Test   Englis App   Excel   Excel   Excel   Excel   Expand-Shrink   Favorites   HR Demo Application   IPA Test   Notes CA App   Code Tables   Employees   Telephone Search   Proc_And_Close   RAMP Test   SubType Test   System i Server   Programming Techniques	By Salary By Name By Date of Clear List Employee Surname B Employee Surname Employee Number Employee Number Employee Surname Employee Given Name(s) Street No and Name Suburb or Town State and Country Post / Zip Code Home Phone Number	Birth Search Search Per (A0070-BROWI Per Documents A0070 BROWN VERONICA ANN 12 Railway Street Baulkham Hills NSW Australia 2153 TRN	Empl Description A0070 BROWN VE A0090 BLACK FRE A1031 BLOGGS JO A2005 BURGESS K A2006 BRYERS MI A2007 BERESFOR A3451 BOSS SHET N VERONICA ANNI	X RONICA ANN D HN EVIN KE JOHN D BRUCE C A Save			
	Messages Ready v	IfpgmlibNL ENG \	/LFPGMLIB 10/06/0	9 9:59 🥥			
RAMP-TSAD08 ステップ3. 画面ラッパー・コードについて このステップでは、画面ラッパーのコードを検証します。

- 1. Visual LANSAエディターで画面ラッパーのソース・コードを表示します。
- 画面ラッパーはコマンド・ハンドラーです。このため、その祖先は #VF\_AC010である必要があります。

まず、EFINE\_COMステートメントに注意します。このステートメントは、コマンド・ハンドラーが基盤の5250画面と通信できるようにする画面ラッパー・コンポーネントを定義します。



4. 次に、uInitializeイベント・ルーチンで、画面ラッパーのuCommand プロパティを#COM\_OWNERに設定するステートメントに注意しま す。



uCommandは常に#com\_ownerに設定する必要があります。そうでない場

#### 合、画面ラッパーがコマンド・ハンドラーにアクセスできないため、 VF\_INIT\_ERRORタイプのエラー・メッセージが出力されます。

5. 次に、uExecuteメソッド・ルーチンのMakerampTSavailableメソッド により、RAMPの実行が開始されます。

B-Mthroutine Name(uExecute) Options(\*REDEFINE)
Invoke Method(#Com Ancestor.uExecute)
#mvscreen wrapper\_makerampTSavailable
Set Com(#Save\_Button) Enabled(False)
#com\_owner.enabled := false
Endroutine

6. RAMPが利用可能になったら、ラッパーを移動するDestination画面を #myscreen\_wrapper.RampTSAvailableイベント・ルーチンで指定しま す。

B-Evtroutine Handling(#myscreen\_wrapper.RampTSAvailable)
Invoke Method(#myscreen\_wrapper.navigatetoscreen) Name('DisplayEmployee')
Endroutine

 最初に、vHandleArriveイベント・ルーチンは、到着した画面が DisplayEmployeeであることをチェックします。エラー処理の場合、 UPDATE\_EMPLOYEEペイロードがあるかどうかをチェックします (ユーザーが[保存]ボタンをクリックしたときに発行)。ない場合、ラッ パーは画面上のフィールドから値を取得して表示します。

E-Evtroutine Handling(#myscreen\_wrapper.vHandleArrive) Arrivedscreen(#CurrentScreen)
Previousscreen(#PreviousScreen) Arrivedpayload(#Payload)
@-Case
(#CurrentScreen)
When Value Is(= 'DisplayEmployee')
Set Com(#SAVE\_BUTTON) Enabled(false)
\* Error handling: Payloads are destroyed when the ARRIVE script finishes executing. Therefore, a payload
of UPDATE\_EMPLOYEE would most likely mean there was a validation error.
@-If (#Payload = UPDATE\_EMPLOYEE)
Use Builtin(message\_box\_show) With\_Args(ok ok info "Validation Error" "Please correct any errors")
Else
\* Unlock the framework
Set Com(#avFrameworkManager) Ulocked(false)
#myscreen\_wrapper.getvalue From('txtEmpno') Value(#surname.value)
#myscreen\_wrapper.getvalue From('txtSurname') Value(#givename.value)
#myscreen\_wrapper.getvalue From('txtAddress1') Value(#address1.value)
#myscreen\_wrapper.getvalue From('txtAddress2') Value(#address3.value)
#myscreen\_wrapper.getvalue From('txtAddress3') Value(#address3.value)
#myscreen\_wrapper.getvalue From('txtPostcode') Value(#POSTCODE.value)
#myscreen\_wrapper.getvalue From('txtPostcode') Value(#POSTCODE.value)
#com\_owner.enabled := true
Endcise
Endroutine

 画面ラッパー上の[保存]ボタンを処理するイベント・ルーチンを確認します。このイベント・ルーチンは、ラッパーフィールド上の値を 画面上のフィールドに割り当て、次にEnter キーを押す処理をエミュレートします。

£	Handle Save
t	hroutine Name(Save)
	* Set the 5250 field values to the values from this panel
1	<pre>#myscreen_wrapper.setvalue Infield('txtSurname') Value(#surname.value)</pre>
I	<pre>#myscreen_wrapper.setvalue Infield('txtGivename') Value(#givename.value)</pre>
I	<pre>#myscreen wrapper.setvalue Infield('txtAddress1') Value(#address1.value)</pre>
I	#myscreen wrapper.setvalue Infield('txtAddress2') Value(#address2.value)
I	<pre>#myscreen wrapper.setvalue Infield('txtAddress3') Value(#address3.value)</pre>
I	#myscreen wrapper.setvalue Infield('txtPhone') Value(#phonehme.value)
I	<pre>#myscreen wrapper.setvalue Infield('txtPostcode') Value(#POSTCODE.value)</pre>
I	* Send the Enter key with the pavload
l	#myscreen wrapper, sendkey Key(#myscreen wrapper, KeyEnter) Payload(UPDATE EMPLOYEE)

最後に、#myscreen\_wrapper.RampMessageイベントを処理するルーチンを確認し、RAMP-TSが画面ラッパーの各種のエラーを処理する方法を確認します。

```
* Listen to messages from RAMP and the 5250 application

@Evtroutine Handling(#myscreen_wrapper.RampMessage) Umessagetype(#MsgType) Umessagetext(#MsgText)

@Case (#msgtype.value)

"When Value_Is('= VF_ERROR')

* Fatal messages reported by Ramp (e.g. Navigation request failed, etc). If in design mode, show the

underlying 5250 screen. Otherwise, make the error message appear in a message box on top of the

command

@If (#usystem.iDesignMode = true)

Set Com(#myscreen_wrapper) Visible(True)

#Else

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value)

#com_owner.avshowmessages

=Thdif

* Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value)

* Failure to initialize RAMP. Could occur for mainly one of two reasons

- When Value_Is('= VF_INIT_ERROR')

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value)

# Com_owner.avshowmessages

- Otherwise

Use Builtin(message_box_show)

With_Args(ok ok info *Component ('Unknown message type ' + #MsgType + 'encountered'))

The result of executing expression ('Unknown message type ' + #MsgType + 'encountered') will be checked to

Endcusine

Endcusine
```

# RAMP-TSAD08 ステップ4. ラッパーからの画面の更新のテスト

#### このステップでは、以下の作業を実行します。

- 1. フレームワークの[Details Wrapper]で従業員の詳細を表示します。
- [フレームワーク]メニューで、アプリケーション・トレースをオン にします。
- 3. 従業員の姓などを変更し、[保存]ボタンを押します。
- 4. トレースを検証します。最初に画面ラッパーがどのようにラッパー 上のフィールド値を基盤の5250画面のフィールドに割り当てて、Enter キーを送信しているかに注意してください。RAMPは、次にインスタ ンス・リストを更新するFindEmployee画面に移動し、DisplayEmployee 画面に戻ります。

Application Level Trace Details					
Component	Window	Event			^
EOMRMP03	MAIN	SETVALUE of txtSurname index = 0 to value BLOGGS (type = string ) .Current	: form is DisplayEmploye	e (15:38:20:18)	
EOMRMP03	MAIN	<ul> <li>SETVALUE of txtSurname completed. Current form is DisplayEmployee (15:38:</li> </ul>	20:18)		
EOMRMP03	MAIN	<ul> <li>SETVALUE of txtGivename index = 0 to value JOHN ALAN (type = string ) .Cut</li> </ul>	rrent form is DisplayEmp	loyee (15:38:20:3	33
EOMRMP03	MAIN	SETVALUE of txtGivename completed. Current form is DisplayEmployee (15:38	):20:33)		
EOMRMP03	MAIN	SETVALUE of txtAddress1 index = 0 to value 3 Woodbury Road (type = string	) .Current form is Displ	ayEmployee (15:38	8:
EOMRMP03	MAIN	SETVALUE of txtAddress1 completed. Current form is DisplayEmployee (15:38	:20:49)		
EOMRMP03	MAIN	SETVALUE of txtAddress2 index = 0 to value Winston Hills (type = string ) .Cu	irrent form is DisplayEm	oloyee (15:38:20:4	49
EOMRMP03	MAIN	SETVALUE of txtAddress2 completed. Current form is DisplayEmployee (15:38	:20:49)		
EOMRMP03	MAIN	SETVALUE of txtAddress3 index = 0 to value NSW Australia (type = string ) .0	Eurrent form is DisplayE	mployee (15:38:20	J))
EOMRMP03	MAIN	SETVALUE of txtAddress3 completed. Current form is DisplayEmployee (15:38	:20:64)		
EOMRMP03	MAIN	<ul> <li>SETVALUE of txtPhone index = 0 to value (02) 9668 9235 (type = string ) .Cu</li> </ul>	rrent form is DisplayEmp	oloyee (15:38:20:6	54
EOMRMP03	MAIN	<ul> <li>SETVALUE of txtPhone completed. Current form is DisplayEmployee (15:38:20</li> </ul>	:80)		
EOMRMP03	MAIN	SETVALUE of txtPostcode index = 0 to value 2100 (type = string ) .Current fo	rm is DisplayEmployee (	15:38:20:80)	
EOMRMP03	MAIN	<ul> <li>SETVALUE of txtPostcode completed. Current form is DisplayEmployee (15:38)</li> </ul>	:20:80)		
EOMRMP03	MAIN	SENDKEY Enter requested. Current form is DisplayEmployee (15:38:20:96)			
EOMRMP03	MAIN	Note => A payload accompanied this SENDKEY operation. The value was UP	DATE_EMPLOYEE (15:3	8:20:96)	
EOMRMP03	MAIN	Posting screen. Current form is DisplayEmployee (15:38:20:96)			
EOMRMP03	MAIN	Executing vHandle_DEPART function in form DisplayEmployee (15:38:20:9	6)		
EOMRMP03	MAIN	Execution of script associated with DisplayEmployee completed. No error d	letected. (15:38:20:96)		
EOMRMP03	MAIN	SENDKEY Enter completed. Server response will be asynchronous. Your script	should end now.Curren	t form is DisplayEm	ip 👘
EOMRMP03	MAIN	Form with name "FindEmployee" has arrived. (15:38:20:439)			
EOMRMP03	MAIN	Executing vHandle_ARRIVE function in form FindEmployee (15:38:20:439)			
EOMRMP03	MAIN	Note => A payload accompanied this request. The value was UPDATE_Ef	MPLOYEE (15:38:20:439	9)	
EOMRMP03	MAIN	AVSIGNAL executed with WithId= Update_List_Entry To= BUSINESSOB.	JECT SendAInfo1= A10	31 SendAInfo2= 3	56
EOMRMP03	MAIN	Signalling avEvent WithId= Update_List_Entry Wait= FALSE To= BUSINESSO	BJECT SendAInfo1= A1	031	
EOMRMP03	MAIN	NAVIGATE_TO_SCREEN invoked to navigate to screen DisplayEmployee	(15:38:20:439)		
EOMRMP03	MAIN	Preparing a navigation plan to get to screen DisplayEmployee (15:38:20	1:439)		
EOMRMP03	MAIN	An optimized pre-existing navigation plan from FindEmplovee to DisplayE	Emplovee will be used. (	15:38:20:439)	
			Save Trace to File	Clear Trace	
			Save made to mile	Cical frace	

まとめ

重要なポイント

- 画面ラッパーを作成する前に、ラッパーで使用される5250画面上の すべてのフィールドに名前を付けます。
- また、ラップするDestination画面とのナビゲーション・パスがある ことも確認する必要があります。
- 画面ラッパーは、クラス#VF\_SY122を定義する、祖先#VF\_AC010 (コマンド・ハンドラー)を使用するVisual LANSAの再使用可能なパー ツです。
- 画面ラッパーで、MakerampTSavailableイベントを呼び出してRAMP の実行を開始します。RAMPが実行されたら、ルーチン処理 RampTSAvailableイベントでラッパーが移動するDestination画面を指定 します。
- ScreenWrapper.GetValueメソッドおよびScreenWrapper.SetValueメソッドを使用して、5250画面とラッパー間で値を渡します。
- RAMP以外のコマンド・ハンドラーでスナップインする場合と同様
   に、画面ラッパーをフレームワーク内にスナップインします(RAMP ツールではなく、ビジネス・オブジェクト・プロパティを使用しま す)。

覚えるべき事柄

• 基本的な画面ラッパーを作成する方法

### RAMP-TSAD09: サブファイル付きの画面ラッパー

このチュートリアルでは、画面ラッパーを拡張して、従業員の写真、お よびリスト・ビューとして表示される資格サブファイルを含めます。 目的

- 画面ラッパーでリスト・ビューとしてサブファイルを処理する方法 を学習する
- 直感的なインターフェースで画面を容易にラップできることを学習 する

💰 Employees					
Eile Edit View Help Windows (Framework) (Administration)					
Spooled Files 💥 Sign Off 🛛 🚔 Reports 🔲 Details Using SETCURSOR					
Employees					
X		X		X	
🗉 🌯 Administration 🧹			n 🕂 🖻 🔺 🙆 🗟		
🗄 👰 Ajax Test このチュー	トリアルでは、画面ラッパーを拡	張し、			
🗉 💓 CmhSort 写真と資料	格サブファイルをリスト・ビューと	17	Emplo Description		
🗄 🐞 LoiHdg Test	表示します。	Search	AUU/U BROWN VERONICA ANN	=	
Englis App			A1030 BLACK FRED	_	
Expand-Shrink RAMP	で処理した5250画面([詳細]タブ)	2	A2005 BURGESS KEVIN		
🗉 🚰 Favorites この簡素	なラッパーを比較すると、基本的	内な	A2006 BRYERS MIKE JOHN	~	
🗉 🛃 HR Demo Applica 画面でさ	らに情報を簡単に示すことがで	きる L			
🗄 👹 IPA Test	ことがわかります。	(40070 000)-//1		X	
🗄 🞽 Notes CA App		NWUND-DINUWN	YERUNICA ANNJ	<u> </u>	
± Nireelest	Details Wra	apper 📄 Documents 🛛 🦉	A Events		
Code Tables	Employee Number	A0070	and the second second	Save-	
	Employee Surpame	RDOWN	1 S	Juio	
🝶 Telephone Search	Linployee Jurname	DROWN	26		
🗉 👹 Proc_And_Close	Employee Given Name(s)	VERONICA ANN			
🗄 👰 RAMP Test	Street No and Name	12 Railway Street Baulkham Hills			
🗄 👹 SubType Test	Suburb or Town				
🗄 📑 System i Server	State and Country	NSW Australia	192	-	
🗉 🍯 VisualStyle	Post / 7in Code	2153			
🗉 🙀 XL Spreadsheet		2135			
🗄 🤯 Programming Techniques	Home Phone Number	TRN			
	Acquired Skill Code S	ikill Comment	Gra		
	1/03/98 ADMIN2 A 10/12/95 ADVPGM A 4/05/98 COM C 5/06/90 CS C 20/02/90 INDUCT C 5/02/98 INTRO C 5/02/98 KEY K 10/12/97 MANAGE1 M 1/03/98 MANAGE2 M 10/12/97 MARKET1 M 3/05/98 MARKET2 M	administra dvaninistra computer Very good re company I company I Met require. tanageme fanageme tarketing Execellent tarketing tarketing	P P D e D P P P P M M P p V		
	▼ △ Messa	ges Ready vlfpgm	nlibNL ENG VLFPGMLIB 10/06/0	19 13:16 🥥	

この目的を達成するには、以下のステップを完了してください。

RAMP-TSAD09 ステップ1. 画面ラッパーへのイメージの追加 RAMP-TSAD09 ステップ2. ラッパーへの資格リスト・ビューの追加 RAMP-TSAD09 ステップ3. リスト・ビューの値を設定するコードの追加 まとめ

始める前に

このチュートリアルを完了するには、「RAMP-TSAD08:画面ラッパーの基本」を終えている必要があります。

このチュートリアルについて

一部のシステムでは、このチュートリアルで作成した画面ラッパーによ リ、「複数のsendkey」エラーが発生することがあります。この問題は現 在調査中です。対処方法は、更新モードのときに画面に別の名前を付け ることです。例えば、到着スクリプトでF21 keypressを使用する代わりに UpdateEmployeeを使用して、表示モードと更新モードで画面を区別しま す。 **RAMP-TSAD09 ステップ1. 画面ラッパーへのイメージの追加** このステップでは、資格情報の検索 / 保守画面ラッパーにイメージを追 加します。

この例では、フレームワークに付属の標準的な.gifファイルを使用しま す。実際のアプリケーションでは、サーバーのファイルに従業員の写真 が格納されており、その写真をその他の従業員の詳細情報と一緒に取得 するなどのケースが考えられます。

- 1. Visual LANSAエディターでiiiRMP03画面ラッパーを表示します。
- 2. [デザイン]タブを表示します。
- イメージ・コントロールを[お気に入り]の[共通の制御]タブから画面 にドラッグします。

これは、画面を管理するレイアウト・マネージャによって自動的に画面 ラッパーのフィールドに配置されます。ここでは、イメージを手動で配 置します。

- 4. [ソース]タブを表示し、イメージとそのために作成されたレイアウト・アイテムのDEFINE\_COMステートメントを見つけます。
- フロー・レイアウト・アイテムを定義しているステートメントをコ メント・アウトします。

□ DEFINE\_COM CLASS(#PRIM\_IMGE) NAME(#IMGE\_1) DISPLAYPOSITION(9) HEIGHT(117) LEFT(4) PARENT(#BODY\_HEAD) TABPOSITION(9) TOP(188) WIDTH(108) \*DEFINE\_COM CLASS(#PRIM\_FWLI) NAME(#FWLI\_1) MANAGE(#IMGE\_1) PARENT(#BODY\_HEAD\_FLOW)

[デザイン]タブに切り替えて、以下のようにイメージを移動して配置します。



7. 画面ラッパーのフィールドの値を取得したら、[ソース]タブを表示し、以下のステートメントを追加し、イメージ・ファイルをイメージ・コントロールに割り当てます。

Set Com(#imge\_1) Filename('C:\Program Files\LANSA\LANSA\Imports\VLFRAMEWORK\df\_im001.gif')

ここで、パス情報をdf\_im001.gifが存在するシステムのパスに置き換える 必要があります。コードは以下のようになります。

<pre>-Else     #com_owner.enabled := true     #nyscreen_wrapper.getvalue From('txtEmpno') Value(#empno.value)     #myscreen_wrapper.getvalue From('txtSurname') Value(#surname.value)     #myscreen_wrapper.getvalue From('txtGivename') Value(#givename.value)     #myscreen_wrapper.getvalue From('txtAddress1') Value(#address1.value)     #myscreen_wrapper.getvalue From('txtAddress3') Value(#address3.value)     #myscreen_wrapper.getvalue From('txtPhone') Value(#phonehme.value)     #myscreen_wrapper.getvalue From('txtPhone') Value(#address3.value)     #myscreen_wrapper.getvalue From('txtPhone') Value(#phonehme.value)</pre>
Set Com(#imge_1) Filename('C:\Program Files\LANSASP5Final\LANSA\Imports\VLFRAMEWORK\df_im001.gif') Endif

- 8. 画面ラッパーをコンパイルします。ラッパーが使用されているため にコンパイルが失敗した場合は、フレームワークを再起動します。
- 9. フレームワークで、従業員の[Details Wrapper]を表示します。

x				
👹 Employee : Details Wrapper (A0070-BROWN VERONICA ANN)				
Details Details Wrapper Documents A Events				
Employee Number	A00705	ave		
Employee Surname	BROWN			
Employee Given Name(s)	VERONICA ANN			
Street No and Name	12 Railway Street			
Suburb or Town	Baulkham Hills			
State and Country	NSW Australia			
Post / Zip Code	2153			
Home Phone Number	TRN			

10. フレームワークを閉じます。

RAMP-TSAD09ステップ 2. ラッパーへの資格リスト・ビューの追加

このステップでは、資格情報の検索/保守画面に資格サブファイルを表示するリスト・ビューを画面ラッパーに追加します。

「RAMP-TSAD08 ステップ1. ラッパーで使用されるフィールドに名前を 付ける」で、サブファイル列およびプラス記号に名前を付けていることに 注意してください。

- 1. Visual LANSAエディターで画面ラッパー・ユーザー・インター フェースを表示します。
- 2. リスト・ビュー・コントロールを画面ラッパーにドラッグし、以下 のようにサイズを設定します。

EOMRMP03 - DisplayEmp	oloyee Wrapper
社員N♀	ABCDE
社員姓	ABCDEFGHIJKLMNOPQRST
社員名	ABCDEFGHIJKLMNOPQRST
住所1	aAbBcCdDeEfFgGhHiIjJkKlLm
住所2	aAbBcCdDeEfFgGhHiIjJkKlLm
住所3	aAbBcCdDeEfFgGhHiIjJkKlLm
郵便番号	123456
自宅電話番号	ABCDEFGHIJKLMNO

3. Skillsという名前を付けます。

詳細 🗳			
Skills - リストビューはリストを表示するためのコンポーネントです。			
<mark>7<sup>°</sup>፲</mark> ハ <sup>°</sup> ティ イベント   メンッド			
📝 Height	125	~	
🖻 Hint			
🖻 HintShow	True		
PhintShowOfParent	True		
🖻 IconAlignment	Left		
🖻 KeyboardPositioning	SortColumn		
🖻 Left	4		
🖻 ModifiedRules			
🖻 Name	Skills		
🖉 NotificationStyle	Default		
🦻 Owner	#EOMRMP03		
🖻 Parent	#BODY_HEAD		
🖻 PopupMenu	*NULL		
PopupMenuFocusSelect	True		
🖉 SelectedRowCount	0		
SelectionStyle Multiple			

- 4. リポジトリでPSLSKLファイルを見つけて、ファイルの[DATEACQ] フィールドと[SKILCODE]フィールドをリスト・ビューにドラッグし ます。
- 5. リポジトリでSKLTABファイルを見つけて、[SKILDESC]フィールド をリスト・ビューにドラッグします。
- 6. 最後に、[COMMENT]フィールドと[GRADE]フィールドをPSLSKL ファイルからリスト・ビューに追加します。
- リスト・ビューの外観は、以下のようになります。

EOMRMP03 - DisplayEm	oloyee Wrap	per	
社員№	ABCDE		
社員姓	ABCDEFGHIJ	KLMNOPQRST	
社員名	ABCDEFGHIJ	KLMNOPQRST	
住所1	aAbBcCdDeE	fFgGhHiIjJkKlLm	
住所2	aAbBcCdDeE	ifFgGhHiIjJkKlLm	
住所3	aAbBcCdDeE	ifFgGhHiIjJkKlLm	
郵便番号	123456		
自宅電話番号	ABCDEFGHIJKLMNO		
取得日(YY 資格コード	資格記述	資格取得に関	資格等級
→ 12/3 ABCDEFG	ABCDEFG	ABCDEFGHIJK	Α
<	1111		>

7. ラッパーを保存します。

RAMP-TSAD09 ステップ3. リスト・ビューの値を設定する コードの追加

このステップでは、リスト・ビューの値を設定するコードを追加しま す。

- 1. 画面ラッパーのソース・コードを表示します。
- #myscreen\_wrapper.RampTSAvailableイベント・ルーチンを見つけて、このコードを追加し、画面が到着したときに画面ラッパーのフィールドとサブファイルをクリアします。

Clr\_List Named(#skills) #XG\_HEAD := \*null

コードは、以下のようになります。

```
Fevtroutine Handling(#mvscreen_wrapper.RampTSAvailable)
Clr_List Named(#skills)
#XG_HEAD := *null
Invoke Method(#myscreen_wrapper.navigatetoscreen) Name('DisplayEmployee')
Endroutine
```

# 3. 画面ラッパー・ソースの最後に以下のメソッド・ルーチンを追加します。

Mthroutine Name(uGetSubfilePage)

Define\_Map For(\*result) Class(#prim\_boln) Name(#NextPage) Define Field(#MoreVal) Type(\*char) Length(1)

#listcount := 1 #NextPage := false

Dowhile (#myscreen\_wrapper.check\_field\_exists( "dateacq" #listcount ))

```
#myscreen_wrapper.getvalue From("dateacq") Value(#vf_eltxts)
Defaultvalue(#ddmmyy) Index(#listcount)
#myscreen_wrapper.getvalue From("skillcode") Value(#skilcode)
Index(#listcount)
#myscreen_wrapper.getvalue From("skilldesc") Value(#skildesc)
Index(#listcount) Defaultvalue("Defalt value")
```

#myscreen\_wrapper.getvalue From("comment") Value(#comment)
Index(#listcount)
#myscreen\_wrapper.getvalue From("grade") Value(#grade) Index(#listcount)

\* You can put some tracing #com\_owner.avframeworkmanager.avRecordTrace Component(#com\_owner) Event("Adding entry = " + #vf\_eltxts + ", " + #skilcode + ", " + #skildesc + ", " + #grade)

Add\_Entry To\_List(#skills)

#listcount += 1

Endwhile

\* when identifying this screen we set the name of the "+" sign = "moreindicator". The presence of that field in the last row of the subfile tells us whether there is another page. The last row is one less than the current value of #listcount.

#listcount -= 1

```
If (#myscreen_wrapper.check_field_exists( "moreindicator" #listcount ))
#myscreen_wrapper.getvalue From("moreindicator") Value(#MoreVal)
Index(#listcount)
#NextPage := (#MoreVal.trim *NE "")
Endif
```

Endroutine

このコードは、まずDATEACQフィールドに対応する最初の列が存在す るかどうかをチェックして、資格サブファイルを解析します。画面に存 在する場合、このルーチンはすべてのサブファイル列の値を取得しま す。

RAMP-TSサブファイルのフィールドには、1から始まるインデックスが 付けられます。7つの行を含むサブファイル・ページは、サブファイルの 各フィールドに7つのインスタンスを含みます。ここでは、#listcountを増 やして使用し、フィールドの値を取得します。

getvalueメソッドを使用するときは、デフォルト値を指定できることに注

意してください。

moreindicatorは、画面のプラス記号に付けた名前です。

- 4. ここで、#myscreen\_wrapper.vHandleArriveイベント・ルーチンを見つ けます。
- 5. 資格サブファイルにさらにレコードがあるかどうかを確認するラッパーで使用される、ブール型クラス#MoreRecordsを定義します。

Define\_Com Class(#prim\_boln) Name(#MoreRecords)

6. 次に、このコードを追加して、uGetSubfilePageメソッドで返される 値をチェックし、さらにレコードがある場合は、次のサブファイル・ ページを取得するPageDownキーを送信します。

#MoreRecords := #com\_owner.uGetSubfilePage If (#MoreRecords) if (#previousscreen = 'DisplayEmployee') #myscreen\_wrapper.sendkey Key(#myscreen\_wrapper.KeyPageDown)

(以前の画面のチェックにより、画面ラッパーの処理が遅くなります。このため、高速なマシン上でSENDKEY要求を繰り返し実行すると、これらの要求は適切に待ち行列に入れられます。) コードは、以下のようになります。



- 7. 画面ラッパーをコンパイルします。
- 8. フレームワークを起動し、従業員の[Details Wrapper]コマンド・ハンドラーを表示します。

👹 Employee : Details Wra	apper (A0070-BROWN VERONICA ANN)					
🗖 Details 📃 Details Wrap	🔽 Details 💭 Details Wrapper 📄 Documents 🔥 Events					
Employee Number	A0070 Save					
Employee Surname	BROWN					
Employee Given Name(s)	VERONICA ANN					
Street No and Name	12 Railway Street					
Suburb or Town	Baulkham Hills					
State and Country	NSW Australia					
Post / Zip Code	2153					
Home Phone Number	TRN					
Acquired Skill Code Skill	Comment Grade Obtai					
25/03/98 ADMIN1 Admir 1/05/98 ADMIN2 Admir 4/05/98 COM Comm 5/05/98 CS Comp 30/06/96 ADMIN1 Admir 1/03/98 ADMIN2 Admir 10/12/95 ADVPGM Advan	nistratn Met req D					

まとめ

重要なポイント

画面ラッパーでリスト・ビューにサブファイル・レコードを表示し、ページを下に移動する必要がないようにすることができます。

覚えるべき事柄

- イメージなどのコンポーネントを追加して画面ラッパーを拡張する 方法
- 画面ラッパーでリスト・ビューとしてサブファイルを表示する方法

# スクリプティング

RAMPでは、スクリプトを使用して刷新されたアプリケーションで5250 画面を管理します。

学習

使用方法

デバッグ

## 学習

「スクリプトの構造」では、RAMPスクリプトを紹介します。 「Javascriptの基本要素」では、スクリプトを書く場合によく使用するい くつかの基本テクニックを説明します。

アプリケーション・レベルの追跡

フレームワーク・メニューのトレース・オプションを使用してアプリケーション・レベルのトレースを開始します。

# スクリプトの構造

## 分類されている画面には、1つのスクリプトが関連付けられています。ス クリプトは、常に以下のもので構成されています。



Special画面のスクリプト Junction画面のスクリプト Destination画面のスクリプト vHandle\_NAVIGATETOファンクション vHandle\_ARRIVEファンクション RAMP-TS画面スクリプトによるJavaScriptオブジェクトの定義

# Special画面のスクリプト

## 通常、SPECIAL画面に関連付けられているスクリプトは以下の構造を備 えています。

€ ←	【で始めて、JavaScriptオブジェクトを定義します。
<pre>vHandle_ARRIVE: function(oPayload, oPreviousForm) ← {      &lt;&lt; logic - discussed in detail later &gt;&gt; },</pre>	<ul> <li>vHandle_ARRIVEというファンクションを含んでいます。</li> <li>このファンクション内で生成されるまたはコーディングする</li> <li>ロジックは、Special画面が到着したときに発生する内容を定義します。</li> <li>通常、Special画面はキーを送信し、即座に消えます。</li> <li>例えば、ユーザーのログオン時に表示される5250のメッセージの 表示画面は、Special画面として扱われ、Enterキーを5250サーバー</li> <li>に送信すると、この画面は即座に消えます。</li> </ul>
<pre>//<sysinfo> etc, etc //</sysinfo> }</pre>	このブロック内のコードを変更しないでください。

# Junction画面のスクリプト

## 通常、JUNCTION画面に関連付けられているスクリプトは以下の構造を 備えています。

<pre>{</pre>	<ul> <li>「で始めて、JavaScriptオブジェクトを定義します。</li> <li>vHandle_NAVIGATETOファンクションを含んでいます。</li> <li>このファンクション内で生成されるまたはコーディングする ロジックは、Junction画面に別の画面への移動が要求された ときに発生する内容を定義します。</li> </ul>
<pre>vHandle_ARRIVE: function(oPayload, oPreviousForm)</pre>	<ul> <li>vHandle_ARRIVEファンクションを含んでいます。</li> <li>このファンクション内のロジックは、Junction画面が到着した ときに発生する内容を定義します。</li> <li>Junction画面が到着ロジックを持つことはまれです。</li> </ul>
// <sysinfo> etc, etc //</sysinfo> }	このブロック内のコードを変更しないでください。 

# Destination画面のスクリプト

## 通常、DESTINATION画面に関連付けられているスクリプトは以下の構 造を備えています。

{ •	〔で始めて、JavaScriptオブジェクトを定義します。
<pre>vHandle_ARRIVE: function(oPayload, oPreviousForm)  {      &lt;&lt; logic - discussed in detail later &gt;&gt; },</pre>	■ vHandle_ARRIVEファンクションを含んでいます。 このファンクション内のロジックは、Destination画面が到着した ときに発生する内容を定義します。
<pre>vHandle_BUTTONCLICK: function(sButton) {      &lt;&lt; logic - discussed in detail later &gt;&gt; },</pre>	vHandle_BUTTONCLICKファンクションを含んでいます。 このファンクション内のロジックは、ユーザーがこのDestination 画面でボタンをクリックしたり、ファンクション・キーを使用した ときに発生する内容を定義します。
<pre>vHandle_NAVIGATETD: function(sToForm, oPreviousForm) {      &lt;&lt; logic - discussed in detail later &gt;&gt; },</pre>	vHandle_NAVIGATETOファンクションを含んでいます。 このファンクション内で生成されるまたはコーディングする ロジックは、Destination画面に別の画面への移動が要求された ときに発生する内容を定義します。
// <sysinfd> etc, etc //</sysinfd>	―― このブロック内のコードを変更しないでください。
} ←	一一」で終わります。

## vHandle\_NAVIGATETOファンクション

これは、サンプル画面"JUNCTION\_A"のvHandle\_NAVIGATETOファン クションです。このファンクションは、RAMPフレームワークからのナ ビゲーション要求を処理しま

す。"JUNCTION\_A"は、"DESTINATION\_B"および"JUNCTION\_C"に移動 する要求を処理できると考えてください。

<pre>vHandle_NAVIGATETD: function(sToForm, oPreviousForm) {     var bReturn = true;     HIDE_CURRENT_FORM();     STENSY(frum);</pre>	<ul> <li>現在のフレームワーク・フォームを非表示にして</li> <li>ナビゲーションを非表示にし、フレームワークをビジー</li> <li>状態にします。</li> </ul>
switch (sToForm) { /* <navigateswitch> - Do not remove or alter this lin</navigateswitch>	<ul> <li>この特殊なタグの行を削除または移動してはなりません。</li> <li>これは、このスクリプト内への新しいナビゲーションの 挿入場所をRAMPツールに指定します。</li> <li>*/</li> </ul>
<pre>case "DESTINATION_B"+ {     SETVALUE("menuoption","2");     SENDKEY(KeyEnter);     Q_CHECK_CURRENT_FORM("DESTINATION_B",etc) } break;</pre>	メニュー・オプションを2に設定し、Enterキーを送信して、 結果として正しいフォームが到着したことを確認することで、 "DESTINATION_B"画面への移動要求を処理します。
<pre>case "JUNCTIDN_C"# {     SETVALUE("menuoption","47");     SENDKEY(KeyEnter);     Q_CHECK_CURRENT_FORM("JUNCTION_C",etc) } break;</pre>	- メニュー・オブションを47に設定し、Enterキーを送信して、 結果として正しいフォームが到着したことを確認することで、 "JUNCTION_C"画面への移動要求を処理します。
<pre> <etc, etc="">  }  return(bReturn);  },</etc,></pre>	— 成功または失敗を示すブール値をRAMPフレームワークに 返します。

## vHandle\_ARRIVEファンクション

これは、サンプル画面"DESTINATION\_B"のvHandle\_ARRIVEファンク ションです。このファンクションは、"DESTINATION\_B"が表示される たびに実行されます。

<pre>vHandle_ARRIVE: function(oPayload, oPreviousForm)  {      var bReturn = true;</pre>	vHandle_ARRIVEファンクションは、オプションのペイロード、 および前に表示された画面への参照を受け取ります。 詳細は後述の内容を参照してください。
SHOW_CURRENT_FORM(true);	- 現在のフレームワークRAMP画面を表示し、フレームワークで ファンクション・キーをボタンとして表示します。
HEDE_5250_BUTTONS(); <	-5250画面で表示されているすべてのボタンを非表示にします。
SETBUSY(false);	ビジー状態を終了させ、ユーザーの対話処理を可能にします。
/* <arrive></arrive> - Do not remove or alter this line */ ←	- この特殊なタグの行を削除または移動してはなりません。 これは、このスクリプト内への新しいコードの挿入場所を RAMPツールに指定します。
return(bReturn);	<ul> <li>成功または失敗を示すブール値をRAMPフレームワークに 返します。</li> </ul>

## RAMP-TS画面スクリプトによるJavaScriptオブジェクトの定 義

RAMP-TS画面ごとに作成するスクリプトは、JavaScriptオブジェクトを定 義します。他のJavaScriptオブジェクトと同様に、このスクリプトはファ ンクションとプロパティを含んでいます。

標準で付属のRAMP-TSファンクションとプロパティは次のとおりです (一般的に使用される機能を太字で示しています)。

名称	タイプ	スリト変可かう	説明/コメン ト
vFKC	文字 列の 配列	いい え	ファンク ション・ キーのキャ プション
vFKERTS	文字 列	いい え	5250画面で のファンク ション・ キーの使用 可能性
vFKEVLF	文字 列	いい え	RAMP-TSで のファンク ション・ キー/ボタン の使用可能 性
vFKSEQ	文字 列	いい え	ファンク ション・ キーの順序 番号

vFKSND	文字 列の 配列	いい え	5250サー バーに送信 されるファ ンクショ ン・キー
vGUID	文字 列	いい え	画面定義の GUID
vHandle_ARRIVE	ファ ンク ショ ン	N/A	画面の到着 処理ファン クション
vHandle_AVEVENT	ファ ンク ショ ン	N/A	画面のVLF イベント処 理ファンク ション
vHandle_BUTTONCLICK	フンシン	N/A	画のボタ ン・クリッ ク/ファンク ション・ キー アン ション ション
vHandle_DEPART	ファ ンク ショ ン	N/A	画面の出発 処理ファン クション
vHandle_NAVIGATETO	ファ ンク ショ ン	N/A	画面のナビ ゲーション 処理ファン クション
vHandle_USER_NAVIGATION_PLAN	ファ ンク ショ ン	N/A	画面ナビ ゲーショ ン・プラン のオーバー

			ライド・ ファンク ション
vLastMessage	文字 列	いい え	この画面に 到着した最 新のメッ セージ
vLatestVariant	文字 列	いい え	到着した最 新の画面バ リアント
vName	文字 列	いい え	画面の名前
vOverrideExitJunction	文字列	はい	この画面に 関連付けら れている Exit Junction をオーバー ライドしま す。
vTargets	文字 列の 配列	いいえ	この画面か ら移動でき る画面
VTYPE	文字 列	いい え	画面のタイ プ

独自のファンクションやプロパティを追加できます。

注:独自のファンクションやプロパティの先頭に"v"(小文字)や"V"(大文字)を使用しないでください。v\*/V\*ネームスペースは、標準で付属のファンクションやプロパティの将来的な拡張用にRAMP-TS製品で予約されています。

## 従来の詳細表示でのスクリプト

この例では、2つの非常に単純な5250画面をRAMPで刷新する方法を説明します。

1番目の画面GETORDERは、受注番号の入力を要求し、2番目の画面 SHOWORDERは、受注の詳細を表示します。



ユーザーは、F12ファンクション・キーを使用して受注の照会を繰り返します。

刷新バージョン

アプリケーションを刷新するには、5250画面を識別して、その画面の操作のスクリプトをRAMPに定義します。

- GETORDER画面はJunction画面になります。この画面は表示されません。
- SHOWORDER画面はDestination画面になります。

Order		Sec. 1				
Filter for Orders This fiber would allow you to b orders in many ways. It is not what your existing \$250 applic moment. For example, it might => Orders for a customer => Orders for aip code	uild instance lists of constained in any way be also can do at the build lists of:	Order     O	Description     Order number 1     Order number 2     Order number 3     Order number 4     Order number 5     Order number 5     Order number 6     Order number 7     Order number 9     Order number 9			GETORDER Junction 5250画面
	Emulate Sec	eth		Overlat		
🌽 Order i Detala (SRDER0003-Ord	ler number ()				0	and the second second
5250 Display Order Screen	Goes Bern				01	Order Number :
This panel will display the \$250			Ded fo	rom the instance		1
list above. It will have no fund	Order Numb	er : 30003	XXXXX	maily to display		
an order.	Customer : )	0000000	x00000x			
	Address : XX	0000000	00000			/
	Phone : XXX	0000000	xxxxx			/
	Zip : XXXXXX	00000000	X0000K			ORDERNUMBERフィールド
		n an			-	
		1				
	SHO	NORDER				

Destination 5250画面

刷新されたアプリケーションでは、ユーザーがインスタンスリストから 受注を選択すると、SHOWORDER画面に選択した受注の詳細が表示され ます。

GETORDERには、vHandle\_NAVIGATETOファンクションを含むスクリプ トがあります。このファンクションはSHOWORDER画面を表示します。 SHOWORDER画面には、以下を含むスクリプトがあります。

- vHandle\_NAVIGATETOファンクション。これには、SHOWORDER 画面をキャンセルするコードが含まれています。
- vHandle\_BUTTONCLICKファンクション。すべてのファンクション・キーはSHOWORDER画面で表示されず、使用不可に設定されているので、このファンクションはSHOWORDERに対して実行されません(ユーザーは、インスタンス・リストの別の受注をクリックするだけで、別の受注の詳細を表示できます)。
- ユーザーがインスタンス・リストの受注をクリックするたびに、 SHOWORDERのvHandle\_NAVIGATETOファンクションがナビゲー ション・ネットワークに戻るために実行されます。そこでは、選択された受注の詳細を示すSHOWORDERS画面を表示するためのナビゲー ション・パスが作成されます。

このタイプのナビゲーションの詳細については、チュートリアル 「RAMP-TS009:ナビゲーションの追跡」を参照してください。

## Javascriptの基本要素

RAMPは、JavaScriptスクリプトを使用して、現代のアプリケーション内での5250画面を管理します。

JavaScriptは世界で最も一般的に使用されているスクリプト言語です。 MicrosoftのJScript拡張も使用できます。JavaScriptのスキルは、LANSA for the WebやHTML操作など、その他多くの状況で使用できます。

このセクションではJavascriptの基本要素についていくつか説明します。
 JavaScriptに関する外部ドキュメント
 Javascriptの基本構文
 値の読み込み、書き出し、保存
 alert()
 数字から文字列への変換
 文字列から数字への変換
 文字列操作ファンクション
 変数が数字か文字列かを見分ける方法
 JavaScriptのコード・スタイル
 objGlobalオブジェクトの使用方法

## JavaScriptに関する外部ドキュメント

JavaScriptの公式ドキュメントにアクセスするには、次のサイトをクリックしてください。

http://www.w3schools.com/jsref/

また、多くの優れた書籍も利用できます(Danny Goodmanの『JavaScript Bible (ISBN 0-7645-3188-3)』など)。

## Javascriptの基本構文

- コメントは/\* \*/でマーク付けされています。
- 行はセミコロン(;)で終了します。
- リテラルは二重引用符(")で囲まれています。
- スクリプトが参照できるフレームワーク・オブジェクトについて は、「スクリプトが参照できるフレームワーク・オブジェクト」を参 照してください。
- 条件切り替えステートメントの構造は次のとおりです。

```
switch(n)
{
case 1:
    execute code block 1
    break
case 2:
    execute code block 2
    break
default:
    code to be executed if n is
    different from case 1 and 2
}
```

値の読み込み、書き出し、保存

値の読み込み

スクリプトでは、次のようにインスタンス・リストから値を読み込むこ とができます。

myVariable = objListManager.Akey3[0];

「ビジュアル識別子とプログラム的識別子」を参照してください。

ユーザーがインスタンス・リストのいくつかのエントリーを選択してい る場合、次のようにループ内ですべての値を読み込むことができます。 var i = 0; var strMessage = ""; for (i = 1; i <= objListManager.TotalSelected; i++) { strMessage += "Selected Employee " + objListManager.AKey3[i] + " "; } alert(strMessage);

または、5250画面に定義されたフィールドからは次のように読み込むこ とができます。

MyVariable = GETVALUE("utxtEmployeeCode");

## 値の書き出し

スクリプトでは、次のように値を画面に挿入することができます。 SETVALUE("utxtEmployeeCode", "myText");

#### 値の保存

Javascript変数の値を保存し、次にその値を読み込んだり書き出すことが できます。

```
Var MyString = "";
```

```
MyString = objListManager.Akey3[0];
```

この値はスクリプトが実行されている間のみ存在します。スクリプト間
で情報を共有するには、objGlobalのプロパティを作成および設定する必要があります。

objGlobal.uLastValue = "anything";

これで別のスクリプトがこの値を読み込むことができます。 myVariable = objGlobal.uLastValue;

スクリプトの一部を簡単に取得する スクリプト・ポップアップ・メニュー

# alert()

alert()ファンクションは、誤ったスクリプトをデバッグするのに最も役に 立つツールです。

例:

alert("About to send the enter key"); alert("The value of x is " + x.toString()); alert("The customer number is " + objGlobal.CustomerNumber); さらに、次のトピックも参照してください。

スクリプトの不明な動作

**Object expected** 

# 数字から文字列への変換

JavaScript変数に含まれている数字を文字列に変換する場合、toString() ファンクションを使用します。例:

var number = 5.65; var stringnumber = number.toString();

alert(stringnumber);

SETVALUE("Amount",stringNumber); SETVALUE("Amount",number.toString());

#### 文字列から数字への変換

文字列を数字に変換する場合、parseInt()メソッドを使用します。たとえば、次のスクリプトでは1234と43を含む整数値がそれぞれのXに返されます。

X = parseInt("1234",10);

X = parseInt("34abc",10);

2番目の引数の(10)は、十進法を使用することを示します。このパラメー タに10以外の値を指定することは一般的ではありません。デフォルトは 予測できないため常にこの値を指定する必要があります(詳しい理由につ いては、たとえば、http://www.w3schools.com/jsref/jsref\_obj\_global.aspなど を参照してください)。

少数部を持つ必要がある場合は、parseFloat()を使用します。例えば、次のスクリプトでは1234.345と34.7を含む浮動少数点数値がそれぞれのXに返されます。

X = parseFloat("1234.345");

X = parseFloat("34.7abc");

これらの値は浮動小数点数値です。したがって、符号付き10進数値や パック10進数値のように必ずしも正確または予測可能とは限りません。

# 文字列操作ファンクション

JavaScriptの文字列変数には、多くの役に立つ文字列ファンクションがあ ります。最も一般的な使用方法の例を次に示します。

操作 / ファンクション	例
連結 (+)	var S1 = "Customer"; var S2 = "123456"; var S3 = S1 + " " + S2 + "could not be found"; <i>Customer 123456could not be found</i> が変数S3に 挿入されます。
<b>IndexOf</b> – 文字列内での 文字列の最初の出現を検 索する	/* 012345678901 */ var S1 = "ABCDHELLOABC"; var pos = S1.indexOf("HELLO"); 数字4が変数posに挿入されます。
<b>lastIndexOf</b> – 文字列内で の文字列の最後の出現を 検索する	/* 012345678901 */ var S1 = "ABCDHELLOABC"; var pos = S1.lastIndexOf("AB"); 数字9が変数posに挿入されます。
charAt – 文字列内の特定 の位置の文字を返す	<pre>/* 012345678901 */ var S1 = "ABCDHELLOABC"; var S2 = S1.charAt(4); var S3 = S1.charAt(9); "H"がS2に挿入され、"A"がS3に挿入されま す。</pre>
length – 文字列の長さを 返す	/* 012345678901 */ var S1 = "ABCDHELLOABC"; var I = S1.length; 数字11が変数Iに挿入されます。
substring – 開始位置と終 了位置を使用して文字列 のサブストリングを返す	/* 01234567789 */ var a = "Hello World"; var b = a.substring(4,8); "o Wor"がbに挿入されます。
substr – 開始位置と長さ	/* 01234567789 */

を使用して文字列のサブ ストリングを返す	var a = "Hello World"; var b = a.substr(2,3); "llo"がbに挿入されます。
toLowerCase – 小文字の 文字列を返す	var a = "Hello World"; var b = a.toLowerCase(); "hello world"がbに挿入されます。
toUpperCase – 大文字の 文字列を返す	var a = "Hello World"; var b = a.toUpperCase(); "HELL WORLD"がbに挿入されます。

このような文字列ファンクションは他にもあります。詳細は以下を参照 してください。

http://www.w3schools.com/jsref/jsref\_obj\_string.asp

#### 変数が数字か文字列かを見分ける方法

Javascript内の変数が数字か文字列か分からない場合があります。このような場合は、次のようにtypeof()演算子を使用して変数のタイプをテストできます。

Var x = 1.234; Var y = "Hello"; Var Type1 = typeof(x); Var Type2 = typeof(y);

Alert(Type1 + " and " + Type2);

このコードはメッセージ"number and string"を表示します。

typeofから返される可能性のある値

は、"number"、"string"、"boolean"、"object"、"function"、"undefined"の6 個です。最も役に立つ値は、"number"、"string"、"undefined"です。 "undefined"は、まだ存在していない(つまり未定義である)ということを示

すため役に立ちます。たとえば、次のようなコードで使用される場合が あります。

if (typeof(objGlobal.CustomerNumber) = "undefined"))
objGlobal.CustomerNumber = "12345";

# JavaScriptのコード・スタイル

JavaScriptで、RAMP-TSスクリプトをコーディングする場合、以下のコード・フラグメントはすべて標準であり等価です。

x = new Object(); x.a = 1; x.b = "Hello";

 $x = \{ a : 1, b : "Hello" \}$ 

x = { "a" :1, "b" :"hello" } <====== which is the JavaScript format that was chosen for use in AJAX-JSON strings.

-----

-----

-----

-----

-----

x = { }; x["a"] = 1; x["b"] = "hello";

x = new Object(); x["a"] = 1; x["b"] = "hello";

# objGlobalオブジェクトの使用方法

objGlobalは、スクリプトが参照できるフレームワーク・オブジェクトの1 つです。その目的は、独自のプロパティを格納することです。 このセクションではこのオブジェクトを使用する際のテクニックをいく つか説明します。

#### 整理する

objGlobalを使用してオプション・パラメータを渡す objGlobalを使用して共通使用されるファンクションを定義する 整理する

objGlobalオブジェクトを頻繁に使用する場合、その使用方法を何らかの 方法で整理することに気を付ける必要があります。1つの方法は、アプリ ケーションまたは使用方法により複数のサブオブジェクトに分けること です。

たとえば、次のようなコードをログオン・スクリプトに含めた場合:

objGlobal.AppA = new Object();

objGlobal.AppB = new Object();

objGlobal.AppC = new Object();

参照がスクリプト内で互いに誤って干渉しないようにできます。

例えば、objGlobal.AppA.CurrentCustomerは、

objGlobal.AppB.CurrentCustomerやobjGlobal.AppC.CurrentCustomerとは異なる変数です。

# objGlobalを使用してオプション・パラメータを渡す

前のセクションの考え方を少し広げると、スクリプトに渡すオプション・パラメータの概念を導入することができます。いくつかのオプション・パラメータを別のスクリプトに渡す必要のあるスクリプトでは、次のようなコードを含めることができます。

objGlobal.OptParms = new Object(); objGlobal.OptParms.CustNumber = "12345"; objGlobal.OptParms.CustName = "ACME ENGINEERING"; NAVIGATE TO SCREEN("uShowCustomer");

また、このオプション・パラメータを受け付けるスクリプトでは、次の ような構造のコードを含めることができます。

```
var CustNumber = "some default value";
var CustName = "some default value";
```

```
if (objGlobal.OptParms != null)
{
CustNumber = objGlobal.OptParms.CustNumber;
CustName = objGlobal.OptParms.CustName;
objGlobal.OptParms = null;
}
```

/\* Now we proceed to use the values in CustNumber and CustName \*/

このスタイルの処理では、objGlobal.OptParms = null;の行が非常に重要 です。この行は、一時オブジェクトOptParmsを破棄するためです。

# objGlobalを使用して共通使用されるファンクションを定義する

多くの場所で再利用するJavaScriptファンクションを作成する場合、サイ ンオン・スクリプトにたとえば次のような処理を含めることができま す。

objGlobal.Mult = function (x,y) {
 var z = x \* y;
 return(z); }
objGlobal.Add = function (x,y) {
 var z = x + y;
 return(z); }

この操作では、MultとAddという名前でobjGlobalに2つのファンクション が定義され、それぞれのファンクションに含まれるコードが定義されて います。

これが完了すると、objGlobal.AddとobjGlobal.Multのそれぞれのファンク ションは他のスクリプトで次のように実行できます。

```
var q = objGlobal.Add(222,3);
alert( q.toString() );
```

q = objGlobal.Mult(22,33); alert( q.toString() );

この結果、225と726がそれぞれ表示されます。

使用方法

スクリプトを使用したインスタンス・リストとの対話 スクリプト・ポップアップ・メニュー RAMP画面からのインスタンス・リストの更新 サブファイル/ブラウズリスト スクリプトのスケルトン スクリプト・ファンクション スクリプトが参照できるフレームワーク・オブジェクト ユーザー定義のスクリプト・ファンクション いつスクリプトを再ロードしたら変更をテストできますか? 反復確認をオフにする 高度なスクリプト スクリプトを使用したインスタンス・リストとの対話 インスタンスリストとは、通常フレームワークウィンドウの右上に表示 されるビジネスオブジェクトインスタンスのリストです。例えば、組み 込みのデモ・システムでは、赤色で囲まれた次のようなインスタンスリ ストを持つEmployeeビジネスオブジェクトを使用します。

\delta by Name 🔗 by Location 📓 by Skill 🔍 Other	Number	Name	Phone	Address 📥
Specify a full or partial employee name	A1001	BEN JONES	799 5268	144 Froc
specify a full of partial employee fiame.	A1012	PATRICK PAUL	687 1717	6 Camillo
Employee Surname	A1013	GEORGE PATTISON	750 2562	12 Augu
	A1015	BRADLEY WOODS	450 1236	59 Darle
	A1020	ADAM PETER DOUGLAS	674 5310	6 Readin
C 4	A1021	DAVID MCCULLY	762 1321	15 Baker
Clear the current list of employees Search	A1025	MARY ROBINSON	126 3598	14 Whitt 🔽
	۲.	111		

多数のスクリプトがインスタンスリストと対話する必要があります。以下のトピックでこの方法を説明します。

リスト・マネージャー ビジュアル識別子とプログラム的識別子 選択されたすべてのエントリーの処理 リスト・マネージャー

スクリプトとインスタンスリストの対話は、objListManager (リスト・マ ネージャー)というフレームワークのJavaScriptオブジェクトのプロパティ にアクセスすることによって行います。

例えば、従業員の詳細を表示する画面を表示するスクリプトでは、 SETVALUEコマンドでobjListManagerを使用して、従業員をインスタンス リストで選択されたエントリーに設定します。

/\* Check for arrival at uFindEmployee \*/
if ( !(CHECK\_CURRENT\_FORM("uFindEmployee","Unable to navigate to
form uFindEmployee")) ) return;

/\* Set the employee to be displayed to the employee selected in the \*/
/\* instance list (which is identified by the programmatic identifier AKey3) \*/
SETVALUE("utxtEmployeeCode",objListManager.AKey3[0]);

/\* Send the key required to navigate to uDisplayEmployee \*/
SENDKEY(KeyEnter);

#### 参照

objListManager ハードコードされた従業員番号を現在のインスタンスリストのエント リーに置き換える

## ビジュアル識別子とプログラム的識別子

インスタンスリストのエントリーには、必ずビジュアル識別子とプログ ラム的識別子を定義する識別プロトコルがあります。インスタンスリス トを制御するフィルターを作成するときにこれらの識別子を設定しま す。

(識別プロトコルの詳細は、『Visual LANSAフレームワーク ガイド』の 「リスト・マネージャーとインスタンス・リスト」を参照してください。)

例えば、従業員のフィルターにある次のLANSAコマンドは、インスタン スリストにエントリーを追加し、そのエントリーのプログラム的識別 子、ビジュアル識別子、および追加の列を設定します。

Invoke Method(#avListManager.AddtoList) Visualid1(#Empno) Visualid2(#FullName) Akey1(#Deptment) Akey2(#Section) Akey3(#Empno) AColumn1(#PhoneHme) AColumn2(#Address1) nColumn1(#PostCode)

この識別プロトコルの説明

- 3番目のプログラム的識別子(AKey3)には、従業員番号が入ります。
- 2番目のビジュアル識別子(VisualId2)には、従業員の名前が入ります。

識別プロトコルを理解していれば、インスタンスリストで現在選択され ている従業員の番号と名前を表示するJavaScriptを作成できます。

```
/* Get the current instance list details */
{
    var strEMPNO = objListManager.AKey3[0]; /* 3rd Akey is the number */
    var strNAME = objListManager.VisualId2[0]; /* 2nd VisualId is the name */
```

```
alert("Current employee number is " + strEMPNO);
alert("Current employee name is " + strNAME);
}
```

次のように表示されます。

Microsoft Internet Explorer 🛛 🔀					
Current employee number is A1020					
ОК					
t Internet Explorer					
Current employee name is ADAM PETE	R DOUGLAS				
ОК					
	Current employee number is A1020 OK It Internet Explorer Current employee name is ADAM PETE OK				

#### 選択されたすべてのエントリーの処理

#### インスタンスリストでは複数のエントリーを選択できます。次のスクリ プトは、選択されたすべての従業員の番号と名前をメッセージに表示し ます。

```
/* Get all the selected employees */
{
  var i = 0;
  var strMessage = "";
  for (i = 1; i <= objListManager.TotalSelected; i++)
  {
    strMessage += "Employee " + objListManager.AKey3[i];
    strMessage += " - " + objListManager.VisualId2[i] + "\x0D";
  }
  alert(strMessage);
}</pre>
```

次のように3つのインスタンスリストのエントリーが選択され、このスク リプトが使用された場合:

Number	Name	Phone	Address 🔺
A1001	BEN JONES	799 5268	144 Froc
A1012	PATRICK PAUL	687 1717	6 Camillo
A1013	GEORGE PATTISON	750 2562	12 Augu
A1015	BRADLEY WOODS	450 1236	59 Darle
A1020	ADAM PETER DOUGLAS	674 5310	6 Readin
A1021	DAVID MCCULLY	762 1321	15 Baker
A1025	MARY ROBINSON	126 3598	14 Whitt 🔽
<			>

次のアラートメッセージが表示されます。



#### スクリプト・ポップアップ・メニュー スクリプトポップアップメニューを使用して、スクリプトをフォーマッ トして編集できます。メニューを表示するには、スクリプト・エリアを 右クリックします。

X	·切り初め - Cirl+X
100	⊐E'Ctrl+C
Gh.	貼り付け - Ctr HV
40	元に戻す - Oirl+Z
ē	編り返し - Cirl+Y
ō	置换 - Cirl+H
Q	検索 - Ctrl+F
	全て選択-CuHA
	大文字
	小文字
-	選択された行をコットにする
5	選択された行のロメントを無効にする
0	選択された行をインナントする
0	課状された行のインデントを課効にする
3	自己的
4	大きいフォント
-	小さいフォント
-	行動を表示
	<b>フレームワー</b> ク
	アフリケーション
	とう/ネス オフシンカト
	1821-1
	インスタンス リスト エントリー
	セッション エントロール
	ファンジションキーのオーバーライト
	SENDKEY - ファンジションキー
	共通RAMPスクリン学校能

上部に表示されているオプション切 り取り、コピー、貼り付け、元に戻 す、やり直しは、多数のエディター でよく使用されるオプションなの で、ここでは改めて説明しません。 大文字オプションと小文字オプショ ンは、スクリプト・エディターで現 在選択されているテキストの大文字 小文字を変更します。Javascriptは大 文字小文字を区別することに注意し てください。 小文字オプションと大文字オプショ ンでは、テキスト・エディターで使 用されているフォントのサイズを変 更できます。 行数を表示オプションは、テキス ト・エディターの行番号を表示また は非表示にします。 以下のオプションは、さまざまなフ レームワーク・オブジェクトのプロ パティをスクリプトに挿入するとき に使用します。使用方法: objFrameworkのプロパティを入 • 力する場合はフレームワーク objApplicationのプロパティを入 • 力する場合はアプリケーション objBusinessObjectのプロパティ • を入力する場合はビジネス オブ ジェクト

- objCommandのプロパティを入 力する場合はコマンド
- objListManagerのプロパティを 入力する場合はインスタンス リ スト エントリー

5250サブファイルの処理オプション は、サブファイル/ブラウズリストの コードを挿入するときに使用しま す。

セッション コントロールオプション は、よく使用されるファンクション とobjUserパラメータをスクリプトに 入力するときに使用します。

例:

- ハードコードされたユーザー名を現在のフレームワーク・ユーザー に置き換える
- ハードコードされた従業員番号を現在のインスタンスリストのエン トリーに置き換える
- スクリプトのポップアップ・メニューに独自のオプションを作成する

ハードコードされたユーザー名を現在のフレームワーク・ユー ザーに置き換える

次のスクリプト行にあるハードコードされたユーザー名"QPGMR"を現在のフレームワーク・ユーザーの名前に置き換える場合:

SETVALUE("utxtUserName", "QPGMR");

"QPGMR" (引用符も含む)を選択し、右クリックしてセッションコント ロールを選択し、ユーザーネームオプションを選択します。



#### これにより、定数"QPGMR"は現在のフレームワーク・ユーザーの代替値 に置き換えられました。

SETVALUE("utxtUserName", objUser.Name);

ハードコードされた従業員番号を現在のインスタンスリストの エントリーに置き換える

トラッキング情報を使用してスクリプトを自動的に生成すると、そのス クリプトにはユーザーが入力したハードコードされたフィールド値が入 ります。このスクリプトが選択されたオブジェクトを処理するようにす るには、ハードコードされた値を該当する識別子に置き換える必要があ ります。

次のスクリプト行にあるハードコードされた従業員番号"A1234"をイン スタンスリストで現在選択されている従業員の名前に置き換えます。

SETVALUE("uEmpNo","A1234");

最初に、従業員の識別に使用されるビジュアル識別子とプログラム的識 別子を見つけます。次に、スクリプトでハードコードされた番 号"A1234" (引用符も含む)を強調表示し、右クリックしてポップアップメ ニューを表示し、インスタントリストエントリーオプションを選択し て該当する識別子を選択します。

		Alpha Key 1
		Alpha Key 2
		Alpha Key 3
		Alpha Key 4
		Alpha Key 5
		Numeric Key 1
		Numeric Key 2
		Numeric Key 3
		Numeric Key 4
🔀 切り取り – Ctrl+X		Numeric Key 5
םピー - Ctrl+C		Visual Identifier 1
🍋 貼り付け – Ctrl+V		Visual Identifier 2
う 元に戻す - Ctrl+Z		Additional Alpha Column 1
🚰 繰り返し - Ctrl+Y		Additional Alpha Column 2
> 置換 - Ctrl+H		Additional Alpha Column 3
Q 検索 - Ctrl+F		Additional Alpha Column 4
🕤 全て選択 - Ctrl+A		Additional Alpha Column 5
大文字		Additional Alpha Column 6
小文字		Additional Alpha Column 7
🖹 選択された行をコメントにする		Additional Alpha Column 8
🦻 選択された行のコメントを無効にする		Additional Alpha Column 9
👂 選択された行をインテントする		Additional Alpha Column 10
③ 選択された行のインテントを無効にする		Additional Numeric Column 1
🍯 印刷		Additional Numeric Column 2
☆ 大きいフォント		Additional Numeric Column 3
→ 小さいフォント		Additional Numeric Column 4
🎍 行数を表示		Additional Numeric Column 5
フレームワーク	•	Additional Numeric Column 6
アフツケーション	•	Additional Numeric Column 7
ビジネス オフジェクト	•	Additional Numeric Column 8
ביאלא אלא אלא אין אין אין אין אין אין אין אין אין אי	•	Additional Numeric Column 9
インスタンス リスト エントリー		Additional Numeric Column 10
セッション コントロール	•	
ファンクションキーのオーバーライト	•	
SENDKEY - ファンクションキー	×.	
共通RAMPス切プト機能	•	

これにより、定数"A1234"は従業員番号のプログラム的識別子に置き換 えられました。

SETVALUE("uEmpNo", objListManager.AKey3[0]);

# スクリプトのポップアップ・メニューに独自のオプションを作 成する

uf\_um835.xmlというxmlファイルを作成して区画の実行ディレクトリに配置することで、スクリプトのポップアップ・メニューに独自のオプションを追加できます。この処理はノートパッドを使用して行うことができます。

作成できるuf\_um835.xmlの例を次に示します。

<?xml version="1.0"?>

<EXTRACT>

<MENUITEM>

<PROPERTY NAME="CAPTION" VALUE="My user defined options" />

<SUBMENUS>

<SUBMENUITEM>

<PROPERTY NAME="CAPTION" VALUE="My caption for option 1" />

<PROPERTY NAME="STRING" VALUE="My returned text for option 1" /> </SUBMENUITEM>

<SUBMENUITEM>

<PROPERTY NAME="CAPTION" VALUE="My caption for option 2 (multiple lines returned)" />

<PROPERTY NAME="STRING" VALUE="My returned line 1 for option 2" /> <PROPERTY NAME="STRING" VALUE="My returned line 2 for option 2" /> <PROPERTY NAME="STRING" VALUE="My returned line 3 for option 2" /> </SUBMENUITEM>

<SUBMENUITEM>

<PROPERTY NAME="CAPTION" VALUE="My caption for option 3 (handling quotes in the text)" />

<PROPERTY NAME="STRING" VALUE="Quotes and greater than and less than need special handling" />

<PROPERTY NAME="STRING" VALUE="Quote - "" />

<PROPERTY NAME="STRING" VALUE="Less than - <" />

<PROPERTY NAME="STRING" VALUE="Greater than - >" />

</SUBMENUITEM>

</SUBMENUS>

</MENUITEM>

</EXTRACT>

uf\_um835.xmlというファイルを作成して上記のテキストを貼り付け、

# uf\_um835.xmlを区画の実行ディレクトリに配置すると、RAMPスクリプトを編集するときに次の新しいオプションを表示できるようになります。

切り取り - Ctrl+X コピー - Ctrl+C 貼り付け - Ctrl+V 元に戻す - Ctrl+Z 繰り返し - Ctrl+Y 置換 - Ctrl+H 検索 - Ctrl+F 全て選択 - Ctrl+A 大文字		IS IN YOUR LANSA partition execute folder. *NOTE** You must backup skeleton file changes. ons when reinstalling or upgrading RAMP-TS.	
小文字 選択された行をコメントにする 選択された行のコメントを無効にする 選択された行をインデントする 選択された行のインデントを無効にする 印刷 大きいフォント 小さいフォント 行数を表示 フレームワーク アプリレーシャン	•	how VLF buttons */ d */ e line */ state */	
<ul> <li>パン リワーンヨン</li> <li>ビ*ジ*ネス オフ*ジ*ェクト</li> <li>コマント*</li> <li>インスタンス リスト エントリー</li> <li>セッション コントロール</li> <li>ファングションキーのオーパ*ーライト*</li> <li>SENDKEY - ファングションキー</li> <li>共通RAMPスクリフ*ト機能</li> <li>My user defined options</li> </ul>		automatically prompted finish now */ My caption for option 1 My caption for option 2 (multiple lines returned) My caption for option 3 (handling quotes in the text)	

オプション1を選択すると、次のテキストがスクリプトに追加されます。

My returned text for option 1

オプション2を選択すると、次のテキストがスクリプトに追加されます。

My returned line 1 for option 2 My returned line 2 for option 2 My returned line 3 for option 2

オプション3を選択すると、次のテキストがスクリプトに追加されます。

Quotes and greater than and less than need special handling Quote - " Less than - < Greater than - >

上記のxmlでは、1番目のサブメニューに表示されるキャプションは captionプロパティから取得され、ユーザーがこのサブメニューをクリッ クした時にスクリプトに戻される値はStringプロパティから取得されるこ とを確認できます。

<SUBMENUITEM> <PROPERTY NAME="CAPTION" VALUE="My caption for option 1" /> <PROPERTY NAME="STRING" VALUE="My returned text for option 1" /> </SUBMENUITEM>

オプション2からは、ユーザーがサブメニューをクリックした時に複数行 がどのように返されるかを確認できます。

<SUBMENUITEM> <PROPERTY NAME="CAPTION" VALUE="My caption for option 2 (multiple lines returned)" /> <PROPERTY NAME="STRING" VALUE="My returned line 1 for option 2" /> <PROPERTY NAME="STRING" VALUE="My returned line 2 for option 2" /> <PROPERTY NAME="STRING" VALUE="My returned line 3 for option 2" /> </SUBMENUITEM>

オプション3からは、スクリプトに返される値に引用符(または大なり記 号や小なり記号)を組み込む場合の特殊処理を確認できます。

<SUBMENUITEM>

<PROPERTY NAME="CAPTION" VALUE="My caption for option 3 (handling quotes in the text)" />

<PROPERTY NAME="STRING" VALUE="Quotes and greater than and less than need special handling" />

<PROPERTY NAME="STRING" VALUE="Quote - "" />

<PROPERTY NAME="STRING" VALUE="Less than - <" />

<PROPERTY NAME="STRING" VALUE="Greater than - >" />

</SUBMENUITEM>

作成したxmlが有効で上記例の構造(EXTRACT、MENUITEM、 SUBMENUS、およびSUBMENUITEM)に従っている限り、そのxmlは正 しく動作します。

注:作成したバージョンのUF\_UM835.xmlがバックアップされるようにしてください。

#### RAMP画面からのインスタンス・リストの更新

チュートリアル「RAMP-TS007:データ入力ファンクションのスナップ イン」では、このトピックを詳しく説明しています。

フィルターは、その関連するインスタンスリストを管理します。RAMP 画面はビジネスオブジェクトインスタンスを削除、追加、または変更し た場合、フィルターに変更が発生したことを通知する必要があります。

コードウィザードを使用したフィルターの作成

RAMP画面からの変更を検査するフィルターを作成するには、コード ウィザードを使用してオプションインスタンスリストを更新するための シグナルを受け取るRoutineを選択します。



このオプションによってインスタンスリストへの変更を自動的に処理するフィルター・コードが作成されます。

ボタン・スクリプトへのAVSIGNALEVENTファンクションの 追加

変更を処理するボタン(通常、保存または削除)のRAMP宛先画面のボタン・スクリプトにAVSIGNALEVENTファンクションを追加して、インスタンスリストを変更する必要があることをフィルターに通知します。 例えば、オブジェクトを更新するRAMP画面で、次のステートメントをSAVEボタン・スクリプトに追加します。

AVSIGNALEVENT("Update\_List\_Entry", "BUSINESSOBJECT", objListManager.AKey1[0]);

送信されるイベントの名前はUpdate\_List\_Entryで、渡される値は更新さ

れたインスタンスの識別子です。

新しく作成されたオブジェクトの保存を処理するには、フィルターにそのオブジェクトの識別子を渡す必要があります。例えば、従業員番号を 使用して新しい従業員を追加するには、最初にGETVALUEファンクションを使用して画面上の従業員番号を取り込み、objGlobalオブジェクトの プロパティとしてその従業員番号を保存します。次に、従業員番号を フィルターに渡します。

objGlobal.utxtEmployeeCode = GETVALUE("utxtEmployeeCode"); SENDKEY(KeyEnter); AVSIGNALEVENT("Add\_List\_Entry", "BUSINESSOBJECT", objGlobal.utxtEmployeeCode);

(utxtEmployeeCodeフィールドは、宛先画面でテキストフィールドとして 定義されている従業員番号フィールドです。)

インスタンスリストの更新に使用できる標準のイベント名は次のとおり です。

- Refresh\_Instance\_List
- Update\_List\_Entry
- Add\_List\_Entry
- Delete\_List\_Entry

# インスタンスリストへの変更を自動的に処理するフィルター・ コード

Program Coding Assistantによって作成される次のRDMLXコードは、 RAMP画面が通知するイベントを自動的に処理します(このコードは参考 用です。変更する必要はありません)。

\* \_\_\_\_\_

\* Handle any external requests to update the Instance List

\* \_\_\_\_\_

EvtRoutine #Com\_owner.avEvent WithId(#EventId) WithAInfo1(#AInfo1) WithAInfo2(#AInfo2) WithAInfo3(#AInfo3) WithAInfo4(#AInfo4) WithAInfo5(#AInfo5) WithNInfo1(#NInfo1) WithNInfo2(#NInfo2) WithNInfo3(#NInfo3) WithNInfo4(#NInfo4) WithNInfo5(#NInfo5) \* put the received values into fields Change #vf\_elIdn #EventId.Value \* Map the AInfo and NInfo values passed, into the key fields - #EMPNO Change #DEPTMENT #AInfo1 Change #SECTION #AInfo2 Change #EMPNO #AInfo3

Case #vf\_elIDN

when '= Refresh\_Instance\_List' \* Reload the Instance List Invoke #Com\_Owner.uSelectData

when '= Add\_List\_Entry'
\* Add an entry to the list view
fetch FIELDS(#XG\_Ident) FROM\_FILE(PSLMST) WITH\_KEY(#EMPNO)
if\_status \*OKAY
\* Start an instance list update
Invoke Method(#avListManager.BeginListUpdate)
\* Set up the visual Identifier(s)
Change #UF\_VisID1 #EMPNO
Change #UF\_VisID2 #GIVENAME
Use BConcat (#UF\_VisID2 #SURNAME) (#UF\_VisID2)
\* Add instance details to the instance list

```
Invoke #avListManager.AddtoList Visualid1(#UF_VisID1)
Visualid2(#UF_VisID2) AKey1(#DEPTMENT) AKey2(#SECTION)
AKey3(#EMPNO) ACOLUMN1(#PHONEHME) ACOLUMN2(#ADDRESS1)
NCOLUMN1(#POSTCODE)
* Instance list updating has been completed
Invoke Method(#avListManager.EndListUpdate)
endif
```

when '= Update List Entry' \* Update an entry that already exists in the instance list fetch FIELDS(#XG\_Ident) FROM\_FILE(PSLMST) WITH\_KEY(#EMPNO) if status \*OKAY \* Start an instance list update Invoke Method(#avListManager.BeginListUpdate) \* Set up the visual Identifier(s) Change #UF VisID1 #EMPNO Change #UF\_VisID2 #GIVENAME Use BConcat (#UF\_VisID2 #SURNAME) (#UF VisID2) \* Add instance details to the instance list Invoke #avListManager.UpdateListEntryData Visualid1(#UF\_VisID1) Visualid2(#UF\_VisID2) AKey1(#DEPTMENT) AKey2(#SECTION) AKey3(#EMPNO) ACOLUMN1(#PHONEHME) ACOLUMN2(#ADDRESS1) NCOLUMN1(#POSTCODE) \* Instance list updating has been completed Invoke Method(#avListManager.EndListUpdate) endif

when '= Delete\_List\_Entry'
Invoke Method(#avListManager.BeginListUpdate)
\* Remove instance details from the instance list
Invoke #avListManager.RemoveFromList AKey1(#DEPTMENT)
AKey2(#SECTION) AKey3(#EMPNO)
Invoke Method(#avListManager.EndListUpdate)
endcase

Endroutine End\_Com サブファイル/ブラウズリスト

5250サブファイルにアクセスするスクリプトを作成しなければならない 場合があります。種々の状況で、以下の方法が役に立つことがありま す。

エントリーごとのサブファイル行 サブファイル・フィールドを識別する サブファイル・フィールドを参照する サブファイルの行を反復する サブファイルの次のページまたは前のページへの移動 サブファイルの特定の値を見つけて選択する

# エントリーごとのサブファイル行

サブファイルでエントリーごとに2つ以上の行が使用されている場合は、 この設定で適切な値を設定する必要があります。



そうでない場合、サブファイル・フィールドのSETVALUEとGETVALUE の結果が不適切になります。 サブファイル・フィールドを識別する

サブファイル・フィールドの命名とその他のフィールドの命名に違いは ありません。概念的には、サブファイルのフィールドの名前を設定する ときは、実際はサブファイル列の名前を設定していると考える必要があ ります。

この図でフォーカスは選択フィールドにあります。5250画面で列全体が どのように強調表示されているか注意してください。

Selフィールド列のフォーカス 選択した列のすべてのフィールド

		4	2	
	SEL			
		4	11	
	コード			
		4	20	
	部門名称			
	SFL_SELECT	5	5	γ
		5	13	٧
	ADM			
П	SEL_DEPTDESC	5	20	γ
	ADMINISTRATOR DE	PT		
		22	2	

	部門	
SEL	コード	部門名称
	ADM	ADMINISTRATOR DEPT
	AUD	INTERNAL AUDITING
	FLT	FLEET ADMINISTRATION
	GAC	GROUP ACCOUNTS
	INF	INFORMATION SERVICES
	LEG	LEGAL DEPARTMENT
	MIS	MANAGEMNT INFORMATIO
	MKT	MARKETING DEPARTMENT
	SD	SALES & DISTRIBUTION
	TRVL	TRAVEL DEPARTMENT

サブファイル・フィールドを参照する

フィールド名はサブファイルの列のみに解決するため、フィールド名だ けでサブファイルのフィールドを参照するのは不十分です。列と行内の フィールドを参照するには、行インデックスを指定する必要もありま す。インデックスは1から始まる整数です。

例えば、3番目の行のセレクター・フィールドの値を"X"に設定するには、以下のように記述します。

SETVALUE("SFL\_SELECT","X", 3);

「SETVALUEファンクション」を参照してください。

7番目の行の部門の説明を取得するには(このフィールドの名前は SFL\_DEPTDESCと付けられています)、以下のように記述します。

var sDeptDesc = GETVALUE("SFL\_DEPTDESC", 7);

「GETVALUEファンクション」を参照してください。
# サブファイルの行を反復する

サブファイルの行を反復するには、ロジックを停止するために最後の行 に達したことを認識する必要があります。これを行うには、以下のよう にCHECK\_FIELD\_EXISTSファンクションを使用します。

while ( CHECK\_FIELD\_EXISTS(sFindField,iInd) )
{
<your logic>
}

ここで、sFindFieldは任意の名前付きのサブファイル・フィールドです。

サブファイルの次のページまたは前のページへの移動

サブファイル・ページを次の画面または前の画面にするには、表示する 別のページがあるかどうかを通知するサブファイル・インジケータに名 前を設定する必要があります。

これは通常はプラス記号("+")ですが、異なる場合もあります。

	17 36 Y					資格	取得日
	Metrequirement	-	資格コード	資格記述	コメント	等級	(DDMMYY)
	17 59 V	=	ADMIN1	Administratn Part 1	Met requirement		98/03/25
	D	-	ADMIN2	Administratn Part 2			98/05/03
	17 68 Y		ECD	Economics Degree			98/05/05
Ir	98/02/25	1	ENG	English Degree	With distinction	D	94/06/05
	SFL_MORE 21 79 Y		INDUCT	Company Induction			96/09/15
IL	+						
1	23 2		F1= ヘルプ	F3= 終了 F4=PROMPT	F12= 取消し F14=M	ISG E21	= 変更
	F1= ヘルプ F3=終了 F4=PROMPT F12=取消		F22= 削除				

フィールドがある場合は、別のページがあると仮定することができま す。

if ( CHECK\_FIELD\_EXISTS("SFL\_MORE") )
{

SENDKEY(KeyPageDown);

ł

# サブファイルの特定の値を見つけて選択する

ケースA:選択肢が見つかったときに、該当する行にカーソルを設定してEnterを押す。

uSubfileSearch:function(sToForm, sFindValue, sFindField, sMoreRecsField)
{

```
/* Subfile indexed fields are one based */
var iInd = 1;
var bFound = false;
while( (CHECK FIELD EXISTS(sFindField, iInd)) && !(bFound) )
/* Found, set the flag to true to cause the loop to end */
if (sFindValue == GETVALUE(sFindField,iInd))
bFound = true;
else /* Increase field index */
iInd++;
/* If found, position the cursor to the field and index and send an Enter key to
cause that entry to be selected */
if (bFound)
SETCURSORTOFIELD(sFindField,iInd);
SENDKEY(KeyEnter);
Q_CHECK_CURRENT_FORM(sToForm,"Unable to navigate to " + sToForm);
/* If not found, check whether the nominated more records indicator field is
present on the screen. If it is we can page down.*/
/* Note the payload accompanying the Sendkey. It is used in the vHandleArrive
function to decide whether we have to repeat this logic.*/
else if ( CHECK_FIELD_EXISTS(sMoreRecsField) )
SENDKEY(KeyPageDown, "Next_Page");
```

}

ケースB: 選択肢が見つかったときに、該当する行にカーソルを設定し、セレクター・フィールド"SFL\_SELECT"の値を"2"に設定し、Enterを 押す。

このケースは、SETCURSORの呼び出しを除き、前のケースとほとんど 同じです。SETCURSORを以下の記述に置き換えます。

SETVALUE("SFL\_SELECT", "2", iInd);

# スクリプトのスケルトン

Destination、Junction、およびSpecialのスクリプトは、区画実行フォル ダーにある固有のスケルトン・ファイルに基づいて作成されます。ファ イルには以下のものがあります。

- vf\_fpm030\_D.dat –Destinationのスケルトン
- vf\_fpm030\_J.dat –Junctionのスケルトン
- vf\_fpm030\_S.dat Specialのスケルトン
- vf\_fpm030\_U.dat Unknownのスケルトン

これらのスケルトンは基本スクリプト・セクションにありますが、一部のデフォルト動作も含んでいます。

デフォルトの動作が特定のニーズに対応していない場合は、スケルトン を変更することができます。メモ帳など、任意のテキスト・エディター を使用してスケルトン・ファイルを編集します。

例えば、Destinationのスケルトンには以下の行があります。

GET\_FORM\_MESSAGE(22)

これは、ほとんどの5250アプリケーションが行22を使用してメッセージ を送信するためです。アプリケーションでメッセージを別の行に送信す る場合は、スケルトンを変更することをお勧めします。 スケルトンは、スクリプトの作成時のみ使用されることに注意してくだ さい。スケルトンの変更は既存のスクリプトに影響を及ぼしません。 警告:製品のアップグレードや再インストールにより、これらのスケル トン・ファイルが上書きされます。各アップグレードまたは再インス トールの後、加えた変更を再適用する必要があります。 スクリプト・ファンクション

ここでは、スクリプトで使用できる付属のRAMP JavaScriptファンクショ ンについて説明します。

これらのファンクションは大文字と小文字が区別されるので、スクリプ トを記述するときは表示されているとおりに同じ大文字と小文字を使用 してください。

ADD_STRINGファンクション	他のスクリプトで使用するため に文字列を固有の番号で定義し ます。
ADD_UNKNOWN_FORM_GUESS ファンクション	RAMPナビゲーション中に不明 なフォームが表示されたときに 送信するファンクション・キー
ALERT_MESSAGEファンクション	メッセージをアラートとして送 信します。
AVCLOSEFORMファンクション	フレームワークに現在の画面を 閉じるように指示します。
AVRESTOREAVALUEおよび AVRESTORENVALUEファンクション	フレームワークのVirtual Clipboardから英数字または数値 を復元します。
AVSAVEVALUEファンクション	フレームワークのVirtual Clipboardに英数字または数値を 保存します。
AVSIGNALEVENTファンクション	フィルターとRAMP画面にイベ ントを送信します。
AVSWITCHファンクション	別のビジネス・オブジェクトへ の切り替え、およびオプション として指定したコマンドの実行 を要求します。
CHECK_CURRENT_FORMファンク ション	RAMPが画面を表示しているか どうかを確認します。
CHECK_FIELD_EXISTSファンクショ	現在の画面にフィールドが存在

ン	するかどうかを確認します。
CLEAR_MESSAGESファンクション	現在スタックにあるすべての メッセージをクリアします。
COPYTOCLIPBOARDファンクション	ユーザーのクリップボードに文 字列をコピーします。
CURRENT_FORMファンクション	現在のRAMP画面のフォーム名 を取得します。
DROP_SPECIAL_FIELD_HANDLER ファンクション	5250の特別なフィールド・ハン ドラーの定義を削除します。
FATAL_MESSAGEファンクション	重大なメッセージを送信しま す。
FATAL_MESSAGE_TYPEファンク ション	致命的なナビゲーション・エ ラーが発生したときにフレーム ワークのシャット・ダウンを停 止します。
GET_FORM_MESSAGEファンクショ ン	指定した行番号の5250メッセー ジ・テキストを取得し、オプ ションでフレームワーク・メッ セージとしてルーティングしま す。
GETVALUEファンクション	RAMP画面のフィールドから値 を取得します。
HANDLE_PROMPTファンクション	フィールドのユーザー定義のプ ロンプタフォームを表示しま す。
HIDE_5250_BUTTONS() <b>ファンクショ</b> ン	ARRIVEスクリプトで使用し、 5250画面でファンクション・ キー・ボタンを非表示にしま す。
HIDE_CURRENT_FORMファンク ション	現在の画面を非表示にして、任 意指定のメッセージを表示しま

	す。
LOCK_FRAMEWORKファンクション	フレームワークをロックしま す。
MESSAGEファンクション	メッセージを送信します。
NAVIGATE_TO_SCREENファンク ション	画面に移動します。
OVERRIDE_KEY_CAPTION_ALL ファンクション	すべての画面でファンクショ ン・キーに新しいキャプション を割り当てます。
OVERRIDE_KEY_CAPTION_SCREEN ファンクション	特定の画面でファンクション・ キーに新しいキャプションを割 り当てます。
OVRSFLAUTOGUIファンクション	画面ごとにAxesシステム・フラ グRecognise subfiles as tablesをオ ンまたはオフにすることができ ます。
Q_CHECK_CURRENT_FORMファン クション	RAMPで画面が表示されている ことをチェックし、次の画面が 到着したときに要求を待ち行列 に入れ、処理します。
Q_NAVIGATE_TO_SCREENファンク ション	指定された5250画面に移動し、 次の画面が到着したときに要求 をキューに入れ、処理します。
Q_SENDKEYファンクション	キーを押す操作をエミュレート し、次の画面が到着したときに 要求をキューに入れ、処理しま す。
Q_SETVALUEファンクション	5250画面のフィールドのコンテ ンツを値に対して設定し、次の 画面が到着したときに要求を キューに入れ、処理します。

RESTART_LAST_NAVIGATIONファ ンクション	次の画面が到着したときに直前 のナビゲーション・プランを再 開します。
SCREENファンクション	指定された画面名の画面オブ ジェクトを返します。
SENDKEYファンクション	キーを押す操作をエミュレート します。
SET_HANDLER_CAPTIONファンク ション	現在のコマンドハンドラーの キャプションを新しい値に設定 します。
SET_SPECIAL_FIELD_HANDLER ファンクション	5250の特別なフィールド・ハン ドラーの現在の定義を設定また はリセットします。
SETBUSYファンクション	指定されたフィールド、テキス ト、またはサブファイル・セル にカーソルを移動します。
SETCURSORTOFIELDファンクショ ン	指定されたフィールド、テキス ト、またはサブファイル・セル にカーソルを移動します。
SETCURSORTOROWファンクション	画面の指定された行と列にカー ソルを移動します。
SETKEYENABLEDファンクション	Destinationのボタンまたは5250 ファンクション・キーを動的に 有効または無効にします。
SETVALUEファンクション	RAMP画面のフィールドに値を 設定します。
SHOW_5250_BUTTONS() <b>ファンク</b> ション	ARRIVEスクリプトで使用し、 5250画面でファンクション・ キー・ボタンを表示します。
SHOW_CURRENT_FORMファンク	現在の画面を表示または非表示

ション	にします。
SHOWSTATUSBARファンクション	エンド・ユーザーの5250ターミ ナル・ステータス・バーを表示 または非表示にします。
STRINGファンクション	指定された文字列ID番号の文字 列を返します。
STRIP_LEADING_NUMBERSファン クション	文字列から先頭の数値を返しま す。
TONUMBERファンクション	文字列または他のJScriptオブ ジェクトを数値にします。
TOSTRINGファンクション	数値、NULL、または他の JScriptオブジェクトを文字列に します。
TRACEファンクション	実行時情報をトレース・パネル に追加します。
TRIM_LEFTファンクション	文字列の先行(左)のスペースを 除去します。
UNLOCK_FRAMEWORKファンク ション	フレームワークのロックを解除 します。

# LOCK\_FRAMEWORK ファンクション

フレームワークを ロックし、表示するロック・メッセージを指定しま す。

これは、VLコン ポーネントで Set #avFrameworkManager uLocked(USER|PROGRAM|PROGRAM\_EXIT) uLockedMessage('message') を使用するのと同じです。

VLF-WIN環境のみ有効で、その他の環境では無視されます。

# 構文

LOCK\_FRAMEWORK(lock type, lock message)

# パラメータ

Lock type	必 須。必要なロックのタイプを"USER"、"PROGRAM"また は "PROGRAM_EXIT"で指定します。
	USER が設定されると、フレームワークはロックされます が、ユーザーの操作により ロックを解除できます。
	PROGRAM が設定されると、フレームワークはロックされ、 解除はプログラムでのみ可 能です。
	PROGRAM_EXIT が設定されると、終了時を除きフレーム ワークはロックされ、 解除はプログラムでのみ可能です。
Lock message	必 須。ユーザーがロック状態に違反する操作を行った場合に 表示されるメッセージです。

## 戻り値

なし

例

LOCK\_FRAMEWORK("PROGRAM", "You need to return to the details screen before attempting this action");

# UNLOCK\_FRAMEWORKファンクション

フレームワークのロックを解除し、現在のロック・メッセージをクリア します。

これは、VLコンポーネントで Set #avFrameworkManager uLocked(FALSE) uLockedMessage("")を使用するのと同じです。

VLF-WIN環境のみ有効で、その他の環境では無視されます。

## 構文

UNLOCK\_FRAMEWORK()

パラメータ

なし

戻り値

なし

例

UNLOCK\_FRAMEWORK();

## RESTART\_LAST\_NAVIGATIONファンクション

次の画面が到着した時に直前のナビゲーション・プランを再開します。 構文

RESTART\_LAST\_NAVIGATION()

パラメータ

## なし

戻り値

""(Null文字列)または直前に使用されたファンクション・キー

備考

このファンクションは、直前に実行されたナビゲーションを再開する、 または直前のキーを再送信するために使用します。

例えば中断メッセージ画面などの不明な画面や特別な画面のスクリプト が到着し、論理的にナビゲーションの流れからそれを取り除く必要があ る場合に使われます。

このファンクションの呼び出しの後には、SENDKEY()ファンクションが 呼び出され、今の画面の流れから現画面を取り除きます。

2つの画面をナビゲート中に中断メッセージが現れた場合は、ファンク ションの戻り値は""(Null文字列)になります。次の画面が到着する と、進行中のナビゲーション・プランが再開され、画面の流れを元に戻 します。

もしユーザーによってとられたアクション、例えばファンクション・ キーやボタンが押されている最中に中断メッセージ画面などが出た場合 は、その寸前に使用されたファンクション・キーが戻されます。これに より、ユーザーは最後のキーを最送信することができ、次の画面フロー に続けることができます。

例

var lastFkey = RESTART\_LAST\_NAVIGATION();

SENDKEY(KeyF3);

if (lastFkey != "") Q\_SENDKEY("",lastFkey); /\* queue sending of the last function key before the break message screen appeared \*/

このファンクションはスクリプトが到着した時のみに使用され、ほぼ毎

回この後にSENDKEY()ファンクションが続いて現在の画面が取り除かれることに注意してください。

# OVRSFLAUTOGUI ファンクション

サブファイルをテーブルとして認識するAxes システム・フラグを画面ご とにオンまたはオフにすることができます。

このスクリプト・ファンクションを使用するには、aXes 2.1 またはそれ 以降のバージョンが必要です。

# 構文

OVRSFLAUTOGUI(sScreenName, bOvr)

# パラメータ

sScreenName	必 須。画面の名前を指定する文字列。
bOvr	省略可能。ブール値。 trueを指定すると、サブファイルはテー ブルとして認識されます。 falseを指定すると、設定はオフになりま す。 値なしを含め、その他の値では、動作は 現在のサブファイルAutoGUIの設定に従 います。

# 備考

このAPIの呼び出しを配置できる唯一の場所は、ログイン・スクリプトのNavigate\_TO セクションです。

```
vHandle_NAVIGATETO:function(sToForm, oPreviousForm)
{
var bReturn = true;
HIDE_CURRENT_FORM();
SETBUSY(true);
OVRSFLAUTOGUI("Employee_Skills", false);
etc
```

## AVSWITCHファンクション

別のビジネス・オブジェクトへの切り替え、およびオプションとして指 定したコマンドの実行を要求します。

# 構文

AVSAVEVALUE(sTo, sNamed, sExecute, sTargetWindow)

	J	ľ	ラ	メ	—	タ
--	---	---	---	---	---	---

sТо	オブジェクトをどのコントロールに切り替える かをFRAMEWORK、APPLICATION、 BUSINESSOBJECTで指定する
sNamed	どのコントロールに切り替えるかをユーザー・ オブジェクト名/APPLICATIONまたは BUSINESSOBJECTのタイプで指定する
sExecute	どのコントロールに切り替えるかをユーザー・ オブジェクト名/ターゲットのFRAMEWORK、 APPLICATIONまたはBUSINESSOBJECTで実行す るコマンドのタイプで指定する
sTargetWindow	切り替え操作を行う対象ウィンドウを指定す る。指定できるのは、CURRENT(現在のウィン ドウ)、MAIN(メイン・ウィンドウ)、具体的な ウィンドウ名のいずれか。デフォルト値はMAIN

## 戻り値

なし

備考

 関数の呼び出し側は、AVSWITCHを呼び出すときにRAMP-TSセッションがビジー状態ではないことを確認する必要があります。開発 モードでは、これにより、RAMPセッション・ビジー・メッセージが 出力されます。

例

/\* Switch to the Combobox Fast Part examples and run Example 1 \*/ AVSWITCH("BUSINESSOBJECT","C0846821929747C295C29FF1E518CCA] /\* Switch to a business object in the same additional window and \*/ /\* run the details command.\*/

AVSWITCH("BUSINESSOBJECT","EMPLOYEES","DETAILS","CURRENT'

# TRIM\_RIGHTファンクション

文字列の後続のスペースを除去します。 構文

var sTrimmed = TRIM\_RIGHT(sString);

パラメータ

Ctring	必須。右スペースを
sstring	除去する文字列。

# 戻り値

文字列。右スペースを除去した文字列を返します。

例

var myString = GETVALUE("PageMarker"); var sTrimmed = TRIM\_RIGHT(myString);

# TRIM\_LEFTファンクション

文字列の先行(左)のスペースを除去します。 構文

var sTrimmed = TRIM\_LEFT(sString);

パラメータ

sString	必須。左スペース を除去する文字 列。
---------	---------------------------

## 戻り値

文字列。左スペースを除去した文字列を返します。

例

var myString = GETVALUE("PageMarker"); var sTrimmed = TRIM\_LEFT(myString);

# SHOW\_CURRENT\_FORMファンクション

現在の画面を表示または非表示にします。

# 構文

SHOW\_CURRENT\_FORM(bShow)

# パラメータ

bShow 必須。現在の画面を表示するかどうかを 示すブール値。

# 戻り値

なし

例

(到着スクリプトで)現在の画面を表示します。

vHandle\_ARRIVE:function(oPayload, oPreviousForm)
{

var bReturn = true;

SHOW\_CURRENT\_FORM(true);

## SHOWSTATUSBARファンクション

エンド・ユーザーの5250ターミナル・ステータス・バーを表示または非 表示にします。

# 構文

SHOWSTATUSBAR(bShow)

# パラメータ

bShow	必須。5250ターミナル・ステータス・バーを表
DSHOW	示するかどうかを示すブール値。

## 戻り値

なし

## 備考

5250ターミナル・ステータス・バーは以下のように表示されます(赤枠内)。

	前の検索/保守	
社員 No	. A1234	
社員姓	JACKSON	
社員名	STEPHEN	
住所1	• 6 Melissa Place	
住所2	• West Pennant Hills	
住所 3	NSW Australia 2125	
自宅電話番号	. (02) 9871 7773	
部門コード	• SD +	
課コード	• ES +	
入社日付 (DDMMYY)	. 96/08/14 +	
退職日付 (DDMMYY)	. 0/00/00 +	
	资格 取得日	
資格コード 資格記述 コ	コメント 等級 (DDMMYY)	
资格コード 资格記述 ADMIN1 Administratn Part 1 Me	コメント 等級 (DDMMYY) Met requirement D 98/03/25	
資格コード 資格記述 ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03	
資格コード 資格記述 コード 資格記述 ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2 ECD Economics Degree	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05	
資格コード 資格記述 コード 資格記述 コード 資格記述 ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2 ECD Economics Degree ENG English Degree With Administration Part 2	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05 With distinction D 94/06/05	
資格コード 資格記述 コード 資格記述 コード 資格記述 ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2 ECD Economics Degree ENG English Degree Wi INDUCT Company Induction	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05 With distinction D 94/06/05 P 96/09/15 +	
資格コード 資格記述 コード 資格記述 コード 資格記述 ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2 ECD Economics Degree ENG English Degree Wi INDUCT Company Induction	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05 With distinction D 94/06/05 P 96/09/15 +	
資格コード 資格記述 コード 資格記述 ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2 ECD Economics Degree ENG English Degree Wi INDUCT Company Induction	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05 With distinction D 94/06/05 P 96/09/15 +	
資格コード 資格記述 コード 資格記述 ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2 ECD Economics Degree ENG English Degree Wi INDUCT Company Induction	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05 With distinction D 94/06/05 P 96/09/15 +	
資格コード 資格記述 : ADMIN1 Administratn Part 1 Me ADMIN2 Administratn Part 2 ECD Economics Degree ENG English Degree Wi INDUCT Company Induction	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05 With distinction D 94/06/05 P 96/09/15 +	
変格コード 資格記述   ADMIN1 Administratn Part 1 Me   ADMIN2 Administratn Part 2   ECD Economics Degree   ENG English Degree Wi   INDUCT Company Induction	コメント 等級 (DDMMYY) Met requirement D 98/03/25 D 98/05/03 D 98/05/05 With distinction D 94/06/05 P 96/09/15 +	

ここで、エンド・ユーザーはページの停止や更新、ターミナルからの メッセージの表示など、さまざまなファンクションを実行できます。 例

(到着スクリプトで)エンド・ユーザーに5250ターミナル・ステータス・ バーを表示します。

vHandle\_ARRIVE: function(oPayload, oPreviousForm)

{
 var bReturn = true;
 SHOW\_CURRENT\_FORM(true);
 HIDE\_5250\_BUTTONS();
 SHOWSTATUSBAR(true);

# TOSTRINGファンクション

# 数値、NULL、または他のJScriptオブジェクトを文字列にします。 構文

myString = TOSTRING(oObject);

# パラメータ

oObject	必須。JScriptオブジェクト。通常、これは文字列に 変換する必要がある数値です。また、文字列または
	他のJScriptオブジェクトの場合もあります。

戻り値



例

数値12.5を文字列に変換します。

var myString = TOSTRING(12.5);

# TONUMBERファンクション

文字列または他のJScriptオブジェクトを数値にします。 構文

myNumber = TONUMBER(oObject);

# パラメータ

oObject	必須。JScriptオブジェクト。通常、これは数値に変 換する必要がある文字列です。また、数値の場合も
	のリより。

# 戻り値

数 必須。 値 数値。	えされる
----------------	------

例

文字列"12.5"を数値に変換します。

var myNumber = TONUMBER("12.5");

#### SETBUSYファンクション

システムのビジー状態を有効/無効にします。SETBYがtrueに設定される と、状態ライトが赤に変わり、全てのユーザー操作が無視されます。

#### 構文

SETBUSY(fState)

パラメータ

fState 必須。システムをビジー状態にする必要があるか どうかを示すブール値。

#### 戻り値

なし

#### 備考

SETBUSY(true)の設定は以下の状態であることを意味します。

- システムが何かを行っているためビジー状態です。
- システムがビジー状態の間はユーザーが行うことは無視されます。
- ユーザーはこのビジー状態を解除することはできませんし、するべきこともありません。ビジーなアクティビティが完了すると、自動的に解除されます。

ユーザーの操作が必要な場合にはこれを有効にしないでください。

このスクリプトでtrueを設定してビジー状態にする場合、処理終了後に falseに設定することを忘れないでください。これを忘れると、ユーザー がaXes画面とのやり取りができなくなってしまいます。

例

ビジー状態を無効にしてユーザーの対話処理を可能にし、システムがビ ジーではないことを示します。 SETBUSY(false);

ビジー状態を有効にしてユーザーの対話処理を無視し、システムがビ ジーであることを示します。

SETBUSY(true);

# CHECK\_FIELD\_EXISTSファンクション

現在の画面にフィールドが存在するかどうかを確認します。オプションのiIndパラメータを使用して、サブファイルのフィールドのインスタンスを参照します。

#### 構文

CHECK\_FIELD\_EXISTS(sFieldName, [iInd])

## パラメータ

sFieldName	必須。確認するフィールドの名前を含む文字列。
iInd	省略可能。整数。ゼロよりも大きい必要がありま す。サブファイル・フィールドの場合、これは フィールドの特定のインスタンスです。

## 戻り値

ブール値。次のいずれかの値を返します。

true	現在の画面でフィールドが見つかりま した。
false	現在の画面でフィールドが見つかりま せんでした。

#### 備考

サブファイル・フィールドの場合、iIndを渡さないことができます。渡 した場合、iIndはゼロよりも大きい必要があります。それ以外はfalseを返 します。

例

PageMarkerというフィールドがこの画面にあることを確認します。

if (CHECK\_FIELD\_EXISTS("PageMarker"))

ページのすべてのサブファイル・エントリーを読み取ります。

```
for ( Index = 1; CHECK_FIELD_EXISTS("ColGivename", Index); Index++ )
{
    var Givename = GETVALUE("ColGivename",Index);
    var Empno = GETVALUE("ColEmpno", Index);
    ...
}
```

#### **DROP\_SPECIAL\_FIELD\_HANDLERファンクション**

5250の特別なフィールド・ハンドラーの定義を削除します。このファン クションを使用する前に、「RAMP-TSAD03:特別なフィールド受け渡 し」チュートリアルを終えている必要があります。

ハンドラー定義が存在しない場合、要求は無視され、エラーにはなりま せん。表示されたハンドラーの定義を削除しても、再表示が必要になる までは影響ありません。

## 構文

DROP\_SPECIAL\_FIELD\_HANDLER(sName,sKey)

## パラメータ

sName	必須。この特別なフィールド・ハンドラーに関連付け られているフィールドの5250名。
sKey	必須。ハンドラーが呼び出されるファンクション・ キー。SENDKEY()の操作で使用されるキー名と同じ キー名を使用します。

戻り値

なし

使用例

例については、「RAMP-TSAD03:特別なフィールド受け渡し」チュートリアルを参照してください。

## SET\_SPECIAL\_FIELD\_HANDLERファンクション

5250の特別なフィールド・ハンドラーの現在の定義を設定またはリセットします。このファンクションを使用する前に、「RAMP-TSAD03:特別なフィールド受け渡し」チュートリアルを終えている必要があります。

# 構文

 $SET\_SPECIAL\_FIELD\_HANDLER (sName, sKey, sHandler, sInfo1, sInfo2, sInfo3) \\$ 

# パラメータ

_sName	必須。この特別なフィールド・ハンドラーに関連 付けられているフィールドの5250名。
sKey	必須。ハンドラーが呼び出されるファンクショ ン・キー。SENDKEY()の操作で使用されるキー名 と同じキー名を使用します。
sHandler	必須。VLコンポーネントの特別なフィールド・ハ ンドラーの名前。VLコンポーネントは、クラス VF_AC017オブジェクトである必要があります。
sInfo1 – sInfo3	省略可能。これが後で呼び出される場合は、ハン ドラーに渡される追加の文字列情報。

## 戻り値

#### なし

使用例

例については、「RAMP-TSAD03:特別なフィールド受け渡し」チュートリアルを参照してください。

# GET\_FORM\_MESSAGEファンクション

# 指定した行番号の5250メッセージ・テキストを取得し、オプションでフレームワーク・メッセージとしてルーティングします。

# 構文

var bMoreRecords = GET\_FORM\_MESSAGE([iRow,] [sMoreIndicator,]
[bRoute])

# パラメータ

iRow	省略可能。メッセージ行番号を指定する整数。 デフォルトは最後の行です。
sMoreIndicator	省略可能。さらにメッセージがあるかどうかを 示すためにアプリケーションで使用される文字 列。デフォルトは"+"です。
bRoute	省略可能。メッセージをフレームワークのメッ セージ・エリアにルーティングするかどうかを 指定するブール値。trueの場合、画面のメッセー ジのテキストは削除されます。デフォルトはtrue です。

# 戻り値

ブール値。次のいずれかの値を返します。

true	メッセージ行に表示されたエレメントでさらに インジケータが見つかりました。
false	メッセージ行に表示されたエレメントでさらに インジケータは見つかりませんでした。

## 備考

さらに、RAMPは取得した最後のメッセージを格納するvLastMessageと いうプロパティを現在のフォーム・オブジェクトに設定します。このプ ロパティをスクリプトで使用するには、以下のようにします。 var sLastMsg = this.vLastMessage;

このポインターの使用は現在のスクリプト内のみで有効です。

## 使用例

GET\_FORM\_MESSAGE(22); /\* Extract messages and hide the message line if (this.vLastMessage != "") ALERT\_MESSAGE("ERROR:" + this.vLastMessa GET\_FORM\_MESSAGE(22, "More");

以下のサンプルは、到着スクリプトでGET\_FORM\_MESSAGESを使用して、すべての5250メッセージをフレームワークにルーティングしています。

このGET\_FORM\_MESSAGEは、5250画面で表示されているテキストを指 定した行で取得します。

5250画面でさらにメッセージがあることが示された場合は、結果として trueを返します。その他のメッセージを読み取るためには、メッセージ を表示する必要があります。これは、メッセージを表示している行に カーソルを設定し、Page Downキーを5250画面に送信することで実現し ます。5250画面が到着したら、新しいメッセージを取得します。これは 高価な演習です。

```
/* Get the 5250 message text from the message area */
var flagMoreRecords = GET_FORM_MESSAGE(22);
/* If there are more messages */
if (flagMoreRecords == true)
{
    /* Move the cursor to the line displaying the Messages */
    SETCURSORTOROW(22);
    /* Bring up the next message */
    SENDKEY(KeyPageDown);
  }
/* <ARRIVE /> - Do not remove or alter this line */
return(bReturn);
},
```

## SCREENファンクション

指定された画面名の画面オブジェクトを返します。 構文

var oScreen = SCREEN(sName)

# パラメータ

Name 必須。画面の名前を指定する文字列。

# 戻り値

Screen	オブジェクト。画面オブジェクトへの参
	照。

## 備考

ナビゲーション中の任意の時点で特定の画面のプロパティにアクセスす るときに役立ちます。

## 使用例

「画面参照を使用する」を参照してください。

# HIDE\_5250\_BUTTONS()ファンクション

ARRIVEスクリプトで使用し、5250画面でファンクション・キー・ボタンを非表示にします。

# 構文

HIDE\_5250\_BUTTONS();

なし

例

HIDE\_5250\_BUTTONS();
## SHOW\_5250\_BUTTONS()ファンクション

ARRIVEスクリプトで使用し、5250画面でファンクション・キー・ボタンを表示します。

### 構文

SHOW\_5250\_BUTTONS();

パラメータ

なし

例

SHOW\_5250\_BUTTONS();

## COPYTOCLIPBOARDファンクション

ユーザーのクリップボードに文字列をコピーします。

### 構文

COPYTOCLIPBOARD(sString);

# パラメータ

sString	必須。ユーザーのクリップボードにコピーす るデータを含む文字列。

# 戻り値 なし

使用例

COPYTOCLIPBOARD("ABC");

```
COPYTOCLIPBOARD(MAKESUBFILEINTOSTRING("uDataGrid", true, true, "0,2,3,4,5", "1,2,3,4,6"));
```

/\* Copy to a spreadsheet \*/

var MyString = ""; var TAB\_Char = "\x09" ; var End\_Of\_Line\_Char = "\x0D\x0A" ;

```
MyString = "Line 1 Cell 1" + TAB_Char + "Line 1 Cell 2" +
End_Of_Line_Char;
MyString += "Line 2 Cell 1" + TAB_Char + "Line 2 Cell 2" +
End_Of_Line_Char;
COPYTOCLIPBOARD(MyString);
```

### 注:

このファンクションを使用すると、ユーザーは使用している実際のク リップボードにデータをコピーし、Wordドキュメントやスプレッドシー トに貼り付けることができます。

### FATAL\_MESSAGE\_TYPEファンクション

致命的なナビゲーション・エラーが発生したときにフレームワークを シャット・ダウンしたくない場合は、このファンクションを使用しま す。

### 構文

FATAL\_MESSAGE\_TYPE(sType)

### パラメータ

sType	省略可能。以下のメッセージ・タイプが入った文字 列。
	FATAL (デフォルト) – エンド・ユーザー・モード で、フレームワークはシャット・ダウンします。
	HIDE – RAMPコマンド・タブに5250画面は表示さ れず、エラーが表示されます。
	INFO – エラー・メッセージ がフレームワークの メッセージ・エリアにルーティングされます。

戻り値

なし

例

FATAL\_MESSAGE\_TYPE("HIDE");

### **SETKEYENABLEDファンクション**

Destinationのボタンまたは5250ファンクション・キーを動的に有効または無効にします。

このファンクションは、5250セッションにログオンしている間、 Destinationのファンクション・キーの有効性をオーバーライドします。 このオーバーライドは、Destination画面のすべての将来の表示に影響を 及ぼします。

# 構文

SETKEYENABLED (sDestinationName,sKeyName,bEnableVLF,bEnableNL)

# パラメータ

sDestinationName	必要。Destinationの名前を含む文字 列です。
sKeyName	必要。キーの名前を含む文字列で す。「SENDKEYファンクション のファンクション・キー名」を参 照してください。
bEnableVLF	任意。ブール値。trueに設定して ボタンを表示します。falseに設定 して非表示にします。nullに設定 して無視します。
bEnableNL	任意。ブール値。trueに設定して 5250ファンクション・キーを有効 にします。falseに設定して無効に します。nullに設定して無視しま す。

#### 戻り値

なし

### 備考

即座に有効にするには、到着スクリプトの SHOW\_CURRENT\_FORM(true)の前にSETKEYENABLEDを実行する必要 があります。

SETKEYENABLED("DisplayEmployee",KeyF5, false, false);

SHOW\_CURRENT\_FORM(true);

例

uDisplayEmployeeというDestinationはプロンプト・ボタンを表示するよう に設定されていましたが、F4 5250ファンクション・キーを無効にしま す。

これらの設定を反対にオーバーライドするには

SETKEYENABLED("uDisplayEmployee", KeyF4, false,true);

ボタンの元の設定を残すが、F4ファンクション・キーも有効にするには

SETKEYENABLED("uDisplayEmployee", KeyF4, null,true);

#### SETVALUE ファンクション

5250画面のフィールドの内容を値に設定します。フィールドは、名前または画面上での順序によって識別されます。

構文

名前による設定 - SETVALUE(sVariable, sValue, iIndex)

パラメータ

名前による設定:

sVariable	必 須。RAMPフィールド名を含む文字列。
sValue	必 須。フィールドに設定する値が入った文 字列。
iIndex	省略可能。サブファイルの一部である フィールドについて、フィールドのサブ ファイル行を指定する整数。 注:指定した行インデックスが現在のサブ ファイル・ページに存在する必要がありま す。CHECK_FIELD_EXISTSを使用して、 特定の行がサブファイルに存在するかどう かをチェックできます。

#### 戻り値

なし

備考

画面上のフィール ドの値を名前で設定するには、フィールドに名前を指 定する必要があります。

順序によるフィールドの識別は、名前を使用する場合よりもフォーム・ レイアウトの変更の影響を受ける可能性が高くなります。

順序によるフィールドの初期設定は、名前による設定よりも多くのリ ソースを消費しますが、画面フィールドの順序の詳細は、後続のアクセ スが速くなるようにキャッシュされます。キャッシュのロジックでは、 特定画面上のフィールドの相対順序がサイン・オンした5250画面上で変 更されないことを前提としています。

例

SETVALUE("utxtSignOn", objUser.Name);

SETVALUE("utxtPassword",objUser.Password);

SETVALUE("utxtSelectionOrCommand","90");

SETVALUE("utxtTransaction","MOV");

### Q\_SETVALUEファンクション

5250画面のフィールドのコンテンツを値に対して設定し、次の画面が到着したときに要求を待ち行列に入れ、処理します。フィールドは、名前または画面上での順序によって識別されます。

#### 構文

function Q\_SETVALUE(argCondition, sVariable, sValue, iIndex)

## パラメータ

argCondition	省略可能。以下の値として渡すことができます。
	<ul> <li>NULLまたは""。条件が適用されないことを示しています。</li> </ul>
	<ul> <li>"=&lt;&lt;待ち行列に入れられていた命令が実行されたときの画面名&gt;&gt;"。次の画面が指定されているとおりの画面である場合は、結果のSENDKEY/SETVALUEのみを実行する必要があることを示しています。</li> </ul>
	<ul> <li>"!=&lt;&lt;待ち行列に入れられていた命令が実行されたときの画面名&gt;&gt;"。次の画面が指定されていない画面である場合は、結果のSENDKEY/SETVALUEのみを実行する必要があることを示しています。</li> </ul>
	次の画面または結果の画面が到着したときに、画面 名の条件がテストされます。指定された名前は、現 在の画面名に影響を与えません。
sVariable	必須。RAMPフィールド名を含む文字列。
sValue	必須。フィールドに設定する値が入った文字列。
iIndex	省略可能。サブファイルの一部であるフィールドに ついて、フィールドのサブファイル行を指定する整 数。
	注:指定した行インデックスが現在のサブファイ ル・ページに存在する必要がありま
	す。 CHECK_FIELD_EXISTSを使用して、特定の行 がサブファイルに存在するかどうかをチェックでき

## GETVALUEファンクション

RAMP画面のフィールドから値を取得します。

### 構文

GETVALUE(sVariable)

## パラメータ

sVariable	必須。フィールド名が入った文字列。
sIndex	省略可能。サブファイルで列として使用されている フィールドの行を指定します。
sIndex	省略可能。フィールドのサブファイル行を指定する 整数。 注:指定した行インデックスが現在のサブファイル・ ページに存在する必要があります。

### 戻り値

文字列。フィールドの値を文字列として返します。

例

MyString = GETVALUE("utxtSignOn") ;

### **SENDKEYファンクション**

キーを押す操作をエミュレートします。

#### 構文

SENDKEY(sKeyName, oPayload)

### パラメータ

SKeyName	必要。キーの名前を含む文字列。「SENDKEYファ ンクションのファンクション・キー名」を参照して ください。
oPayload	省略可能。ファンクションで渡されるオブジェク ト。

#### 戻り値

なし

#### 備考

通常、このファンクションは非同期の5250サーバー側操作を開始しま す。RAMP-TSスクリプトは、このファンクションを呼び出した直後にす べての処理を終了し、非同期の操作が完了するまで何も実行するべきで はありません。

非同期操作の完了は、結果の5250画面表示の到着スクリプトの実行によ リ示されます(待ち行列に入れられるスクリプト・ファンクションは、こ のスクリプト・ファンクションの実行の前に待ち行列に入れられる必要 があります)。

#### 使用例

SENDKEY(KeyEnter);

以下のサンプルは、SENDKEYおよびQ\_SENDKEYファンクションでペ イロード・パラメータを使用する方法を示しています。 オブジェクトが作成され、EnterキーBUTTONCLICKイベントで値がロー ドされます。次に、オブジェクトはSENDKEYファンクションの oPayloadパラメータとして渡されます。

1	/~			
,	/*	=======================================	BUTTONCLICK	=======================================

```
vHandle_BUTTONCLICK: function(sButton)
{
```

```
var bReturn = true;
```

```
if (HANDLE_PROMPT()) return(bReturn); /* If the focus element is automa
```

/\* <BUTTONCLICK /> - Do not remove or alter this line \*/

```
/* Handle function keys and buttons */
```

```
switch (sButton)
{
 case KeyEnter:
  var objEmp = new Object();
    objEmp.strEmpno = GETVALUE("empno");
    objEmp.strGName = GETVALUE("givename");
    objEmp.strSName = GETVALUE("surname");
    SENDKEY(KeyEnter, objEmp);
    break;
 case KeyF3:
    SENDKEY(KeyF3);
    break;
 case KeyF4:
    SENDKEY(KeyF4);
    break;
 case KeyF12:
    SENDKEY(KeyF12);
    break;
 case KeyF14:
    SENDKEY(KeyF14);
    break;
 case KeyF21:
    SENDKEY(KeyF21);
    break:
```

```
case KeyF22:
		SENDKEY(KeyF22);
		break;
		default:
		SENDKEY(sButton);
		break;
	}
return(bReturn);
},
```

次に、結果の画面のvHandle\_Arriveファンクションはペイロードが渡された場合にペイロードから値を取得します。

```
/* _____
/* ============ vHandle ARRIVE ================*/
/* _____
/* Handle arrival at this Destination */
/* oPayload: The payload supplied by the event initiator */
/* oPreviousForm: Reference to previous object Form*/
vHandle_ARRIVE: function(oPayload, oPreviousForm)
 var bReturn = true;
 SHOW_CURRENT_FORM(true); /* Show the form in the framework and sh
                       /* Hide any 5250 style buttons displayed
 HIDE 5250 BUTTONS();
 GET_FORM_MESSAGE(22); /* Extract messages and hide the message lin
                 /* Last thing done - turn off the busy state
 SETBUSY(false);
                                                            */
/* if there is something in the payload */
if (oPayload != null)
 {
  ALERT_MESSAGE("Employee Details from the payload are: Employee N
 }
 /* <ARRIVE /> - Do not remove or alter this line */
```

```
return(bReturn);
},
```

### Q\_SENDKEYファンクション

キーを押す操作をエミュレートし、次の画面が到着したときに要求を待 ち行列に入れ、処理します。

## 構文

Q\_SENDKEY(argCondition,sKeyName, oPayload)

# パラメータ

argCondition	省略可能。以下の値として渡すことができます。
	<ul> <li>NULLまたは""。条件が適用されないことを示しています。</li> </ul>
	<ul> <li>"=&lt;&lt;待ち行列に入れられていた命令が実行されたときの画面名&gt;&gt;"。次の画面が指定されているとおりの画面である場合は、結果のSENDKEY/SETVALUEのみを実行する必要があることを示しています。</li> </ul>
	<ul> <li>"!=&lt;&lt;待ち行列に入れられていた命令が実行されたときの画面名&gt;&gt;"。次の画面が指定されていない画面である場合は、結果のSENDKEY/SETVALUEのみを実行する必要があることを示しています。</li> </ul>
	次の画面または結果の画面が到着したときに、画面 名の条件がテストされます。指定された名前は、現 在の画面名に影響を与えません。
sKeyToSend	必要。キーの名前を含む文字列。「SENDKEYファ ンクションのファンクション・キー名」を参照して ください。
oPayload	省略可能。ファンクションで渡されるオブジェク ト。

例

Q\_SENDKEY("",KeyF12); /\* Unconditionally send F12 aginst the resulting s

SENDKEY(KeyEnter); /\* Send enter against the current screen \*/

#### または

Q\_SENDKEY("=Confirm",KeyEnter); /\* If the resulting screen is named Con Q\_SENDKEY("!=Confirm",KeyF12); /\* else send F12 SENDKEY(KeyEnter); /\* Send enter against the current screen \*/

「SENDKEYファンクション」も参照してください。

### **CHECK\_CURRENT\_FORMファンクション**

RAMPが画面を表示しているかどうかを確認します。

### 構文

CHECK\_CURRENT\_FORM(sFormName [, sMessageText1] [, sMessageText2] ...)

パラメータ

sFormName	必須。フォーム名を指定する文字列。
sMessageText1	省略可能。送信する最初のメッセージ文字列。
sMessageText2	任意。最初のメッセージ文字列と連結する文字列 (各文字列の間には区切りスペースが自動的に挿入 されます)。

#### 戻り値

ブール値。次のいずれかの値を返します。

true	現在表示されているフォームに指定されたフォーム名が あります。
false	現在表示されているフォームに指定されたフォーム名が ありません。

### 備考

スクリプトまたはユーザーが特定の画面に進んだかどうか、または前の 画面で中止したかどうかを確認するために使用します。

CHECK\_CURRENT\_FORMは、falseを返した場合、現在のRAMP画面を 自動的に非表示にして所定のメッセージを表示します。

予定の画面が到着したこと、および到着していない場合に現在の画面が まだ表示されていることをスクリプトでテストしたい場合は、 CHECK\_CURRENT\_FORMファンクションを使用せずに、代わりに以下 を使用する必要があります。

if (CURRENT\_FORM() == "My\_Form");

画面上の検証エラーを処理するスクリプトを記述する場合は、検証エ

ラーが発生しても現在の画面を表示するのが一般的であり、ユーザーは 想定される次の画面に進みません。 このため、この状況では CHECK\_CURRENT\_FORMを使用するべきではありません。

## 例

if ( !(CHECK\_CURRENT\_FORM("uItemMasterBrowse","Unable to navigate
to form uItemMasterBrowse")) ) return;

## Q\_CHECK\_CURRENT\_FORMファンクション

RAMPで画面が表示されていることをチェックし、次の画面が到着した ときに要求を待ち行列に入れ、処理します。

### 構文

Q\_CHECK\_CURRENT\_FORM(sFormName [, sMessageText1] [, sMessageText2] ...)

# パラメータ

sFormName	必須。フォーム名を指定する文字列。
sMessageText1	省略可能。送信する最初のメッセージ文字列。
sMessageText2	省略可能。最初のメッセージ文字列と連結する文 字列(各文字列の間には区切りスペースが自動的に 挿入されます)。

### 戻り値

ブール値。次のいずれかの値を返します。

true	現在表示されているフォームに指定されたフォーム名が あります。
false	現在表示されているフォームに指定されたフォーム名が ありません。

## 備考

スクリプトまたはユーザーが特定の画面に進んだかどうか、または前の 画面で中止したかどうかを確認するために使用します。

例

#### AVCLOSEFORMファンクション

フレームワークに現在のフォームを閉じるように指示します。

#### 構文

AVCLOSEFORM()

#### パラメータ

### なし

#### 戻り値

なし

### 備考

RAMP画面が個別のフォームとして実行されていて、完了後に自動的に 閉じる必要がある場合にAVCLOSEFORMを使用します。

AVCLOSEFORMの発行時点で現在のフォームをVisual LANSAフレーム ワークが認識でき、このJunction画面またはDestination画面に有効な vHandle\_NAVIGATETOファンクションがあることを確認してください。 これにより、フレームワークはサイン・オフしてセッションを正常に終 了するために戻ることができます。

この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が戻 されるまで、その呼び出しの結果は表示されないか、使用できない可能 性があります。

#### 例

/\* Close this command handler, since the Delete is now done \*/

/\* We should ensure we are on a Junction or Destination at this point \*/ /\* so that the Framework can cleanly navigate the session to sign off \*/ AVCLOSEFORM();

#### HIDE\_CURRENT\_FORMファンクション

現在のフォームを非表示にして、任意指定のメッセージを表示します。 このファンクションを使用して、現在の5250画面をユーザーに非表示に して、ユーザーがその画面を手動で操作できないようにします。 例えば、スクリプトが5250サブファイル検索を実行し、予期した製品番

号が見つからなかった場合に、この処理を実行します。

HIDE\_CURRENT\_FORM("Product number", strProductNumber, "could not be found. You may not be authorized to view it.");

このスクリプトは、ユーザーにエラーメッセージを表示し、現在の5250 フォームを非表示にします。 これにより、そのフォームは、他のスクリ プト制御アクションだけが操作できるようになります。

#### 構文

HIDE\_CURRENT\_FORM([sMessageText1] [, sMessageText2] ...[, sMessageTextN])

パラメータ

sMessageText1	省略可能。送信する最初のメッセージ文字 列。
sMessageText2 -> N	省略可能。前のメッセージ・テキストと連結 する文字列(区切りスペースが挿入されま す)。

#### 戻り値

なし

注:

この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が戻 されるまで、その呼び出しの結果は表示されないか、使用できない可能 性があります。

例

HIDE\_CURRENT\_FORM("Inventory item ", objListManager.AKey1[0] , " was deleted.");

### CURRENT\_FORMファンクション

現在のRAMP画面のフォーム名を取得します。

構文

CURRENT\_FORM()

なし

戻り値

文字列。現在の画面のフォーム名を文字列として返します。

例

MyString = CURRENT\_FORM() ;

#### SETCURSORTOROWファンクション

画面の指定された行と列にカーソルを移動します。

このファンクションは、現在表示されているwebページの画面操作では なく、それを支える5250画面のカーソルの動き(5250の行と列)をプログ ラム上で操作するためのものです。通常このファンクションを呼び出し た後には、直後にSENDKEYファンクションの呼び出しが続きます。

#### 構文

SETCURSORTOROW(iRow, iColumn)

#### パラメータ

iRow	必須。画面の行を指す整数。最上部の行1から始 まります。
iColumn	省略可能。画面の列を指す整数。ゼロより大き い必要があります。デフォルトは1です。

戻り値

なし

備考

カーソルは画面上の任意の位置に設定できます。

例

カーソルを上から9番目の行、左から43番目の列に設定します。

SETCURSORTOROW(9, 43);

#### SETCURSORTOFIELDファンクション

指定されたフィールド、テキストの一部、またはサブファイル・セルに カーソルを移動します。

このファンクションは、現在表示されているwebページの画面操作では なく、それを支える5250画面のカーソルの動き(5250の行と列)をプログ ラム上で操作するためのものです。通常このファンクションを呼び出し た後には、直後にSENDKEYファンクションの呼び出しが続きます。

#### 構文

SETCURSORTOFIELD(sFieldName, iInd)

#### パラメータ

sFieldName	必須。カーソルの設定先のフィールドの名前を含 む文字列。これは、名前付きのフィールド、テキ スト、またはサブファイル列です。
iInd	サブファイルに必要。整数。ゼロよりも大きい必 要があります。サブファイル・フィールドの場 合、これはフィールドの特定のインスタンスです (列内の行)。

#### 戻り値

なし

#### 備考

カーソルを画面の名前付きのフィールドやテキストまたはサブファイル 列に設定できます。

#### 例

RAMP-TSで、givenameというフィールドにカーソルを設定します。

SETCURSORTOFIELD("givename");

# サブファイルの上から5行目のColDepartmentというフィールドにカーソ ルを設定します。

SETCURSORTOFIELD("ColDeptment", 5);

### ALERT\_MESSAGEファンクション

メッセージをアラートとして送信します。

### 構文

ALERT\_MESSAGE(sMessageText1 [, sMessageText2] ...)

# パラメータ

sMessageText1	必須。送信する最初のメッセージ文字列。
sMessageText2	省略可能。最初のメッセージ文字列と連結する文 字列(各文字列の間には区切りスペースが自動的に 挿入されます)。

#### 戻り値

なし

### 注:

この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が戻 されるまで、その呼び出しの結果は表示されないか、使用できない可能 性があります。

例

ALERT\_MESSAGE("Inventory item ", objListManager.AKey1[0], " was deleted.");

#### CLEAR\_MESSAGESファンクション

現在スタックにあるすべてのメッセージをクリアします。 構文

CLEAR\_MESSAGES()

パラメータ

なし

戻り値

なし

注:

この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が戻 されるまで、その呼び出しの結果は表示されないか、使用できない可能 性があります。

例

CLEAR\_MESSAGES();

#### FATAL\_MESSAGEファンクション

重大なメッセージを送信し、VLFアプリケーション全体を終了させます (ただし、デザイン・モードで実行されている場合を除きます)。 デザイン・モードでは、メッセージの詳細がRAMPパネル・エリアの中 央に表示され、アプリケーションは実行し続けます。実行モードでは、 VLFアプリケーション全体が終了します。

#### 構文

FATAL\_MESSAGE(sMessageText1 [, sMessageText2] [, sMessageText3]...)

#### パラメータ

sMessageText1	省略可能。送信する最初のメッセージ文字列。
sMessageText2	省略可能。最初のメッセージ文字列と連結する文字 列(各文字列の間には区切りスペースが自動的に挿入 されます)。

#### 戻り値

なし

#### 注:

この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が戻 されるまで、その呼び出しの結果は表示されないか、使用できない可能 性があります。

#### 例

FATAL\_MESSAGE("Inventory item ", objListManager.AKey1[0], " was deleted.");

### MESSAGEファンクション

メッセージを送信します。

#### 構文

MESSAGE(sMessageText1 [, sMessageText2] ...)

パラメータ

sMessageText1	必須。送信する最初のメッセージ文字列。
sMessageText2	省略可能。最初のメッセージ文字列と連結する文 字列(各文字列の間には区切りスペースが自動的に 挿入されます)。

#### 戻り値

なし

注:

この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が戻 されるまで、その呼び出しの結果は表示されないか、使用できない可能 性があります。

このメッセージ文字列が結合された場合の最大長は130文字を超えること はできません。

例

MESSAGE("Inventory item ", objListManager.AKey1[0], " was deleted.");

### AVSIGNALEVENTファンクション

# フレームワーク・フィルターとRAMP画面にイベントを送信します。 構文

AVSIGNALEVENT([sId] [,sTo] [,sAInfo1] [,sAInfo2] [,sAInfo3] [,sAInfo4] [,sAInfo5] [,nNInfo1] [,nNInfo2] [,nAInfo3] [,nNInfo4] [,nNInfo5])

# パラメータ

SId	必須。イベントの識別子が入った文字列。
STo	有効な値: FRAMEWORK = この通知はフレームワーク全体 にブロードキャストされます。 BUSINESSOBJECT = この通知は、現在のビジネ スオブジェクトのフィルターとRAMP画面にだ けブロードキャストされます。
sAInfo1	省略可能。シグナルを検査しているオブジェク トが使用できる追加情報が入った文字列。
sAInfo2	省略可能。シグナルを検査しているオブジェク トが使用できる追加情報が入った文字列。
sAInfo3	省略可能。シグナルを検査しているオブジェク トが使用できる追加情報が入った文字列。
sAInfo4	省略可能。シグナルを検査しているオブジェク トが使用できる追加情報が入った文字列。
sAInfo5	省略可能。シグナルを検査しているオブジェク トが使用できる追加情報が入った文字列。
nNInfo1	省略可能。検査しているオブジェクトが使用で きる追加情報が入った数値。
nNInfo2	省略可能。検査しているオブジェクトが使用で きる追加情報が入った数値。
nNInfo3	省略可能。検査しているオブジェクトが使用で きる追加情報が入った数値。

nNInfo4	省略可能。検査しているオブジェクトが使用で きる追加情報が入った数値。
nNInfo5	省略可能。検査しているオブジェクトが使用で きる追加情報が入った数値。

戻り値

なし

例

この例では、インスタンスリストのエントリーが削除されたことを通知 します。

AVSIGNALEVENT("Delete\_List\_Entry", "BUSINESSOBJECT", objListManager.AKey1[0] );

「RAMP画面からのインスタンスリストの更新」を参照してください。

### TRACEファンクション

ユーザーがスクリプトからの実行時情報をアプリケーションのトレー ス・パネルに追加できるようにします。

### 構文

TRACE(sTraceText1 [, sTraceText2] ...)

### パラメータ

sTraceText1	必須。表示するトレース情報が入った文字 列。
sTraceText2	省略可能。前のトレース・テキストと連結する 文字列(区切りスペースが挿入されます)。

#### 戻り値

#### なし

#### 注:

この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が戻 されるまで、その呼び出しの結果は表示されないか、使用できない可能 性があります。

#### 例

TRACE("Inventory item ", objListManager.AKey1[0], " was deleted.");

#### HANDLE\_PROMPTファンクション

関連するプロンプタフォーム(VLハンドラー)をフィールドの横に表示します。フィールドとプロンプタフォームは、「高度なプロンプト」で説明しているように特別なフィールド受け渡しエリアで指定します。 必要に応じて、追加情報を渡したり、プロンプタフォームから追加情報を取得したりできます。

#### 構文

HANDLE\_PROMPT(sArgument1 [, sArgument2] [, sArgument3]...)

## パラメータ

SArgumentn	省略可能。ユーザー定義のプロンプタが要求する可
	能性がある値が入った文字列。デフォルトでは、
	ユーザー定義のプロンプタは、5250画面のすべての
	名前付きフィールドに双方向でアクセスできます。

#### 戻り値

なし

例

if ( HANDLE\_PROMPT() ) return;

#### プロンプタフォームでのsArgument1、sArgument2などで渡さ れる値へのアクセス

RAMPスクリプトにある次のようなファンクションは、

HANDLE\_PROMPT("HELLO","THERE",123);

#### プロンプタフォームで次のようにアクセスされます。

Invoke Method(#Com\_Owner.uGet5250Field) Name(UARG1) Value(#Arg1Value) ... returns "HELLO" in #Arg1Value. Invoke Method(#Com\_Owner.uGet5250Field) Name(UARG2) Value(#Arg2Value) ... returna "THERE" in #Arg2Value. Invoke Method(#Com\_Owner.uGet5250Field) Name(UARG3) Value(#Arg3Value) ... returns "123" as a string in #Arg3Value. 渡すことができる引数の数に制限はありません。

数値を渡すことができますが、数値はVLコンポーネントでは文字列で示 されるので、再度数値に変換する必要があります。

渡されない引数を参照しても問題はありません。以下のコードがありま す。

#Arg15Value := "TEST" Invoke Method(#Com\_Owner.uGet5250Field) Name(UARG15) Value(#Arg15Value)

このコードを実行すると#ARG15Valueは"TEST"のまま残りますが、以下 のコードを実行することによって値が渡されたかどうかを実際に確認で きます。

Invoke Method(#Com\_Owner.uGet5250Field) Name(UARG15) Value(#Arg15Value) Found(#Found)

If (#Found = TRUE) /\* 15<sup>th</sup> argument was passed to HANDLE\_PROMPT \*/ Else /\* 15<sup>th</sup> argument was not passed)

プロンプタフォームが呼び出された画面上にない値にアクセスする必要 がある場合は、HANDLE\_PROMPTファンクションに追加の引数を使用 できます。

例えば、最初の画面で顧客情報が入力され、次の画面で請求書番号の入 力が指示される場合にこれを利用できます。この請求書番号が最初の画 面で入力された顧客情報に依存し、次の画面でその顧客情報にアクセス できない場合は、必要な顧客情報をobjGlobal変数に格納して、プロンプ タ・フォーム・コードで適切な選択基準に対してHANDLE\_PROMPT()パ ラメータとしてこの変数を渡すことができます。

### NAVIGATE\_TO\_SCREENファンクション

指定された5250画面に移動します。

現在のDestination画面の名前を指定した場合は、要求は無視されます。 構文

NAVIGATE\_TO\_SCREEN(sScreenName, ForAction)

# パラメータ

SScreenName	必須。画面の名前を含む文字列。
ForAction	Char 256 -省略可能。 ユーザー定義の値を含む文字列。この値は、 vHandleArriveイベントの画面ラッパーに返され ます。

#### 戻り値

なし

#### 備考

通常、このファンクションは非同期の5250サーバー側操作を開始しま す。RAMP-TSスクリプトは、このファンクションを呼び出した直後にす べての処理を終了し、非同期の操作が完了するまで何も実行するべきで はありません。

非同期操作の完了は、結果の5250画面表示の到着スクリプトの実行によ リ示されます(待ち行列に入れられるスクリプト・ファンクションは、 このスクリプト・ファンクションの実行の前に待ち行列に入れられる必 要があります)。

例

NAVIGATE\_TO\_SCREEN("Enrol Employee");
## **Q\_NAVIGATE\_TO\_SCREENファンクション**

指定された5250画面に移動し、次の画面が到着したときに要求を待ち行 列に入れ、処理します。

# 構文

Q\_NAVIGATE\_TO\_SCREEN(sScreenName)

# パラメータ

SScreenName 必須。画面の名前を含む文字列です。	
-------------------------------	--

戻り値

なし

例

Q\_NAVIGATE\_TO\_SCREEN("Enrol Employee");

このファンクションは、ボタン・スクリプトのみから呼び出す必要があ ります。

### STRIP\_LEADING\_NUMBERSファンクション

文字列の先頭にある数値を呼び出し元に返します。 構文

STRIP\_LEADING\_NUMBERS(sSourceString)

# パラメータ

sSourceString	文字列。	必須。	数値が取り出される文
	字列。		

#### 戻り値

文字列。取り出された数値。

例

このコードが実行されると、メッセージ"String returned was 15"が表示されます。

var strResult = STRIP\_LEADING\_NUMBERS("015. Office Tasks"); alert("String returned was " + strResult);

### ADD\_STRINGファンクション

固有の数字から、他のスクリプトで使用するための文字列を定義しま す。このファンクションは特に複数言語対応アプリケーションで役に立 ちます。

## 構文

ADD\_STRING(iStringNumber , sText)

# パラメータ

iStringNumber	文字列に割り当てる数字
sText	文字列のテキスト

### 戻り値

なし

#### 例

「STRINGファンクション」の定義を参照してください。

### STRINGファンクション

指定された文字列の識別番号の文字列を返します。このファンクション は特に複数言語対応アプリケーションで役に立ちます。

### 構文

STRING(iStringNumber)

### パラメータ

iStringNumber 文字列の識別番号

#### 戻り値

指定された識別番号を使用してADD\_STRINGにより以前定義された文字 列、またはテキスト"String number n not found."が含まれる文字列。

#### 例

サイン・オン・ファンクションでADD\_STRING()ファンクションを使用 し、異なる言語コードに基づいて次のような複数言語対応文字列が定義 されている場合:

ADD\_STRING(1,"OK"); ADD\_STRING(2,"Cancel"); ADD\_STRING(3,"Customer not found");

複数言語対応文字列へのアクセスが必要な他のすべてのスクリプトは、 コード内で言語独自の方法によりファンクションSTRING(n)を参照しま す。コードの例: for (i = 0; i <= 4; i++)

```
{
alert( STRING(i) );
```

#### 次の文字列が表示されます。

String number 0 not found. OK Cancel Customer not found String number 4 not found 同様に、サイン・オン・スクリプトに次のような2つの文字列が定義されている場合:

ADD\_STRING(1,"Customer number ");

ADD\_STRING(2," could not be found or you are not authorized to view them.");

#### 次のような別のスクリプトで複数言語対応メッセージを動的に作成でき ます。

var strMessage = STRING(1) + CustomerNumber.toString() + STRING(2);
alert(strMessage);

## **OVERRIDE\_KEY\_CAPTION\_SCREENファンクション**

特定画面のファンクション・キーに新しいキャプションを割り当てま す。

## 構文

OVERRIDE\_KEY\_CAPTION\_SCREEN (sDestinationName,sKeyName,sOverrideCaption)

# パラメータ

sDestinationName	必要。Destinationの名前を含む文字列です。
sKeyName	必要。キーの名前を含む文字列です。 「SENDKEYファンクションのファンクショ ン・キー名」を参照してください。
sOverrideCaption	必要。ボタンに使用される新しいキャプ ションです。

# 戻り値

なし

### 例

OVERRIDE\_KEY\_CAPTION\_SCREEN("uDisplayEmployee", KeyF1, "Aide");

### 注:

このファンクションは、サイン・オン・スクリプトでも使用できます。

### **OVERRIDE\_KEY\_CAPTION\_ALLファンクション**

任意の画面のファンクション・キーに新しいキャプションを割り当てま す。

### 構文

OVERRIDE\_KEY\_CAPTION\_ALL (sKeyName,sOverrideCaption)

# パラメータ

sKeyName	必要。キーの名前を含む文字列です。 「SENDKEYファンクションのファンクショ ン・キー名」を参照してください。
sOverrideCaption	必要。ボタンに使用される新しいキャプショ ンです。

#### 戻り値

なし

#### 例

OVERRIDE\_KEY\_CAPTION\_ALL( KeyF1, "Aide");

注:

このファンクションは、通常サイン・オン・スクリプトで使用されま す。複数言語対応アプリケーションで使用でき、すべてのファンクショ ン・キーのキャプションを別の言語に設定します。

## AVSAVEVALUEファンクション

英数字値または数値をVLF 仮想クリップボードに保存します。

構文

AVSAVEVALUE(vValue, sID1, sID2, sID3, iInstance, sLanguage, bPersist)

パラメータ

vValue	必要。仮想クリップボードに保存する英数 字または数値。 このパラメータがタイプ文字列のJavaScript 変数である場合、英数字値としてクリップ ボードに提示され、AVRESTOREAVALUE ファンクション(または同等のもの)を使用 して正確に検索されます。 タイプ数値の場合は、クリップボードにタ イプ数値として提示され、 AVRESTORENVALUEファンクション(また は同等のもの)を使用して正確に検索されま す。
sID1	必要。仮想クリップボード識別子1を含む 文字列。
sID2	任意。仮想クリップボード識別子2を含む 文字列。
sID3	任意。仮想クリップボード識別子3を含む 文字列。
iInstance	任意。インスタンス番号を含む整数。指定 されない場合のデフォルトは1です。イン スタンスは、通常、クリップボード値のリ ストの作成に使用され、現在リストにいく つのエントリが存在するのかを示す別のク リップボード値が付随します。
sLanguage	任意。言語コードを含む文字列。指定され ない場合のデフォルトはすべての言語で

	す。
bPersist	任意。保管されている値がRAMPアプリ ケーションの現在の実行を越えて存在する 必要があるかどうかを示すブール値。デ フォルトはtrueです。VLF仮想クリップ ボード値はWEBアプリケーションで持続し て存在しないため、このパラメータはVLF- WEB RAMPアプリケーションでは何の意味 もありません。

戻り値

なし

備考

- VLF仮想クリップボードに値を保存するには、AVSAVEVALUEを RAMPスクリプトで使用します。仮想クリップボードに関する詳細 は、フレームワーク・ガイドの「仮想クリップボード」に掲載されて います。
- パラメータ長の詳細については、「VF\_SAVEAVALUEと VF\_SAVENVALUE」を参照してください。
- RAMPスクリプトからのクリップボード値の送信は、非同期です。
   値を送信するとき、RAMPスクリプトが実行を完了し、フレームワークに制御が戻されるまで、その値はクリップボード上で物理的には処理されません。
- 仮想クリップボードは、主にフィルター、コマンド・ハンドラー などで実行されるRAMPスクリプトとRDML(X)コードの間で情報を受 け渡すように設計されています。
- 仮想クリップボードは、主にRAMPスクリプト間で情報を受け渡すようには設定されていません。JavaScript objGlobalオブジェクトは、 RAMPスクリプト間で排他的に情報を受け渡すより効率的な方法です。
- Webブラウザ・アプリケーションで実行されるRAMPスクリプトが 仮想クリップボードに値を送信するとき、フィルターまたはコマン ド・ハンドラーで実行されるRDML(X)コードによる以降のアクセス のために、サーバーに送信される必要があります(これは、サーバー

上で実行されるためです)。つまり、クリップボードに配置する情報のボリュームは、クライアントとサーバー間で送信される必要のある 情報の量に影響を与えるということです。

 この要求は非同期で処理されることがあります。現在アクティブな RAMP-TSスクリプトの実行が完了し、RAMP-TSマネージャに制御が 戻されるまで、その呼び出しの結果は表示されないか、使用できない 可能性があります。

### 使用例

クリップボード値を保存/リストアする、フィルターまたはコマンド・ハ ンドラーのRDMLXコードは次のとおりです。

\* Save values onto the clipboard

Invoke #avFrameworkManager.avSaveValue WithID1(Test) WithID2(EMPNO) FromAValue(("A0090") Invoke #avFrameworkManager.avSaveValue WithID1(Test) WithID2(SURNAME) FromAValue("FRED") Invoke #avFrameworkManager.avSaveValue WithID1(Test) WithID2(GIVENAME) FromAValue("BLOGGS") Invoke #avFrameworkManager.avSaveValue WithID1(Test) WithID2(POSTCODE) FromNValue(2150) Invoke #avFrameworkManager.avSaveValue WithID1(Test) WithID2(SALARY) FromNValue(123456.78)

\* Restore values from the clipboard

Invoke #avFrameworkManager.avRestoreValue WithID1(Test) WithID2(EMPNO) ToAValue(#EMPNO) UseAValueDefault("NA") Invoke #avFrameworkManager.avRestoreValue WithID1(Test) WithID2(SURNAME) ToAValue(#SURNAME) UseAValueDefault("NA") Invoke #avFrameworkManager.avRestoreValue WithID1(Test) WithID2(GIVENAME) ToAValue(#GIVENAME) UseAValueDefault("NA") Invoke #avFrameworkManager.avRestoreValue WithID1(Test) WithID2(POSTCODE) ToNValue(#PostCode) UseNValueDefault(0) Invoke #avFrameworkManager.avRestoreValue WithID1(Test) WithID2(SALARY) ToNValue(#Salary) UseNValueDefault(0) 同等の操作を実行するRAMP JavaScriptコードは、次のとおりです。

/\* Save values onto the clipboard – note POSTCODE and SALARY are numeric \*/

AVSAVEVALUE("A0090","TEST","EMPNO"); AVSAVEVALUE("FRED","TEST","SURNAME"); AVSAVEVALUE("BLOGGS","TEST","GIVENAME"); AVSAVEVALUE(2150,"TEST","POSTCODE"); AVSAVEVALUE(123456.78,"TEST","SALARY");

/\* Restore values from the clipboard \*/

```
var vEMPNO = AVRESTOREAVALUE("NA","TEST","EMPNO");
var vSURNAME = AVRESTOREAVALUE("NA","TEST","SURNAME");
var vGIVENAME = AVRESTOREAVALUE("NA","TEST","GIVENAME");
var vPOSTCODE = AVRESTORENVALUE(0,"TEST","POSTCODE");
var vSALARY = AVRESTORENVALUE(0,"TEST","SALARY");
```

### AVRESTOREAVALUEおよびAVRESTORENVALUEファンク ション

英数字値または数値をVLF仮想クリップボードからリストアします。 構文

AVRESTOREAVALUE/AVRESTORENVALUE(Default, sID1, sID2, sID3, iInstance, sLanguage)

#### パラメータ

Default	必要。値が見つからなかった場合に返すデ フォルト値を含む文字列/数値。
sID1	必要。仮想クリップボード識別子1を含む 文字列。
sID2	任意。仮想クリップボード識別子2を含む 文字列。
sID3	任意。仮想クリップボード識別子3を含む 文字列。
iInstance	任意。インスタンス番号を含む整数。指定 されない場合のデフォルトは1です。
sLanguage	任意。言語コードを含む文字列。指定され ない場合のデフォルトはすべての言語で す。

#### 戻り値

なし

#### 備考

VLF仮想クリップボードから値をリストアするには、RAMPスクリプト でAVRESTOREAVALUE/AVRESTORENVALUEを使用します。仮想ク リップボードに関する詳細は、フレームワーク・ガイドの「仮想クリッ プボード」に掲載されています。

パラメータ長に関する詳細は、フレームワーク・ガイドの 「VF\_RESTOREAVALUEとVF\_RESTORENVALUE」を参照してくださ L١。

例

```
var sSavedSurname = AVRESTOREAVALUE("Not Found", "NewEmployee",
"Surname", "", 1, FRA);
var sSavedPostcode = AVRESTOREAVALUE(9999, "NewEmployee",
"Postcode");
```

#### ADD\_UNKNOWN\_FORM\_GUESSファンクション

RAMPナビゲーションの間に表示されるフォームが不明のときに送信されるファンクション・キー。Windowsでのみ使用可能です。

#### 構文

ADD\_UNKNOWN\_FORM\_GUESS(sKeyName)

### パラメータ

SKeyName	必要。キーの名前を含む文字列。「 <u>SENDKEY</u> ファ
	ンクションのファンクション・キー名」を参照して
	ください。

#### 戻り値

なし

#### 備考

このファンクション呼び出しをセッションのサイン・オン・スクリプト で使用します。

不明な5250画面に遭遇すると、不明な5250フォームが表示されたときフレームワークをロックするプロパティがオンになっている場合、フレームワークはロック状態になります。定義済みの5250画面に移動するまで、ユーザーはフレームワーク内で動き回ることができなくなります。 ADD\_UNKNOWN\_FORM\_GUESSファンクションは、ユーザーが定義済み5250画面に移動せずに異なるフレームワーク・アクションを実行しようとするとき(例えば、異なるアプリケーション、ビジネス・オブジェクト、コマンドなどをクリックするなど)、送信するファンクション・キーを指定して、このような状況でも動き回れるようにします。ロック状態になる前に、フレームワークは追加された順に追加キーを送信していきます。

例えば、RAMPアプリケーションで、F12=キャンセルを使用して閉じる ことができる未定義のF4=プロンプト・ポップアップ・ウィンドウが表 示されているとします。このとき、RAMPに、不明な画面(たとえば、 F4=プロンプトウィンドウ)が表示されたとき、ロック・メッセージを表 示する前にまず(そのウィンドウを閉じることができるかどうか)F12キー を試行するように指示できます。

このファンクションの使用は、ファンクション・キーの送信の使用を不

明な5250画面がサポートするか、また、キーの送信後に表示される画面 でこの機能の動作を定義されているかどうかによって異なります。それ を設定するのは設計者です。

未定義の画面全体にこのファンクションを適用して使用するときは、注 意する必要があります。

使用例

ADD\_UNKNOWN\_FORM\_GUESS(KeyF3); ADD\_UNKNOWN\_FORM\_GUESS(KeyF12);

#### **SET\_HANDLER\_CAPTIONファンクション**

現在のコマンド・ハンドラー・キャプションを新しい値に設定します。 構文

SET\_HANDLER\_CAPTION(sCaption)

### パラメータ

名前による設定:

sCaption	必要。現在のコマンド・ハンドラーの新しいキャ
	プションを含む文字列。

#### 戻り値

なし

#### 備考

SET\_HANDLER\_CAPTIONを使用すると、フレームワークが表示するデフォルトのコマンド・ハンドラー・キャプションは上書きされます。 このファンクションは、任意のスクリプトでいつでも呼び出すことができます。

このファンクションは、WindowsアプリケーションやWeb RAMPアプリ ケーションで使用できます。

### 使用例

SET\_HANDLER\_CAPTION("New Command Handler Caption")

スクリプトが参照できるフレームワーク・オブジェクト

付属のRAMP JavaScriptオブジェクトは、標準情報をすべてのスクリプト でアクセスできるようにします。例えば、JavaScriptオブジェクト objUserは、プロパティ*NameとPassword*を公開します。

つまり、次のようなスクリプトで現在のユーザーの名前とパスワードに アクセスし、渡すことができます。

if (objUser.Name == "QSECOFR") alert("Your are signed on as the security
officer!");

これらの名前は大文字と小文字が区別されます。スクリプトを記述する ときは表示されているとおりに同じ大文字と小文字を使用してくださ い。

- objGlobal
- objFramework
- objApplication
- objBusinessObject
- objCommand
- objListManager
- objUser

これらのオブジェクトとそのプロパティを簡単に入力できる方法については、「スクリプト・ポップアップ・メニュー」を参照してください。

### objGlobal

objGlobalを使用して独自のプロパティを格納できます。

これは、1つのスクリプトから情報を格納し、その情報を後で別のスクリ プトで使用する必要がある場合に役に立ちます。

その情報には、後のスクリプトで参照される必要のある、画面からの フィールド値などを使用できます。

また、スクリプトが通ってきたパスを特定することもできます。した がって、同じ画面が2つのパスから使用されている場合、そのスクリプト は自身が通ってきたパスを判別できます。

プロパティ	タイプ	説明
<<任意のプロパ ティ名>>	文 字 列	割り当て先の任意のプロパティ

例

ユーザーが通ってきたパスとユーザーが作業を行っているアイテムを保存します(画面1)。

/\* Store the Item number that the user entered - this field has to be defined on this form\*/

objGlobal.utxtItemNumber = GETVALUE("utxtItemNumber");

/\* Store the action that is being performed (so that shared screens can know whether its an add or a copy) \*/

objGlobal.uLastAction = "COPY";

Remember the path the user is on, and the item the user is working with (On Screen 4).

/\* Get the action that is being performed \*/

if (objGlobal.uLastAction == "COPY")

{

ALERT\_MESSAGE("Inventory item ", objGlobal.utxtItemNumber, " was copied from ", objListManager.AKey1[0] );

}

else

ALERT\_MESSAGE("Inventory item was added.", objGlobal.utxtItemNumber

, "has been saved.");

objGlobalは5250セッション内でグローバルであることに注意してくださ い。各5250セッションには独自のobjGlobalインスタンスがあります。 詳細については、「objGlobalオブジェクトの使用」を参照してくださ い。

# objFramework

objFrameworkには、現在のフレームワークに関する情報をスクリプトに 提供する読み取り専用のプロパティが含まれます。

プロパティ	タイ プ	説明
uCaption	文字 列	現在のフレームワークのキャプ ション。
ExecutionEnvironment	文字 列	実行環境を"WIN"または"WEB"で 示す。
flagDesignMode	ブー ル値	フレームワークがデザイン・モー ドで実行されているかどうかを示 す。TrueまたはFalseを含むブール 値。
Language	文字 列	現在のLANSA言語コードを示す ("ENG"、"FRA"など)。
Partition	文字 列	現在のLANSA区画を示す ("DEM"、"SYS"など)。
TraceMode	文字 列	フレームワークがトレースモード で実行されているかどうか を"TRUE"または"FALSE"で示す。

# objApplication

objApplicationには、現在のアプリケーションに関する情報をスクリプト に提供する読み取り専用のプロパティが含まれます。

プロパティ	タ イ プ	説明
uCaption	文 字 列	現在のアプリケーションのキャプション。
uUserObjectType	文 字 列	ユーザー・オブジェクト名/現在のアプリ ケーションのタイプ。

# objBusinessObject

objBusinessObjectには、現在のビジネス・オブジェクトに関する情報をス クリプトに提供する読み取り専用のプロパティが含まれます。

プロパティ	タイプ	説明
uCaption	文 字 列	現在のビジネス・オブジェクトのキャプ ション。
uUserObjectType	文 字 列	ユーザー・オブジェクト名/現在のビジネ ス・オブジェクトのタイプ。

# objCommand

objCommandには、現在のコマンドに関する情報をスクリプトに提供する 読み取り専用のプロパティが含まれます。

プロパティ	タイプ	説明
uCaption	文 字 列	現在のコマンドのキャプション。
uUserObjectType	文 字 列	ユーザー・オブジェクト名/現在のコマン ドのタイプ。
uAlphaArg1	文 字 列	現在のVLFコマンド・ハンドラーの任意の 英数字引数1
uAlphaArg2	文 字 列	現在のVLFコマンド・ハンドラーの任意の 英数字引数2
uNumArg1	整 数	現在のVLFコマンド・ハンドラーの任意の 数字引数1
uNumArg2	整 数	現在のVLFコマンド・ハンドラーの任意の 数字引数2
uExecReason	文字列	現在のコマンド・ハンドラーが実行された 理由。この文字列には"EXECUTE"また は"ACTIVATE"が含まれ、現在のRAMPコ マンドが実行された理由を示しま す。"ACTIVATE" はVLF-WINアプリケー ション・スクリプトにのみ適用されます。 画面ラッパーを含むその他の状況で は、"EXECUTE"が常に利用されます。

### objListManager

objListManagerには、インスタンス・リストに関する情報をスクリプトに 提供する読み取り専用のプロパティが含まれます。

配列プロパティ

配列エントリー[0]は、インスタンス・リスト内の現在のエントリーの値 です(フォーカスされているエントリー)。

配列エントリー[1]、[2]、[3]…は、インスタンス・リスト内の選択された エントリーの値です。

プロパティ	タ イ プ	説明
AKey1[0] - AKey5[0]	文 字 列	インスタンス・リストの現在のインスタ ンスのキー値を示す5個の英字
NKey1[0] - NKey5[0]	文 字 列	インスタンス・リストの現在のインスタ ンスのキー値を示す5個の数値
VisualId1[0]	文 字 列	現在のインスタンス・リスト・エント リーのビジュアル識別子1
VisualId2[0]	文 字 列	現在のインスタンス・リスト・エント リーのビジュアル識別子2
AColumn1[0] – AColumn10[0]	文 字 列	現在のインスタンス・リスト・エント リーの10個の追加英字列の値
NColumn1[0] – NColumn10[0]	文 字 列	現在のインスタンス・リスト・エント リーの10個の追加数値列の値

単一値プロパティ

プロパティ タイ 説明

	プ	
TotalSelected	整数	インスタンス・リスト内の選択された エントリー数

#### リスト・マネージャー・オブジェクトの使用方法については、「スクリ プトを使用したインスタンス・リストとの対話」を参照してください。

# objUser

objUserには、現在のユーザーに関する情報をスクリプトに提供する読み 取り専用のプロパティが含まれます。

プロパ ティ	タイプ	説明
Name	文 字 列	現在のユーザーのプロファイル。
Password	文 字 列	現在のユーザーのパスワード。

### SENDKEYファンクションのファンクション・キー名

次の表は、SENDKEYファンクションで使用する必要があるファンク ション・キー名および対応する5250とWindowsのキー名を示していま す。

キー名は大文字と小文字が区別されるので、この表に示されているとお りにキー名をSENDKEYファンクションに入力する必要があります。

SENDKEY名	Windowsキー ボード	5250キー・アクション の説明	ボタン・ テキスト
KeyAttn	Esc	システム・アテンショ ン	"Attn";
KeyClear	Shift Enter	フィールドEXIT	"Clear";
KeyEnter	Enter	実行	"Enter";
KeyHelp	alt F1	ヘルプ	"Help";
KeyPageDown	Page Down	次ページ	"Page Up";
KeyPageUp	Page Up	前ページ	"Page Down";
KeyPrint	ctrl Pause	ホスト印刷	"Print";
KeyReset	ctrl	リセット	"Reset";
KeySysReq	shift Esc	sys req	"Sys Req";
KeyTestReq	alt Pause	test req	"Test Req";
KeyF1	F1	F1	"F1";
KeyF2	F2	F2	"F2";
KeyF3	F3	F3	"F3";
KeyF4	F4	F4	"F4";

KeyF5	F5	F5	"F5";
KeyF6	F6	F6	"F6";
KeyF7	F7	F7	"F7";
KeyF8	F8	F8	"F8";
KeyF9	F9	F9	"F9";
KeyF10	F10	F10	"F10";
KeyF11	F11	F11	"F11";
KeyF12	F12	F12	"F12";
KeyF13	shift F1	F13	"F13";
KeyF14	shift F2	F14	"F14";
KeyF15	shift F3	F15	"F15";
KeyF16	shift F4	F16	"F16";
KeyF17	shift F5	F17	"F17";
KeyF18	shift F6	F18	"F18";
KeyF19	shift F7	F19	"F19";
KeyF20	shift F8	F20	"F20";
KeyF21	shift F9	F21	"F21";
KeyF22	shift F10	F22	"F22";
KeyF23	shift F11	F23	"F23";
KeyF24	shift F12	F24	"F24";
KeyPA1	Esc 1	プログラム・アテン ション1	"PA1";
KeyPA2	Esc 2	プログラム・アテン ション2	"PA2";

例

SENDKEY(KeyEnter);

# ユーザー定義のスクリプト・ファンクション

「RAMP-TSAD05:SHAREDプロパティおよびファンクションを使用する」を参照してください。

### 反復確認をオフにする

RAMPスクリプトが実行されるたびに、フレームワークはそのスクリプトが反復して呼び出されたかどうかを確認し、反復して呼び出されている場合にはエラーのフラグを立てます。

ただし、1つのスクリプトが反復して呼び出される状況も考えられます。 たとえば、Special画面を2回以上連続して表示する場合などです。このよ うな場合、GLOBAL\_flagRecursionCheckプロパティを使用して反復確認 をオフにして、アプリケーションがエラーで終了することを防ぐことが できます。

このプロパティはスクリプト内で次のように使用できます。

var flagSaveCheckState = GLOBAL\_flagRecursionCheck;

GLOBAL\_flagRecursionCheck = false;

SENDKEY(KeyEnter);

GLOBAL\_flagRecursionCheck = flagSaveCheckState;

これは反復される状況のため、グローバル・プロパティを単純にTRUE またはFALSEに設定するのではなく、この例のように状態を保存して復 元することをお勧めします。Special画面が何度も表示される場合、フ レームワークは3、4レベルの反復(スクリプト・サイズと利用可能なシス テム・リソースにより異なる)を処理します。最上位レベルの反復レベル のみが、GLOBAL\_flagRecursionCheckプロパティを最終的にTRUEに戻し ます。

#### いつスクリプトを再ロードしたら変更をテストできますか?

Destination、Special、Junction (画面)スクリプト

コミット・ボタンをクリックすると、これらは実行環境に再ロードされ ます。

つまり、通常は、変更したスクリプトを再度実行するようにして(インス タンス・リストの別のオブジェクトをクリックして到着スクリプトを再 実行するなど)、スクリプトの変更を動的にテストできます。通常、画面 スクリプトに対する大部分の変更をテストするためにVLF-RAMPセッ ションを保管および再起動する必要はありません。

#### uf\_sy420\_rts.jsのSHAREDスクリプト

VLF-RAMPセッションを開くと、これらのスクリプトがロードされます。

これらのスクリプトを変更しても、別のVLF-RAMPセッションを開くまで影響はありません。

ヒント:SHAREDスクリプトをテストするときは、テストに使用してい るビジネス・オブジェクトを独立したウィンドウで開きます。

SHAREDスクリプトを変更したら、サーバーに保管し、独立したウィンドウを閉じて再起動します。これにより、独立したウィンドウのVLF-RAMPセッションが閉じて再び開かれ、変更したSHAREDスクリプトが取得されます。

高度なスクリプト 独自のナビゲーション・プランナーを作成する 画面参照を使用する vHandle\_DEPARTファンクションを使用する 到着スクリプトと画面間の通信

### 独自のナビゲーション・プランナーを作成する

RAMP Destination画面に関連付けられているコマンドをフレームワー ク・ユーザーが実行すると、現在の5250画面から必要なDestination 5250 画面に移動する方法で構成されるプランが作成されます。

通常、このプランはRAMPフレームワークにより自動的に作成されま す。特別なケースでは、独自のナビゲーション・プランニング・ロジッ クを定義することができます。

Destination画面DestinationAで、DestinationAへの最適な移動方法のプラン ニングを引き継ぐ必要があるとします。

このためには、vHandle\_USER\_NAVIGATION\_PLANというファンクションを画面DestinationAに関連付けられているスクリプトに追加します。

vHandle\_USER\_NAVIGATION\_PLAN: function()
{

var bReturn = true;

// your alternate navigation planning logic goes here

}, /\*  $\leftarrow$  Remember to separate this function from the others with a comma \*/

ユーザーが独自のプランを作成しないで、DestinationAに関連付けられて いるVLFコマンドを実行すると、RAMPフレームワークはDestinationAの vHandle\_USER\_NAVIGATION\_PLANファンクションを呼び出します。 通常、NAVIGATE\_TO\_SCREEN(「画面名」)ファンクションは、独自の ナビゲーション・プランを開始するために、この種のナビゲーション・ プランニング・ファンクションで使用されます。

この機能を使用する前に、次のようにコード化された独自のナビゲー ション・プランナーを含むDestination1という画面を考えてみましょう。

vHandle\_USER\_NAVIGATION\_PLAN: function()
{
 NAVIGATE\_TO\_SCREEN("Destination1");
 return(true);
},

このコードがまったく機能しない可能性があることを理解できない場合は、この機能を使用するべきではありません。

#### 画面参照を使用する

{

RAMP-TSの画面定義に関連付けられているスクリプトは、JavaScriptオブ ジェクトを定義します。実際には、画面スクリプトは画面と画面で実行 できる内容を定義するオブジェクトです。

ここまで、画面の定義にプロパティを追加する概念を扱ってきました。 通常、これらは以下に示すようにスクリプトの先頭で定義されます。

/\* Properties of screen Destination1 \*/ sCurrentOrder : "", fSkipIntroduction : false, fScrolling : true,

この画面定義のさまざまなファンクションはthis.sCurrentOrder、

this.sSkipIntroduction、this.fScrollingとして画面を参照します。

これらは、画面定義内の状態の維持やスクリプト内の異なるファンク ション間のやり取りに役立ちます。

SCREEN(「画面名」)ファンクションを使用することで、名前付きの画面 定義オブジェクトへの参照を取得できます。

例えば、Junction画面Junction1はナビゲーション・スクリプトにこのコードを含んでいる場合があります。

var oDest1 = SCREEN("Destination1");

oDest1.fSkipIntroduction = true; oDest1.fScrolling = false;

これにより、Junction1はDestination画面Destination1で定義されているプロパティばかりでなく、メソッドにも直接アクセスすることができます。

これは、次のようにコード化されていた可能性があります。

SCREEN("Destination1").fSkipIntroduction = true; SCREEN("Destination1").fScrolling

しかし、これは次の2つの理由により最適な解決策ではありません。

何らかの理由により、画面"Destination1"の定義が見つからない場合、SCREEN()はNULL参照を返し、スクリプトは失敗します。
文字列"Destination1"をオブジェクト参照に2回変換する必要があり、効率的ではありません。

したがって、このコードの最も適切なフォームは以下のようになりま す。

```
var oDest1 = SCREEN("Destination1");
if (oDest1 != null)
{
    oDest1.fSkipIntroduction = true;
    oDest1.fScrolling = false;
}
```

#### vHandle\_DEPARTファンクションを使用する

5250画面が到着したときに実行されるvHandle\_ARRIVEファンクション を含む画面スクリプトに習熟しているはずです。

画面スクリプトは、オプションでvHandle\_DEPARTファンクションを含 むことができます。

存在する場合、通常、このファンクションはSENDKEY操作により画面 が離れようとしているときに呼び出されます。

このファンクションを使用する必要がある場合は、以下のように画面の スクリプトに追加します。

vHandle\_DEPART: function()

{

var bReturn = true;

// your departure logic goes here

return(bReturn); /\* ← Remember to return a Boolean success/fail value \*/ }, /\* ← Remember to separate this function from the others with a comma \*/ vHandle\_DEPARTファンクションの使用については、以下の点を理解し ておく必要があります。

- これらのファンクションの使用は特異です。これらのファンクションを何回も使用している場合は、不適切な方法で問題を解決しようとしている可能性があります。
- 通常、これらのファンクションは、SENDKEY操作の結果として、 現在の5250画面がサーバーに送信されたときに呼び出されます。つま り、他のいずれかのスクリプト(例:SENDKEYを送信したスクリプ ト)がアクティブなときにこれらのファンクションがアクティブにな ることがあります。
- 出発スクリプトでSENDKEY操作を使用すると、失敗するか、不適切な結果が生じることがあります。このような操作を行うべきではありません。
- これらのファンクションは画面の送信をキャンセルすることができません。これらのファンクションには、画面が送信されていることが通知され、関連する閉じるアクションやクリーンアップ・アクションが許可されます。falseが返された場合、vHandle\_DEPARTスクリプトが失敗したことを示しています。ただし、画面送信をキャンセルする必要があることは示されていません。
- これらのファンクションは画面が送信されるたびに呼び出されま

す。

## 到着スクリプトと画面間の通信

すべての画面は到着ファンクションを含んでいます。 通常、画面の到着スクリプトは単一のデフォルトの動作を定義します。 代表的な単一のデフォルトの動作は以下のとおりです。

- Junction画面 何も実行しません。
- Special画面 画面を非表示にするキーを送信します。
- Destination画面 基盤の5250画面を表示します。

ただし、Junction、Special、またはDestination画面を変更して、複数の異なる動作をサポートしなければならない場合もあります。

これを実行する最も構造化された方法は、最初に到着スクリプトでサ ポートする動作を決定することです。次に、以下のサンプルのように画 面スクリプトの開始時にこれらの動作を明確に定義し、ドキュメント化 します。

RequestedArrivalBehaviour : 0,

ArrivalBehaviours :

{

Default : 0, /\* Default behaviour \*/ SearchNext : 1, /\* Handle scroll up request \*/ SearchLast : 2, /\* Handle scroll down request \*/ ForcedNavigation : 3, /\* Handle a forced navigation \*/ AutoConfirmation : 4 /\* Handle auto confirnmation \*/ },

これは、この画面が5つの異なる到着動作をサポートできることを非常に 公式に定義しています。

注1:この方法で動作を定義する必要はありません。また、異なる 名前を使用することができます。これは公式な方法の例です。こ の方法は、ドキュメント化とデバッグで利点があります。このセ クションの最後のより簡素な方法に関するフォームの注記を参照 してください。

注2:これらの動作と名前は架空のものです。任意の名前を使用し て必要な数の動作を使用することができます。 次に、画面の到着スクリプトを実際に変更し、これらの複数の動作を処 理する必要があります。以下のDestination画面のサンプル到着スクリプ トのように、これも構造化された方法で実現できます。

```
vHandle_ARRIVE: function(oPayload, oPreviousForm)
 {
  /* Extract a copy of the requested behaviour */
  var RequestedBehaviour = this.RequestedArrivalBehaviour;
  /* Reset the requested behaviour back to the default behaviour */
  this.RequestedArrivalBehaviour = this.ArrivalBehaviours.Default;
  /* Now preform the requested behaviour */
  switch (RequestedBehaviour)
  {
    case this.ArrivalBehaviours.Default:
       SHOW CURRENT FORM(true);
       HIDE_5250_BUTTONS();
       SETBUSY(false);
       break;
    case this.ArrivalBehaviours.SearchNext:
       /* Logic to handle search next page behaviour*/
       break:
    case this.ArrivalBehaviours.SearchLast:
       /* Logic to handle search last page behaviour*/
       break:
    case this.ArrivalBehaviours.ForcedNavigation:
       /* Logic to handle a forced navigation, whatever that may be */
       break:
    case this.ArrivalBehaviours.AutoConfirmation:
       /* Logic to handle a an auto confirmation, whatever that may be */
       break:
    default:
       ALERT_MESSAGE(this.vName,"arrival script –
invalid behaviour requested",RequestedBehaviour.toString());
  }
  /* <ARRIVE /> - Do not remove or alter this line */
  return(true);
 },
```

以上で、画面がサポートする異なる到着動作、およびそれらを実装する ために必要なコードを公式に定義しました。

使用方法

まず、ペイロードを使用する代わりに画面独自のスクリプト内でこれら を使用します。

以下のようなページ検索操作を要求するボタン・クリックを考えてみま しょう。

SENDKEY(KeyPageUp, "SEARCHNEXT");

従来、これはキー・ストロークをサーバーに送信し、ペイロードを含ん でいました。このため、到着スクリプトはフォームが再び戻ってきたと きに実行する内容を認識できます。

注:これを実行した場合、少なくとも2つの動作を提供する到着スクリプトがすでにあった可能性があります。

代わりに、以下のようにコード化します。

this.RequestedArrivalBehaviour = this.ArrivalBehaviours.SearchNext; SENDKEY(KeyPageUp);

注:なぜこのようにするのでしょうか?手間がかかるように感じます。こ こで、すでに1つの利点を得ています。ペイロード・テクニックを使用し て、誤ってSENDKEY(KeyPageUp,"SEARCHNET")とコード化したとしま す。プログラムをデバッグし、"SEARCHNET"が間違ってい て"SEARCHNEXT"にする必要があることがわかるまでしばらく時間がか かります。ArrivalBehaviours.SeachNetをコード化した場合、それを実行 するとスクリプトが失敗し、何か問題があることが即座に通知されま す。

これを使用する2番目の場所はその他の画面です。

例えば、別の画面("AnotherScreen")が複数動作の画面("MultiScreen")に最 終的に到着するまたはその画面を通過することがわかっている一連のイ ベントを発生させようとしています。

また、この画面は"MultiScreen"が到着時に「自動確認」を実行することを確認する必要もあるとします。

"AnotherScreen"には、以下のコードを含めることができます。

var oMS = SCREEN("MultiScreen"); /\* Get a reference to "MultiScreen" \*/
oMS.RequestedArrivalBehaviour = oMS.ArrivalBehaviours.AutoConfirmation;
<< Now execute code to start events that will go to/though "MultiScreen" >>

つまり、"AnotherScreen"は、「到着したときに通常のデフォルトの動作 ではなく自動確認動作を実行する」という、"MultiScreen" (RequestedArrivalBehaviour)のプロパティを設定しています。

"MultiScreen"はDestination画面である必要はありません。これは、Special 画面のJunctionと同等であると考えられます。必要なのは、これが異なる 動作を実行できる必要がある到着スクリプトを含む画面であるというこ とです。

このテクニックは、画面がその間で意図を通信するための非常に公式か つ構造化された方法を示しています。公式でありすぎたり、構造化しす ぎる必要はないばかりでなく、提案されている長い名前を使用する必要 もありません。

以下のように、"MultiScreen"で簡潔に宣言できます。

Action :0, /\* Declare the action code for arriving scripts \*/

#### 次に、以下のように到着スクリプトを構造化します。

Switch (Action) Case :0 Case :1 Case :2 Case :3

他のコードは、this.Action = 2またはSCREEN("MultiScreen").Action = 3を 使用します。

また、以下のように文字列を使用することもできます。

Action :"Default", /\* Declare the action code for arriving scripts \*/

次に、以下のように到着スクリプトを構造化します。

Switch (Action) Case :"Default" Case :"Up"

Case	:"Down"
Case	:"Jump"

他のコードは、this.Action = "Up"またはSCREEN("MultiScreen").Action = "Jump"を使用します。

使用した宣言のテクニックは取るに足りません。また、長くなっていま す。これは、構造化され、ドキュメント化されています。公式な宣言(列 挙型)テクニックの利点は、単にこれが機能の非常に公式なドキュメント であること、および不適切な値を使用した場合、常にコードが失敗する ことです。 デバッグ デバッグと診断 一般的なスクリプト・エラー トレース スクリプトでのALERT\_MESSAGEの使用

#### デバッグと診断

#### トレースをオンにする

デバッグ時に最初に実行する必要のある処理は、「トレース」です。ト レースを確認し、認識されていない画面または空白の名前が付いている 画面を探します。

#### 警告ステートメントを追加する

スクリプトのデバッグを容易にする方法は、「alert()」ステートメントを 追加して、実行時の値をポップアップウィンドウに表示することです。

#### Alert\_Messageファンクションを追加する

同様に、「ALERT\_MESSAGEファンクション」を使用して、値をポップ アップウィンドウに表示できます。

#### トレース・ファンクションを追加する

アプリケーションの実行を中断せずに値をトレースに記録する場合、スクリプト内で「TRACEファンクション」を使用します。

#### フィルターをデバッグする

フィルターをデバッグする場合、フィルター・プログラム内で avRecordTraceメソッドを使用できます。

Invoke avFrameworkManager.avRecordTrace Component(#Com\_Owner) Event('Search Button click handler started')

詳細については、「基本追跡サービス」を参照してください。

[現在の52507ォームを表示(このボタンはデザインモードだけで使用可能)]をクリックします。

オンになることが予測されていない画面でフレームワークが終了した場合、空白の画面が表示され、エラーメッセージ「フォームを表示できない」が表示されます。

デザイン・モードで[スクリーンの調査]ボタンを使用する 現在の画面について何が認識されているかを調査します。 一般的なスクリプト・エラー
 フォームを表示できない
 エラー80020101が原因で操作を完了できない
 Object expected
 スクリプトの不明な動作
 スクリプトがまったく実行されない

## フォームを表示できない

RAMP画面を実行した結果、次のような画面が表示されることがあります。



# このエラーの意味

フレームワークは有効なナビゲーション・パスを作成しました。 ほとんどのスクリプトは、表示されている画面が予期した画面であるこ とを確認します。このため、ほとんどのスクリプトの最後には次のよう な行があります。

/\* Check for arrival at <form name> \*/
if ( !(Q\_CHECK\_CURRENT\_FORM("<form name>","Unable to display form
<form name>")) ) return;

Unable to display formというメッセージは、ナビゲーション中のある段階で、識別された画面が必要だったが、別の画面を受け取ったことを意味します。

Unable to navigateというメッセージは、Destination画面の vHandle\_NAVIGATETOファンクションによって送信されます。これは、 Destination画面のスクリプトを実行する前に、アプリケーションが適切 な画面を表示していることを確認するためのチェックです。これによ り、目的の画面とは異なる画面にキーストロークを入力または送信する のを防ぎます。

定義されていない画面に到達できない場合があります。これは、予期せず表示された画面がナビゲーションを続けるためには削除する必要がある画面のとき、通常は、中断メッセージのときに起こる可能性があります。

#### 対策

[現在の52507ォームを表示(このボタンはデザインモードだけで使用可能)]をクリックして、現在アクティブな画面を表示します。表示される画面は、予期しなかった画面です。

フレームワークメニューのRAMPツールオプションを選択して、RAMP 画面が実行するはずだったナビゲーションを手動で実行します。

- 各画面を移動しているときに、以下の質問に答えます。
- 画面は定義されていますか。

画面のスクリプトを注意して調べます。スクリプトは画面の動作に一致 していますか。

各画面の動作、入力する内容、押すキーがわかっているので、予期しな い画面に手動で到達できるはずです。

# エラー80020101が原因で操作を完了できない いずれかのスクリプトを実行すると、次のようなエラーメッ セージが表示される場合があります。

Could not complete the operation due to error 80020101. Error Could not complete the operation due to error 80020101. detected in script INVOKE\_SCRIPT\_1

このエラーの意味 スクリプトに、スクリプトの実行を妨げる構造上の欠陥があります。例 えば、次のコードをスクリプトに記述します。

if (1 == 2) {

この例では、必要な右括弧}を追加していません。RAMPエディター は、}が不足していることを警告しますが、この警告を無視して進み、ス クリプトを実行します。スクリプトに}が不足しているので、80020101エ ラーが発生します。

}が不足しているので、スクリプト全体がまったく意味をなしません。 同様に、次のコードは、右大括弧を二重に使用しているので、エラーが 発生します。

SETVALUE("utxtBankAccountID",objListManager.AKey3[0]])

対策

スクリプトに次のようなバランスが取れていない部分があるかどうかを 調べます。

- 閉じる/対応する)のない(
- 閉じる/対応する } のない {
- 閉じる/対応する]のない[
- 閉じる/対応する "または'のない"または'(終了していない文字列 定数)
- 閉じる/対応する \*/ のない /\* (終了していないコメント)

構造上正しくない他のJavaScriptの構造

## **Object expected**

# いずれかのスクリプトを実行すると、次のような"Object Expected"エラーが表示される場合があります。

Object expected Error Object expected detected in script INVOKE\_SCRIPT\_1

このエラーの意味

存在しないオブジェクトをスクリプトで参照している可能性がありま す。このエラーの一般的な原因は、単純な入力間違いか大文字小文字の 間違いです。

スクリプトは以下のとおりです。

NaVIGATE\_TO\_JUNCTION("uOS400MainMenu"); NAVIGATE\_TO\_JUNCTIN("uOS400MainMenu");

上記のスクリプトは両方とも"object expected"エラーを引き起こします。 NaVIGATE\_TO\_JUNCTIONおよびNAVIGATE\_TO\_JUNCTINという名前の オブジェクトが実際に存在しないことが原因です。正しいJavaScriptファ ンクション名は、NAVIGATE\_TO\_JUNCTIONです(JavaScriptでは大文字 と小文字が区別されます)。

#### 対策

"Object expected"エラーが発生した場合は、以下を実行してみてください。

- 参照しているオブジェクト名のスペルを確認する
- 参照しているオブジェクト名の大文字小文字を確認する(例: UserprofileまたはUserProfile)。

スクリプトでエラーの原因となっている行を正確に特定するのが難しい 場合があります。

この問題を最も簡単に解決するには、JavaScriptアラート・ファンクションを積極的に使用します。例:

alert("About to navigate"); NaVIGATE\_TO\_JUNCTION("uOS400MainMenu"); alert("Navigation finished"); かなり早く、NaVIGATE\_TO\_JUNCTION()行がスクリプト・エラーの原 因となっている行であることがわかります。

## スクリプトの不明な動作

スクリプトでの不明な動作は、たいていの場合、"=="比較を正しく使用 していないことが原因で発生します。次の簡単なスクリプトを使用し て、一般的で、時間を浪費するスクリプトの問題について説明します。

```
var X = 1;
alert ("X is " + X);
if (X = 2)
{
    alert("X is 2");
}
```

このスクリプトを実行すると、1番目のアラート・メッセージはXが1で あると表示しますが、2番目のアラート・メッセージはXが2であると表 示します。これはありえません。

この問題は、ifステートメントが次のようになっていないことが原因で 発生します。

```
if (X == 2)
{
alert("X is 2");
}
```

スクリプトがまったく実行されない

スクリプトがまったく実行されてないように見える場合があります。 通常これは、スクリプトが、考えたとおりの方法で5250画面ナビゲー ション内に呼び出されていないためです。

フレームワーク)->(トレース中)->アプリケーションレベルメニュー・ オプションを使用して、アプリケーション内の制御フローをトレース し、ナビゲーションを詳細に把握します。通常これによりスクリプトが 呼び出されない理由が明らかになります。 トレース

トレースは、フレームワークをデザイン・モードで実行中にいつでも開 始できます。

アプリケーションレベルトレース機能を使用してRAMPの実行をトレースします。トレースを開始するには、フレームワークメニューをクリックし、トレース->アプリケーションレベルを選択します。

トレース・ステートメントがトレースウィンドウに表示されます。

RAMPの実行によって多数のステートメントが生成される可能性があり ます。また、ウィンドウを大きくしなければ全体を表示できないくらい 長いステートメントが生成されます。

7777	ねン レベル・トレースの詳細 🔳 🗖 🔀
コンホペーネント	イላንኦ
VF_SY150	Checking lock status (VF_UM037.TabChan
VF_SY150	Returning lock status FALSE (VF_UM037.T
VF_CH006	Command Handler - uTerminate executed
EOMRMP03	Command Handler - uInitialize executed
EOMRMP03	Command Handler - uExecute executed
EOMRMP03	VF_SY122 - Session key = 1.RAMP_DEFAU
EOMRMPU3	VF_SY122 - VF_SY12U already exists - ses
FOMPMP03	VE SV122 - Call VE SV122 utpitialize in VE
VE LM002	GetCurrentInstance returns VisualID1(A10
FOMRMP03	VE_SV122 - Firing signal RampTSAvailable
EOMRMP03	VF_SY122 - Navigate to screen named Dis
VF SY120	VF SY120 - Navigate to screen DisplayEm
EOMRMP03	Ramp handling request Navigate to Screen
EOMRMP03	NAVIGATE_TO_SCREEN invoked to naviga
EOMRMP03	Preparing a navigation plan to get to scree
EOMRMP03	Preparation of final navigation plan to get
EOMRMP03	Preparing a navigation plan to get to scree
EOMRMP03	Destination screen =UNKNOWN_FORM= h
EOMRMP03	An optimized pre-existing navigation plan f
EOMRMP03	The attempt to make a navigation plan fail
VF_SY150	Checking lock status (VF_UM037.TabChan
VF_SY150	Returning lock status FALSE (VF_UMU37.1
EOMRMPU3	Command Handler - Uterminate executed
VF_CHOUI	Command Handler - utritialize executed
Vr_Choor	Command Handler - difficialize executed
	<u>~</u>
ł	レースファイルを保存

RAMPの実行のトレースでは、ファイルのトレースを保存ボタンを使用 して、トレースをtempディレクトリのテキストファイルに保存すること をお勧めします。生成されたトレースファイルの正確な場所とファイル 名は、メッセージに表示されます。

メッセージボタンをクリックすると、トレースファイルの場所を確認で きます。



# 独自のステートメントの追加

付属のJava Script関数TRACE()を使用して、独自のトレース・ステートメントをアプリケーションレベルトレースに追加できます。そのトレース・ステートメントの出力は、アプリケーションレベルトレースウィンドウに送信されます。

例えば、次のトレース・ステートメントは、

TRACE("");

TRACE("Value of AKEY1 is =>" + objListManager.AKey1[0] + "<="); TRACE("");

次のトレースを生成します。

🔳 アプリケーション	V UN*11 HU-	
コンポーネント	ウィンドウ	74%
VF_CH006	MAIN	SETVALUE of txtEmpno completed. Current form is
VF_CH006	MAIN	SENDKEY Enter requested. Current form is named F
VF_CH006	MAIN	Posting screen. Current form is named FindEmploye
VF_CH006	MAIN	Execution of script associated with FindEmploye
VF_CH006	MAIN	SENDKEY Enter completed. Server response will be
VF_CH006	MAIN	Q_CHECK_CURRENT_FORM request queued. Chec
VF_CH006	MAIN	Execution of script associated with FindEmployee com
VF_CH006	MAIN	Form with name "DisplayEmployee" has arrived. (11:40:5:
VF_CH006	MAIN	Executing vHandle_ARRIVE function in form DisplayEn
VF_CHOOG	MAIN	(11: 10: 51: 579)
VF_CH006	MAIN	Value of AKEY1 is =>A1003<= (11:40:51:579)
VF_CH006	MAIN	(11.40.51.579)
VF_CH006	MAIN	SHOW_CURRENT_FORM executed to show the cur
VF_CH006	MAIN	Form named DisplayEmployee is defined and classifi
VF_CH006	MAIN	Form named DisplayEmployee has defined function
VF_CH006	MAIN	Enabled as buttons: Enter PROMPT 変更 削标 (1
VF_CHU06	MAIN	Enabled as keys : Enter F4 F21 F22 (11:40:51:5
VF_CH006	MAIN	Massage ADDED: MODE DECODDS = false (11:40
VF_CH006	MAIN	Message ADDED; MORE RECORDS = Talse (11:40
VF_CH006	MAIN	Execution of action associated with DisplayEmployee c
VF_CH006	MAIN	Execution of script associated with DisplayEmployee C -
		i.i. 🔳
		トレースファイルを保存

実際のトレース・ステートメントの前後にあるブランク行は、単に読みやすくするためにTRACE("")によって生成されます。

トレース・ステートメントの詳細は、「スクリプト・ファンクション」 を参照してください。 スクリプトでのALERT\_MESSAGEの使用

メッセージボックスを表示することが、問題をデバッグする一番簡単で 早い方法の場合があります。

スクリプトでALERT\_MESSAGE()を使用すると、定義済みメッセージとともにダイアログボックスが表示されます。

ALERT\_MESSAGE()は、テキストと可変値を組み合わせて表示することも可能です。

例えば、いずれかのスクリプトで、そのスクリプトに渡されるAkeyの値 を表示する場合は、ALERT\_MESSAGE()を次のように使用します。

ALERT\_MESSAGE("The value of AKEY1 is =>" + objListManager.AKey1[0] + "<=");

これにより、実行中に次のようなメッセージボックスが表示されます。

Empno : Example 1 (A1008-ALLAN SNEDDON)	
The value of AKEY1 is ==> AUD <==	
ОК	
10 New York Contraction of the C	•

画面の拡張

刷新された5250画面の外観をアプリケーションで変更し、表示を改善して、よりWindowsに近づけたい場合があります。

RAMPレイアウト寸法を使用して画面のタイトルを非表示にするなど、 内容を簡素にしたり(「個々のRAMP画面での画面タイトルの非表示」を 参照)、aXes eXtensions(「RAMP-TSAD04:aXesを使用して画面を再設計 する」を参照)、ドロップダウン、チェックボックス、ラジオ・ボタン、 カレンダー、グラフ、ハイパーリンク、イメージ、フルカラー・パレッ トを使用して画面を自由に再設計したりすることができます。

また、RAMPコマンド・ハンドラーの拡張に使用できるHTMLAPIスクリ プト・オブジェクトもあります。これはaXes eXtensionsに先行し、その 使用には確かなJavaScriptおよびHTML DOMスキルが必要です。

画面を拡張する場合は、コマンド・ハンドラーでのスペースが限られて いること、また追加のコンテンツ用の別のコマンド・ハンドラー・タブ を作成することは簡単で迅速であり、イメージ、Googleマップ、その他 をすでにいっぱいになっている5250画面に押し込むよりも通常は良好な ソリューションであることを念頭に置いてください。

## 個々のRAMP画面での画面タイトルの非表示

ほとんどの場合、5250画面タイトルは、RAMP画面では必要ありません。フレームワークウィンドウのナビゲーション・エレメントによって、作業中のオブジェクトおよび実行中のコマンドがわかりやすく示されます。

😹 Human Relations 📃 🗖 🔀				
<u>File Edit View Actions Tools H</u> elp (Framework)	(Administration)			
💥 Sign Off 🛞 New 💌 🚘 Email 🖾 Prin	t 🖂 🗂 Deta	ils 💌 🔁 Transfer	🗊 Calculator	
Employee				
On Tool Bar Human Relations Document Employee Invoices Document	name. Employee A3564 A0090 A0070 A1031	Description FREDDY BROWN FRED JOHN ALAN BLOGGS VERONICA BROWN JOHN BLAKE		
Report	arch		Clear List	
Employee : Edit (A0070-VERC	NICA BROWN)			
🗈 🥨 Programming Tec 📝 Edit 🔗 Email 💭 Video	5			
Employee Number Employee Surname Employee Given Name(s) Street No and Name Suburb or Town State and Country Home Phone Number Department Code Section Code Start Date (DDMIMYY) Termination Date (DDMIMYY)	Browse/Maintain Emp	A0070 BROWN VERONICA ANN 12 Railway Stre Baulkham Hills NSW Australia (02) 639-4627 TRVL 03 28/01/90 0/00/00		
Date Ski Skil Skil Descrip	Comment	t Grade		
0/00/00				
Messages R	eady Local	ENG DCXUSER 3	3/14/06 9:20	

このため、RAMP画面は、フレームワークにタイトルがないほうがより 自然に見えます。

Human Relations	
<u>File E</u> dit <u>View A</u> ctions <u>T</u> ools <u>H</u> elp (Framework) (Admi	inistration )
🎽 Sign Off 🛞 New 💌 🔗 Email 🖾 Print 🖸	🛛 🗂 Details 💌 🖗 Transfer 🗊 Calculator
Employee	
On Tool Bar       Specify a full or partial employee name.         Human Relations       Employee Surname         Employee       Invoices         Payment       Clear the current lis         Report       Search	Employee       Description         A3564       FREDDY BROWN         A0090       FRED JOHN ALAN BLOGGS         A0070       VERONICA BROWN         A1031       JOHN BLAKE
O Demo Application     Demo Application     Programming Teq	BROWN)
Employee Number Employee Surname Employee Given Name(s) Street No and Name Suburb or Town State and Country Home Phone Number Department Code Section Code Start Date (DDMMYY) Termination Date (DDMMYY)	A0070 BROWN VERONICA ANN 12 Railway Stre Baulkham Hills NSW Australia (02) 639-4627 TRVL 03 28/01/90 0/00/00
Date Skl Skill Code Description 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/00/00 0/	Comment Grade
Messages Ready	Local ENG DCXUSER 3/14/06 9:23

タイトルを非表示にする2つの方法があります。

### タイトルを非表示にする2つの方法

タイトルが隠れるまでRAMP画面を上に移動するか、タイトルにマスク を適用することによって、画面タイトルを非表示にできます。

RAMP画面の位置とマスクをRAMPツールウィンドウのデフォルト セッションのセッションの詳細で設定します。

RAMP-TS 5250	セッション 詳細	)				
ーセッション - デウオル	レト セッション――					
キャプション	デフォルト セッション	2				
ユーザー オフジェ	ንト ጵイフ°	RAMP_D	EFAULT_	SESSION		名前の確認
-RAMPの既定	値のレイアウトサ	ビス*――			2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 2000 - 20	
高さ 380	幅	700	上	25-	左	
Top マスクの高	ið	Bott	tom マスク	の幅		
RAMP スカリー	2 64791 2941	l				
• E	定レイアウト			🔘 70-l-	(ፖウト	J
スクロールパー						
□ 水·	平スケールバーを	き表示		📃 垂直ス	加ールバーを新	表示

「Destination画面の詳細」でレイアウト寸法を変更することによって、 個々のDestination画面に対するこの設定をオーバーライドできます。

#### 画面の移動

タイトルが隠れるまで画面を上に移動するには、上プロパティに負の値 を設定します。

RAMP-TS 5250	セッション 詳細		
ーセッション - デフォ	い セッション――		
キャプション	デフォルト もっション	2	
ユーザー オフジェ	ንト ዓイフ°	RAMP_DEFAULT_SESSION	名前の確認
_RAMPの既定	値のレイアウトサ	Hz	
高さ 380	帽	700 上 25- 左	
Top マスクの高	iđ	Bottom マスクの幅	
RAMP スカリー	> 61791 2911	16	
• E	定レイアウト	🔘 ንローレイアウト	J
~スクロールパー-			
	平スクロールバーを	を表示 📃 垂直スクロールパー	を表示

タイトルのマスキング

タイトルをマスキングするには、Topマスクの高さプロパティをタイト

# ルを覆う高さに設定します。

RAMP-TS 5250	セッション 詳細		
~セッション - デウオ	レト セッション		
キャプション	デンオルト セッション		
ユーザー オフジェ	ንト タイフ°	RAMP_DEFAULT_SESSION	名前の確認
_RAMPの既定	目値のレイアウト サイス	*	
高さ 380	幅	700 上 左	
Top マスクの高	5さ 25	Bottom マスクの幅	
RAMP スクリー	ን レイアウト スタイル		
• E	定レイアウト	🔘 ንローレイアウト	J
-スクロールパー			
□ 水·	平スケールバーを表	示 📃 垂直ス知ールバーを表	<b>€</b> 示

RAMP Webには適用されません。

## HTMLAPIスクリプト・オブジェクト

HTMLAPIスクリプト・オブジェクトを使用して、5250画面のビジュア ル・コンテンツを動的に変更することができます。

RAMP-TSで5250画面を表示すると、実際はHTMLドキュメントが表示されています。5250画面を表すHTMLドキュメントは、他のHTMLドキュメントと同様です。これはHTMLを基本としているため、RAMPスクリプトからアクセスおよび変更できるDOM (Document Object Model)を提供します。

HTMLAPIスクリプト・オブジェクトは、現在表示されている5250画面の HTML DOMにアクセスできる小規模な一連のメソッドを提供します。 HTML DOMを使用することで、表示されている5250画面の外観と動作を 変更できます。

Employee Nu	imber		A1004		10.02	
mployee Su	rname		SMITHSON			
mployee Giv	ven Name(s)		PAUL			(COMA)
Street No and	d Name		41 William	Rd.		an P
uburb or To	wn		St Ives			25 3
State and Co	untry		NSW		2144	A CONTRACTOR
lome Phone	Number		419 5656			
epartment C	Code		AUD 廷			4
ection Code			03 🏓			
tart Date (DI	DMMYY)		1/05/80	€		
ermination D	Date (DDMMY	Y)	0/00/00	(F)		
Date Skl Acquired	Skill Code	Skill Description		Comment	Grade	
25/03/98	ADMIN1	Administratn Part 1		Met requirement	D	
5/02/98	ADMIN2	Administratn Part 2		Met requirement	P	
4/05/98	COM	Communications Degre			D	
E (0E (00	22	Lompliter Science Deg			P	

HTMLAPIスクリプト・オブジェクトを使用するには何を理解 する必要がありますか?

JavaScriptおよびHTML DOMの確かなスキルを持っている必要があります。

HTMLAPIスクリプト・オブジェクトの使用や5250画面の HTML DOMへのアクセスに伴うリスクは何ですか? HTMLAPIスクリプト・オブジェクトの使用や5250画面のHTML DOMへ のアクセスでは、次のようなリスクがあります。

- IE HTML DOMモデルの内容や動作の将来的な変更
- aXes 5250 HTMLドキュメント・レイアウトの内容や動作の将来的 な変更
- aXesクライアント側ロジックの動作の混乱や変更

これらのリスクの受け入れ、テスト、管理の準備が整ったら、HTMLAPI で5250 RAMP-TS画面の内容や動作に対して動的で効率的な変更を行う ことができます。

HTMLAPIスクリプト・オブジェクトを使用する HTMLAPIの使用例

## HTMLAPIスクリプト・オブジェクトを使用する

5250画面のビジュアル・コンテンツを動的に変更するときの推奨手法は次のとおりです。

- 可能な場合はいつでも一般的な内容の変更を実装するSHAREDファンクションを作成します。これにより、SHAREDファンクションを他のRAMP-TSスクリプトから再利用することができます。こうすることで、HTMLモデルやaXes DOMモデルの将来の変更が及ぼす影響が若干少なくなります。
- 内容の変更が特定の5250画面のみに適用される場合は、明確な名前 を持つ画面スクリプト・オブジェクト内の1つのファンクションで実 装します。こうすることで、特定の画面レイアウトを変更するロジッ クが若干標準化され、認識しやすくなり、1つの場所に統合されま す。

例:

サンプル・ファンクションの 名前	その役割
SHARED.ApplyStandardLayout()	Destination画面に一般的なアプ リケーション・レイアウト規則 を適用します。ほとんどの Destination画面は、画面が表示 される前にその到着スクリプト でこのファンクションを呼び出 します。
SHARED.AttachPromptImages()	画面フィールド名の配列を受け 取ります。各名前フィールドの 横にクリック可能な小さいイ メージを作成します。通常、増 進可能なフィールドを表示する 5250画面はこのファンクション を使用します。
thisScreen.ApplySpecificLayout()	特定の5250 Destination画面の定 義内のファンクションとして定 義されます。この画面に固有の

## HTMLAPIの使用例

次のサンプルでは、プロセスPSLSYSファンクションINQUIREとして LANSAに組み込まれている5250 Destination画面を使用します。これを使 用すると、社員の詳細を表示し、更新できます。最初は、次のように到 着スクリプトを含んでいます。

```
vHandle_ARRIVE: function(oPayload, oPreviousForm)
{
 /* If the department input field exists on the screen, display it */
 if ( CHECK FIELD EXISTS("DEPTMENT") )
 {
  SHOW CURRENT FORM(true);
  HIDE 5250 BUTTONS();
  SETCURSORTOFIELD("SURNAME");
  SETBUSY(false);
 }
 /* Otherwise send an F21 key to make the screen input capable */
 else
 {
  SENDKEY(KeyF21);
 }
 /* <ARRIVE /> - Do not remove or alter this line */
 return(true);
},
```

次のように表示されます。

	資格情報の授柔/保守
社員Na	: A1002
社員姓	SMYTHE
社員名	JOHN
住所1	20 Cobbitty Avenue,
住所 2	WERRINGTON.
住所3	NSW. 2100
自宅電話番号	047 629 0442
部門コート	• • • • • ADM +
課コード	· · · · 02 +
人社日付 (DDMMYY)	77/01/01 +
退職日付 (DDMMYY)	0/00/00 +
取得日 資格	
(DDMMYY) I - F	資格記述 コメント 資格
98/03/25 ADMIN1	Administratn Part 1 Met requirement D
98/05/02 ADMIN2	Administratn Part 2
98/05/05 CS	Computer Science Deg
98/05/04 ENG	English Degree D
98/02/05 INTRO	Company Introduction Met requirement F +

この5250画面には、次の名前のフィールドがあります。

SURNAME	社員姓
GIVENAME	社員名
ADDRESS1	住所1
ADDRESS2	住所2
ADDRESS3	住所3
DEPTMENT	部門コード
SECTION	課コード
DATE_START_DDMMYY	入社日付
DATE_END_DDMMYY	退職日付
DATE_ACQ_DDMMYY	取得日(サブファイル列)
PageDownMarker	サブファイルにさらにデータがあるこ とを示す"+"記号

これらのフィールドは単なる例であることに注意してください。5250画面で何を実行するかやどのように操作するかは異なります。要件に合わ

せて使用する手法を調整する必要があります。 基本的な標準レイアウト・ファンクションを実装する 標準レイアウト・ファンクションにより画面を汎用的に変更する 特定のレイアウト・ファンクションを使用して個別に画面を変更する 標準の汎用ハンドラーにさらに機能を追加する サブファイルの見出しを変更する フォントを変更する イメージを追加する 注意事項

HTMLAPIファンクションで提供される内容
### 基本的な標準レイアウト・ファンクションを実装する 最初に、以下のファンクションをSHAREDスクリプト・オブジェクトの 一部としてuf\_sy420\_rts.jsファイルに追加します。

```
var SHARED =
{
   /* ------ */
   /* Apply standard layout changes to arriving screens */
   /* ------ */
   ApplyStandardLayout : function()
   {
    /* Use the HTMLAPI to hide lines 1 and 2 on all screens */
    HTMLAPI.hideRow(1);
   HTMLAPI.hideRow(2);
   },
Etc, Etc
```

```
以上で、RAMP画面の到着スクリプトから呼び出すことができる、
SHARED,ApplyStandardLayoutという標準ファンクションを得ました。
例えば、サンプルDestination画面の到着スクリプトを以下のように変更
して、スクリプトにより5250画面が表示される直前にこの新しいファン
クションを使用します。
```

```
vHandle_ARRIVE: function(oPayload, oPreviousForm)
{
    /* If the department input field exists on the screen, display it */
    if ( CHECK_FIELD_EXISTS("DEPTMENT") )
    {
        SHARED.ApplyStandardLayout();
        SHOW_CURRENT_FORM(true);
        HIDE_5250_BUTTONS();
        SETCURSORTOFIELD("SURNAME");
        SETBUSY(false);
    }
    /* Otherwise send an F21 key to make the screen input capable */
    else
    {
        SENDKEY(KeyF21);
        SENDKEY(KeyF21);
    }
}
```

```
}
/* <ARRIVE /> - Do not remove or alter this line */
return(true);
},
```

実行すると、結果の画面は以下のようになります。

社員Na		A0193	
社員姓		SMITHSON	
社員名		FRED	
住所1		121 Cutler Ave	
住所 2		Windsor	
住所3		NSW	2034
自宅電話番号		(02) 546-4657	
部門コート		ADM +	
課コード		05 +	
人社日付 (DDMMYY)		89/07/03 +	
退職日付 (DDMMYY)		0/00/00 +	
取得日 资格			
DDMMYY) I-	下 资格記述	逝 コメント	资格
98/03/25 ADMIN	Administ	rath Part 1 Met requirement	D
98/05/04 COM	Communic	ations Degre	D
98/05/05 CS	Computer	Science Deg	F
98/05/02 HIS	History	Degree	F
98/02/05 INTRO	Company	Introduction Met requirement	F +

画面の行1と2が非表示になります。

これは非常にささいな汎用レイアウト規則ですが、多くの異なる5250画 面にわたってより重要な規則を汎用的に実装できる基盤を得ました。

### 標準レイアウト・ファンクションにより画面を汎用的に変更す る

実行することがある一般的な別の操作として、画面からすべての"+"プロ ンプト・フィールドを削除し、すべてのフィールド・ラベルから末尾の ドットを削除する場合があります。 これを実行するには、以下のようなコードを SHARED, ApplyStandardLayoutファンクションに追加します。 /\* Get all elements between lines 3 and 22 \*/ var aH = HTMLAPI.getElementsinRowRange(3,22); /\* Hide all fields containing "+" signs and strip trailing dots from others \*/ for (i = 0; i < aH.length; i++) { var oH = aH[i]: if ((oH != null) && (typeof(oH.tagName) != "undefined")) { **var** fIsINPUTField = ((oH.tagName == "**INPUT**") || (oH.tagName == "**TEX**" if ( !(fIsINPUTField) ) /\* This NOT an input field on the screen \*/ { if (oH.innerText == "+") HTMLAPI.hideElement(oH); else HTMLAPI.stripTrailingDots(oH); } }

実行すると、サンプルの5250は以下のようになります。

社員施																					
社員Na																					
社員姓	社員	t No.		•	•									:	A1004						
社員名 · · · · · · PAUL 住所 1 · · · · · · · · · · · · · · · · · ·	社員	乱姓											•		SMITHSON						
住所 1	社員	18						•					•	•	PAUL						
住所 2 GRANVILLE. 住所 3 NSW. 2144 自宅電話番号 419 5656 部門コート ADM 課コード 03 人社日付 (DDMMYY) 80/05/01 退職日付 (DDMMYY) 80/05/01 退職日付 (DDMMYY) 0/00/00 取得日 資格 (DDMMYY) コード 資格記述 コメント 資格 98/03/25 ADMIN1 Administratn Part 1 Met requirement D 98/05/04 COM Communications Degre C 98/05/05 CS Computer Science Deg F 98/02/05 INTRO Company Introduction Met requirement F 98/02/05 INTRO Company Introduction Met requirement F	住民	<i>f</i> i 1				•							•		41 William F	Road					
住所3 NSW. 2144 自宅電話番号 419 5656 部門コート ADM 課コード 03 人社日付 (DDMMYY) 03 人社日付 (DDMMYY) 03 取得日 資格 (DDMMYY) コード 変格記述 コメント 資格 98/03/25 ADMIN1 Administratn Part 1 Met requirement ロ 98/05/04 COM Communications Degre ロ 98/05/05 CS Computer Science Deg F 98/02/05 INTRO Company Introduction Met requirement F 98/02/05 INTRO Company Introduction Met requirement F	住良	f 2		•	•	•	•	•	•		•		•		GRANVILLE.						
自宅電話番号	住民	fЗ			•	•		•		•			•		NSW.					2144	
部門コート	自己	6 電話	番	号					•		•		•		419 5656						
課コード	광부	9 1	٢		•	•	•	•			•		•	•	ADM						
人社日付 (DDMMYY)80/05/01 退職日付 (DDMMYY)0/00/00 取得日 資格 (DDMMYY) コード 資格記述 コメント 資格 98/03/25 ADMIN1 Administratn Part 1 Met requirement D 98/05/04 COM Communications Degre D 98/05/05 CS Computer Science Deg F 98/02/05 INTRO Company Introduction Met requirement F 98/02/05 KEY Keyboard Skills Met requirement F	課:	1 - F		•	•		•	•		•	•		•		03						
退職日付 (DDMMYY) 0/00/00 取得日 資格 (DDMMYY) コード 資格記述 コメント 資格 98/03/25 ADMIN1 Administratn Part 1 Met requirement D 98/05/04 COM Communications Degre D 98/05/05 CS Computer Science Deg F 98/02/05 INTRO Company Introduction Met requirement F 98/02/05 KEY Keyboard Skills Met requirement F	人主	t日何	(	D	DM	IY	Y)	•	•			•	•	•	80/05/01						
取得日 資格 (DDMMYY) コード 資格記述 コメント 資格 98/03/25 ADMIN1 Administratn Part 1 Met requirement D 98/05/04 COM Communications Degre D 98/05/05 CS Computer Science Deg F 98/02/05 INTRO Company Introduction Met requirement F 98/02/05 KEY Keyboard Skills Met requirement F	退耳	計日付	(	D	DM	٩Y	Y)	•					•		0/00/00						
取得日 資格 (DDMMYY) コード 資格記述 コメント 資格 98/03/25 ADMIN1 Administratn Part 1 Met requirement D 98/05/04 COM Communications Degre D 98/05/05 CS Computer Science Deg F 98/02/05 INTRO Company Introduction Met requirement F 98/02/05 KEY Keyboard Skills Met requirement F																					
(DDMMYY)     コード     資格記述     コメント     資格       98/03/25     ADMIN1     Administratn Part 1     Met requirement     D       98/05/04     COM     Communications Degre     D       98/05/05     CS     Computer Science Deg     F       98/02/05     INTRO     Company Introduction     Met requirement     F       98/02/05     KEY     Keyboard Skills     Met requirement     F	取得	98			-	資料	格														
98/03/25     ADMIN1     Administratn Part 1     Met requirement     D       98/05/04     COM     Communications Degre     D       98/05/05     CS     Computer Science Deg     F       98/02/05     INTRO     Company Introduction Met requirement     F       98/02/05     KEY     Keyboard Skills     Met requirement     F	(DDM	MYY)				-	-	F			泽	8 格	10	: 过		Ξ.	XVI	•		資格	
98/05/04     COM     Communications Degre     D       98/05/05     CS     Computer Science Deg     F       98/02/05     INTRO     Company Introduction Met requirement     F       98/02/05     KEY     Keyboard Skills     Met requirement     F		98/0	3/	25	A	DM	IN	1		1	Adr	nir	113	str	ath Part 1	Met	requ	iremen	t	D	
98/05/05 CS Computer Science Deg F 98/02/05 INTRO Company Introduction Met requirement F 98/02/05 KEY Keyboard Skills Met requirement F		98/0	5/	04	C	OM					Cor	nmu	ini	ica	ations Degre					D	
98/02/05 INTRO Company Introduction Met requirement F 98/02/05 KEY Keyboard Skills Met requirement F		98/0	5/	05	C	s					Cor	npu	Ite	er	Science Deg					F	
98/02/05 KEY Keyboard Skills Met requirement F		98/0	2/	05	I	NT	RO	)			Cor	npa	In	1 ]	introduction	Met	requ	iremen	t	F	
		98/0	2/	05	K	EY					Ke)	/bo	a	d	Skills	Met	requ	iremen	t	F	

特定のレイアウト・ファンクションを使用して個別に画面を変 更する

実行できる5250画面固有の内容として、表示されるサブファイルにスク ロール・ボタンを追加する場合があります。

このサンプル画面に関連付けられているスクリプトでは、以下のように3 つの新しいファンクションを追加できます。

/\* Apply layout changes specific to this screen \*/
ApplySpecificLayout : function()

{

if (CHECK\_FIELD\_EXISTS("PageDownMarker"))

HTMLAPI.insertSubFileScrollers("/ts/skins/images/pageup.gif",this.HandlePageelse

HTMLAPI.insertSubFileScrollers("/ts/skins/images/pageup.gif",this.Handl },

/\* Handle clicks on the subfile scroller images images \*/

HandlePageDown: function() { EXECUTE\_BUTTON\_SCRIPT(KeyPageDow HandlePageUp: function() { EXECUTE\_BUTTON\_SCRIPT(KeyPageUp); }

また、画面が到着するたびにこの新しいロジックが呼び出されるように 画面定義の到着スクリプトを変更します。

vHandle\_ARRIVE: function(oPayload, oPreviousForm)
{
 var bReturn = true;
 /\* If the department input field exists on the screen, display it \*/
 if ( CHECK\_FIELD\_EXISTS("DEPTMENT") )
 {
 SHARED.ApplyStandardLayout();
 this.ApplySpecificLayout();
}

etc, etc

実行すると、サンプルの5250画面は以下のようになります。

Employee Numb	er		A1004					
Employee Surn	ame		SMITH	ISON				
Employee Given Name(s)								
Street No and Name				41 William Rd.				
Suburb or Town				/es				
State and Cou	ntry		NSW		2144			
Home Phone Nu	mber		419 5	65 6				
Department Co	de		AUD					
Section Code								
Start Date (D	DMMYY)		1/05	/80				
Termination D	ate (DDMMY	Y)	0/00	/00				
Date Skl Acquired	Skill Code	Skill Description		Comment	Grade			
25/03/98	ADMIN1	Administratn Par	t 1	Met requirement	D			
5/02/98 ADMIN2 Administratn Par 4/05/98 COM Communications D			t 2	Met requirement	P			
			egre		D			
4/05/98			-	-				

Page UpとPage Downのクリック可能なイメージがサブファイルの下に表示されていることに注意してください。クリックすると、ハンドラー・ファンクション、HandlePageUpとHandlePageDownが呼び出され、Page Up/Page Downのキーストロークがサーバーに送信されます。

注:これは、vHandle\_BUTTONCLICKファンクションを実行して行われま す。このため、このファンクションでPage UpおよびPage Downキーを正 しく処理できることを確認する必要があります。

#### 標準の汎用ハンドラーにさらに機能を追加する

このサンプルで使用されているRAMP-TSセッションには、以下のように 特別なフィールド受け渡しテーブルがあります。

0	特历	切なフィールト党け渡し			
		52507ィール*名	ファンクションキー	VL バント・ラー (クラス VF_AC017 オフ・シュ・エクト)	^
	1	DEPTMENT	F4	DF_PRM04	
	2	SECTION	F4	DF_PRM04	
	3	DATE_*	F4	DF_PRM07	
	4		F4		
	5		F4		
l	6		F4		~

この意味がわからない場合は、特別なフィールド受け渡しのチュートリアルを終了する必要があります。

特別なフィールド受け渡しテーブルを使用すると、サンプル画面でこれ らのフィールドの自動プロンプトが有効になります。

Employee Number	A1008							
Employee Surname	SNEDD	ON						
Employee Given Name(s)	ALLAN	1						
Street No and Name	22 Ra	ilway	/ Pa	rad	e,			
Suburb or Town	KOGAR	AH						
State and Country	NSW.							216
Home Phone Number	476 2	198						
Department Code	AUD							
Section Code	01							
Start Date (DDMMYY)	20030	9 44	•	Ma	rch 3	2009	•	+ ++
Termination Date (DDMMYY)	0/00	Mon	Tue	Wed	Thu	Eri	Sat	Sun
		23	24	25	26	27	28	1
		2	3	4	5	6	7	8
Date Sk1 Skill	Skill	9	10	11	12	13	14	15
Acquired Code	Administrato Part 1	16	17	18	19	20	21	22
1/05/98 ADMINI A	Administratn Part 2	23	24	25	26	27	28	29
5/05/98 C5 C	Computer Science Deg	30	31	1	2	3	4	5
5/02/98 INTRO C	Company Introduction	N S	To	day:	20/0	03/2	009	
2/02/20 2////00	company and canceron			-				

これは、[入社日付]フィールドにカーソルを置き、F4を押した場合の画面です。特別なフィールド・ハンドラーDF\_PRM07によりカレンダーが表示され、ユーザーは日付を選択することができます。

これは、プロンプト可能なフィールド内にカーソルを置いてF4ファンク ション・キーを使用した場合にのみ実行できます。 F4プロンプト・ロジックは、HTMLAPIを経由して、また汎用的な SHARED.ApplyStandardLayoutファンクションを使用して汎用的に拡張することができます。 まず、SHARED.ApplyStandardLayoutを以下のように変更してオプションのパラメータを取得します。

ApplyStandardLayout :function(aPromptFields)

次に、以下のようなコードをApplyStandardLayoutに追加します。

```
/* Insert prompting images */
if (aPromptFields != null)
{
    for (i = 0; i < aPromptFields.length; i++)
    {
        oH = HTMLAPI.getElementbyName(aPromptFields[i]);
        if (oH != null)
        {
            oI = HTMLAPI.insertImage(oH,"/ts/skins/images/zoom_in_18x18.gif'
            oI.PromptFieldName = aPromptFields[i];
        }
    }
}</pre>
```

aPromptFields == nullを確認することで、設計でパラメータを省略可能に することができます。呼び出し側はこれを渡す必要はありません。 SHAREDオブジェクトには、作成したイメージのクリックを処理するた めに以下の例のようにファンクションを追加する必要もあります。

```
/* ----- */
/* Handle clicking on a prompt image */
/* ----- */
HandlePromptImageClick : function(oE)
{
    var oI = oE.srcElement;
    if (typeof(oI.PromptFieldName) != "undefined")
    {
}
```

```
SETCURSORTOFIELD(oI.PromptFieldName);
EXECUTE_BUTTON_SCRIPT(KeyF4);
}
```

最後に、SHARED.ApplyStandardLayoutを使用しているサンプル5250 Destination画面を変更し、プロンプト可能なフィールドの配列を渡す必 要があります。

まず、スクリプト・コードの先頭で以下のように配列を宣言します。

aPromptFields :Array("DEPTMENT","SECTION","DATE\_START\_DDMMYY","DATE\_ENC

SHARED.ApplyStandardLayoutの呼び出しを変更します。

SHARED.ApplyStandardLayout(this.aPromptFields);

結果の5250画面は、以下のように表示されます。

Employee Numb	ber		A1002		
Employee Surr	name		SMYTH	ESON	
Employee Give	en Name(s)		JOHN		
Street No and	i Name		20 C 0	bbitty Avenue,	
Suburb or Tow	vn		WERR I	NGTON.	
State and Cou	untry		NSW		2100
Home Phone Nu	umber		047 6	29 0442	
Department Co	ode		ADM 🍃	Ð	
Section Code			05 🕀		
Start Date (D	DMMYY)		18030	9 🗩	
Termination D	ate (DDMM)	Y)	10030	9 🕀	
Date Skl	Skill Code	Skill Description		Comment	Grade
Acquired	10117114	Administratn Par	't 1	Met requirement	D
25/03/98	ADMINI				
25/03/98 2/05/98	ADMIN1 ADMIN2	Administratn Par	t 2		F
25/03/98 2/05/98 5/05/98	ADMIN1 ADMIN2 CS	Administratn Par Computer Science	t 2 Deg		F

プロンプト可能なフィールドの横に小さなイメージが表示されているこ

とに注意してください。

ユーザーはイメージをクリックしてフィールドをプロンプトするか、 フィールドにカーソルを置いてF4を押すことができます。結果は同じで す。

イメージをクリックすると、実際は現在の画面の

vHandle\_BUTTONCLICKファンクションを呼び出しています。このため、このファンクションはF4キーを処理できる必要があります。

### サブファイルの見出しを変更する

HTMLAPIが提供するその他の機能の1つに、サブファイルの見出しの動的な変更があります。通常、この機能は5250画面の汎用的な方法で使用できます。

以下のようなコードを標準のSHAREDレイアウト・ファンクション、 ApplyStandardLayoutに追加した場合を考えてみましょう。

/\* Adjust the subfile headings \*/
HTMLAPI.adjustSubFileHeadings("Arial","8pt");

結果の5250画面は、以下のように表示されます。

e,	2164			
e,	2164			
e,	2164			
	2164			
	2164			
01 🕀				
Comment				
uirement	D			
uirement	P			
uirement	P			
-	auirement			

SHARED.ApplyStandardLayoutで以下の変更を行います。

/\* Adjust the subfile headings \*/ HTMLAPI.adjustSubFileHeadings("Arial","8pt","orange",'left',"1px solid bla

これにより、5250サブファイルは以下のようになります。

Employee Number		A1004				
Employee Surname		SMITHSON				
Employee Given Name(	(s)	PAUL				
Street No and Name		41 William Rd.				
Suburb or Town		St Ives				
State and Country		NSW	2144			
Home Phone Number		419 5656				
Department Code		AUD 🔎				
Section Code		03 🗩				
Start Date (DDMMYY)		1/05/80				
Termination Date (DD	MMYY)	0/00/00				
Date Skl Skill	Skill					
Acquired Code	Description	Comment	Grade			
25/03/98 ADMIN1	Administratn Part 1	Met requir	ement D			
5/02/98 ADMIN2	Administratn Part 2	Met requir	ement P			
4/05/98 COM	Communications Degre		D			
5/05/98 CS	Computer Science Deg		P			
			<b>x x</b>			

注:サブファイルに組み込まれている意図は、サブファイルの外観をいく らかリスト・ビューまたはグリッドのようにすることですが、上のオレ ンジ色の例では実現されていません。

### フォントを変更する

最後に、SHAREDレイアウト・ファンクションの変更サンプルを示しま す。ここでは、フォームのすべてのフィールドのフォントを8pt Arialに 変更します。SHARED.ApplyStandardLayoutで以前使用したサンプル・ ループは、以下のように変更することができます。

```
/* Get all elements between lines 3 and 22 */
var aH = HTMLAPI.getElementsinRowRange(3,22);
/* Hide all fields containing "+" signs and strip trailing dots from others */
for (i = 0; i < aH.length; i++)
 var oH = aH[i];
 if ((oH != null) && (typeof(oH.tagName) != "undefined"))
 {
  var fIsINPUTField = ((oH.tagName == "INPUT") || (oH.tagName == "TEXT
  if (fIsINPUTField) HTMLAPI.applyFont(oH,"Arial","7pt");
  else
  {
    if (oH.innerText == "+") HTMLAPI.hideElement(oH);
    else
    {
      HTMLAPI.stripTrailingDots(oH);
      HTMLAPI.applyFont(oH, "Arial", "8pt");
    }
  }
 }
```

結果の5250画面は、以下のように表示されます。

Employee Number	A1004	
Employee Surname	SMITHSON	
Employee Given Name(s)	PAUL	
Street No and Name	41 William Rd.	
Suburb or Town	St Ives	
State and Country	NSW	2144
Home Phone Number	419 5656	
Department Code	AUD 🗩	
Section Code	03 <i>J</i> D	
Start Date (DDMMYY)	1/05/80	
Termination Date (DDMMYY)	0/00/00	

Date Skl	Skill	Skill		
Acquired	Code	Description	Comment	Grade
25/03/98	ADMIN1	Administratn Part 1	Met requirement	D
5/02/98	ADMIN2	Administratn Part 2	Met requirement	P
4/05/98	COM	Communications Degre		D
5/05/98	CS	Computer Science Deg		P
				<b>X X</b>

### 以前は、以下のように表示されていました。

English Degree

4/05/98

ENG



D

+

### イメージを追加する

RAMPコマンド・ハンドラーにイメージを追加するには、以下のサンプ ル汎用コードをSHAREDオブジェクトに追加します。

```
var SHARED =
{
 /* ______*/
 /* Handle clicking on a prompt image
                                            */
 /* ______ */
 oFloatingImage : null,
 InsertImage : function(sBesideField,sSource,iHeight,iWidth,iHOffset,iVOffset
 {
   var oE = HTMLAPI.getElementbyName(sBesideField);
   if (oE == null) return;
   var oC = HTMLAPI.getcontainerDIV(oE);
   if (oC == null) return;
   if (this.oFloatingImage == null)
   {
     this.oFloatingImage = oC.ownerDocument.createElement("
<IMG style='position:absolute; visibility:hidden; display:none;' >");
     oC.ownerDocument.body.insertAdjacentElement("beforeEnd",this.oFloati
   }
   this.oFloatingImage.src = sSource;
   this.oFloatingImage.style.pixelTop = oC.style.pixelTop + iVOffset;
   this.oFloatingImage.style.pixelLeft = oC.style.pixelLeft + oC.style.pixelWi
   this.oFloatingImage.style.pixelHeight = iHeight;
   this.oFloatingImage.style.pixelWidth = iWidth;
   this.oFloatingImage.style.visibility = "visible";
   this.oFloatingImage.style.display = "inline";
```

### また、新しい行を標準のレイアウト・ファンクションに追加して、新し い画面が到着する際にイメージが消去されるようにします。

ApplyStandardLayout : function(aPromptFields) {

/\* Drop any floating images left around from before \*/

if (this.oFloatingImage != null) { this.oFloatingImage.style.visibility = "hidd

以上で、独自の完全に汎用的なInsertImage機能をRAMP-TS 5250画面に追加しました。以下の行を画面の到着スクリプトに追加して試します。

SHARED.InsertImage("SURNAME","/ts/skins/images/TestImage1.gif",123,100

### 以下の結果が得られます。

mployee Nu	umber		A1004		1944	
mployee Su	rname		SMITHSON			
mployee Gi	ven Name(s)		PAUL			( And A
Street No an	d Name		41 William F	łd.		In a Maria
uburb or To	wn		St Ives			25 9
state and Co	untry		NSW		2144	And the second second
lome Phone	Number		419 5656			
epartment (	Code		AUD 🏓			
ection Code			03 🕀		1	
tart Date (D	DMMYY)		1/05/80	))		
ermination I	Date (DDMMY	Y)	0/00/00	(+)		
Date Skl Acquired	Skill Code	Skill Description		Comment	Grade	
25/03/98	ADMIN1	Administratn Part 1		Met requirement	D	
5/02/98	ADMIN2	Administratn Part 2		Met requirement	P	
4/05/98	COM	Communications Degre			D	
E/0E/00	CS	Computer Science Deg			P	

SHARED.InsertImage()は、アプリケーションの任意の場所で再利用する ことができます。Axesフォルダの新しいファイルに対する権限を正しく 設定することを忘れないでください。

また、イメージ用の新しいコマンド・ハンドラー・タブを追加すること は、迅速で簡単なソリューションであることも覚えておいてください。

#### 注意事項

注意事項は、以下のとおりです。

- DOM構造への低レベルのアクセスに本質的に伴うリスクを理解し、受け入れる(詳細については、前のセクションを参照してください)。
- アプリケーションのパフォーマンスに影響を与えていないことを確認する。このような確認は簡単に行うことができます。ロジックをコメント・アウトするか、無効にします。代表的なエンド・ユーザーのPCのようなパフォーマンス・プロファイルを備えているPCを使用して、タイミング・テストを実行します。次に、ロジックを有効にしてテストを繰り返します。
- 5250画面に重点を置きすぎないこと。例えば、顧客照会5250画面が あるとします。Googleマップの画像をアプリケーションに追加して、 顧客の場所を表示したいとします。これには、次の2つの方法があり ます。
  - 必要なJavaScriptロジックを使用して、IFRAMEを5250画面に追加 する。
  - "マップ・ロケーション"というタブをVLFアプリケーションに追加する。

もちろん、実装するには後者の方がシンプルで簡単です。これはオンデ マンドでもあり、多くの場合にほとんどの人が確認する必要のない情報 で画面がいっぱいになることはありません。

### HTMLAPIファンクションで提供される内容

### HTMLAPI.getElementbyName

パラメータ:

記述	タイプ	任意/必須	デフォルト値
エレメント名	文字列	必須	
インデックス	整数	任意	0

戻り値:HTMLエレメントへの参照。見つからなかった場合はNULL。

### HTMLAPI.getElementbyLocation

パラメータ:

記述	タイプ	任意/必須	デフォルト値
行	整数	必須	
列	整数	必須	

戻り値:HTMLエレメントへの参照。見つからなかった場合はNULL。

# HTMLAPI.getElementsinRowRange

### パラメータ:

記述	タイプ	任意/必須	デフォルト値
低い行	整数	必須	
高い行	整数	任意	低い行

戻り値:HTMLエレメントへの参照の配列。

# HTMLAPI.getElementsinColumnRange

記述	タイプ	任意/必須	デフォルト値
低い列	整数	必須	
高い列	整数	任意	低い列

戻り値:HTMLエレメントへの参照の配列。

### HTMLAPI.showElement

パラメータ:

記述	タイプ	任意/必須	デフォルト値
HTMLエレメ ント	HTMLエレ メントの参 照	必須	

戻り値:NULL

### HTMLAPI.hideElement

パラメータ:

記述	タイプ	任意/必須	デフォルト値
HTMLエレメ ント	HTMLエレ メントの参 照	必須	

戻り値:NULL

### HTMLAPI.hideRow

### パラメータ:

記述	タイプ	任意/必須	デフォルト値
行番号	整数	必須	

戻り値:NULL

### HTMLAPI.insertImage

記述	タイプ	任意/必須	デフォルト値
イメージをそ の後に挿入す るHTMLエレ メント	HTMLエレ メントの参 照	必須	
イメージ・	文字列	必須	

ソース			
クリック・イ ベント・ハン ドラー	ファンク ション	任意	NULL
イメージの高 さ	整数	任意	イメージ自体 の高さ
イメージの幅	整数	任意	イメージ自体 の幅
水平オフセッ ト	整数	任意	2
垂直オフセッ ト	整数	任意	0

戻り値:作成されたHTMLエレメントへの参照。エラー状況エレメント ではNULL。

### HTMLAPI.applyFont

パラメータ:

記述	タイプ	任意/必須	デフォルト値
フォントを適 用するHTML エレメント。	HTMLエレ メントの参 照	必須	
フォント・ ファミリー	文字列	必須	
フォント・サ イズ	文字列	必須	

戻り値:NULL

### HTMLAPI.getcontainerDIV

記述	タイプ	任意/必須	デフォルト値

絶対位置決め コンテナーDIV を見つける HTMLエレメ	HTMLエレ メントの参 照	必須	
ント			

戻り値:絶対位置決めコンテナーDIVへの参照。見つからなかった場合はNULL。

### HTMLAPI.stripTrailingDots

パラメータ:

記述	タイプ	任意/必須	デフォルト値
ドットが除去 されるHTML エレメント	HTMLエレ メントの参 照	必須	

戻り値:NULL

### HTMLAPI.insertSubFileScrollers

記述	タイプ	任意/必須	デフォルト値
使用するPage Up イメージ	文字列	任意	NULL
Page Upクリッ ク・ハンドラー	ファン クショ ン	任意	NULL
使用するPage Downイメージ	文字列	任意	NULL
Page Downクリッ ク・ハンドラー	ファン クショ ン	任意	NULL
イメージの高さ	整数	任意	12
イメージの幅	整数	任意	12

水平オフセット	整数	任意	0
垂直オフセット	整数	任意	0

戻り値:NULL

# HTMLAPI. adjustSubFileHeadings

パラメータ:

記述	タイプ	任意/必須	デフォルト値
使用するフォント	文字列	任意	Lucida Console
フォント・サイズ	文字列	任意	8pt
背景色	文字列	任意	Buttonface
テキストの配置	文字列	任意	center
境界線のスタイル	文字列	任意	1px solid darkgray

戻り値:NULL

画面ラッパー

RAMP画面ラッパーは、シーン背後の5250画面にアクセスする、Visual LANSAコンポーネントです。アクセスされる画面およびフィールドは、コレオグラファーによる通常の方法で定義されます。

画面ラッパーは、5250画面から値を取り出し、まったく異なる方法で ユーザーにその値を提供します。同様に、画面ラッパーはユーザーから の入力を受け付け、入力内容を5250画面にマッピングして、5250トラン ザクションが行われるようにします。



5250画面ラッパーを使用するとき 画面ラッパーの基本 イベント メソッド 使用例

#### 5250画面ラッパーを使用するとき

画面ラッパーの主な利点は、はっきりしています。5250画面に組み込ま れたすべてのビジネス・ロジックを分析、書き換え、リセットし、VLコ ンポーネントと置き換えるときに、時間と費用を要することなく、見栄 えを良く、使いやすくし、高度なGUIの外観を持たせることができま す。

これは特に、Systemi唯一のソリューションに満足しているためにプラットフォームの移植性には現実的な興味を持たないユーザーにとっては重要です。

#### 使用例

使用例には次のものがあります。

- 画面ラッパーは、非表示の5250画面から値を取得し、まったく異なる方法でその値を表示します。たとえば、統計情報を抽出し、一連の棒グラフで表示できます(「例3:Systemiディスク使用量の表示」を参照)。
- 画面ラッパーはユーザー入力を受け付け、入力内容を5250画面に マッピングして、5250トランザクションが行われるようにします。た とえば、VLコンポーネントは高度な機能、高ボリュームのオーダー 入力を許可できます。ユーザーが保存をクリックすると、オーダーの 詳細が一連の5250画面および入力にマップされます。
- 画面ラッパーは、1回のクリックで多くの5250画面を実行できます。たとえば、画面ラッパーは20のオーダー番号のリストを表示できます。ユーザーがOKをクリックすると、1回に1つのオーダーを削除することしかできない5250画面を繰り返し実行し、20オーダーすべてが削除されます。

#### 刷新プロジェクトでの役割

System iに依存したソリューションで顧客を満足させるという点で、画面 ラッパーを使用してアプリケーションを刷新することはほとんど考えら れていませんでした。

現実的には、画面ラッパーの開発には時間も費用もかかりますが、特に 刷新プロジェクトのアプリケーション・テスト段階では、同等のVLコン ポーネントの開発にかかる時間と費用よりもはるかに少ないものです。 画面ラッパーは使い物にならないわけではありません。時間と費用が許 せば、5250への依存性を排除して適切なVLコンポーネントに変更できま す。

画面ラッパーとVLコンポーネント、これらの5250画面を併用することで、市場において刷新プロジェクトを推進することになるでしょう。たとえば、この併用はISVにとって適切です。

- 85%-いち早く市場に投入するための5250画面
- 10% -画面ラッパー-頻繁に使用される重要な領域(オーダー入力など)を、多くのビジネス価値を付加する、はるかに使いやすいものとすばやく入れ替える
- 5% -VLコンポーネント-高エンド価値をアプリケーション(電子 メール、PDF文書、MS Excelスプレッドシート、Web統合など)に追加 する

画面ラッパーの基本

画面ラッパーの定義

画面ラッパーは、クラスVF\_SY122のVL再利用可能パーツです。内部の あらゆるタイプのルーチンとは対照的に、グローバルなスコープとして 定義する必要があります。

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#PANL\_1) Visible(False) Displayposition(3)

キー・ポイント:

- 初期表示をFalseに設定します。これにより、指定しない限り表示されなくなります。たとえば、致命的エラーが発生したときにデザイン・モードで表示すれば、現在の5250画面の状態を確認するオプションを選択できます。
- メイン・パネルの中央に配置されたパネルの子にすることができます。これにより、致命的エラーを追跡するために表示することが簡単になります。

### uCommandプロパティの設定

コマンドのuInitializeメソッド・ルーチンで、画面ラッパーのuCommandプロパティを設定します。

Mthroutine Name(uInitialize) Options(\*REDEFINE) \* Do any initialization defined in the ancestor Invoke Method(#Com\_Ancestor.uInitialize) Set Com(#myscreen\_wrapper) Ucommand(#com\_owner) Endroutine

キー・ポイント:

- uCommandは常に#com\_ownerに設定します。
- uCommandの設定に失敗すると、タイプVF\_INIT\_ERRORのエラー・ メッセージが表示されます。

### RAMPを使用できるようにして実行を開始

通常、初めて実行するときは、コマンドの*uExecute*メソッド内部で MakerampTSavailableメソッドを呼び出します。

#myscreen\_wrapper.MakerampTSavailable

キー・ポイント:

• 初回コマンド実行時に初めてRAMPを使用可能にする場合、RAMP

がホストに接続されていないため、イベントの起動には通常より少し 時間がかかります。

 RampTSAvailableイベント・ルーチンでコマンドが制御を取り戻し ます。

### RampTSAvailableイベントの検査

RAMPが接続され、対話処理の準備ができると、このイベントのコマンドにシグナルを戻します。つまり、ナビゲーションを開始する準備が整っています。例:

Invoke Method(#myscreen\_wrapper.navigatetoscreen) Name(EMPLOYEE\_SKILLS)

### RampMessageイベントの検査

RampMessageイベントではエラー処理ロジックを記述し、5250アプリ ケーションで発生したメッセージを処理します。

RampMessageイベント RampTSAvailableイベント vHandleArriveイベント

# RampMessageイベント

RAMPまたはベースとなる5250アプリケーションにより、メッセージが 発行されます。

パラメータ

uMessageType	Char 256	下の表のようなメッセージのタイ プを指定する文字列
uMessageText	Char 132	メッセージのテキストを含む文字 列

# この表は、使用可能なメッセージ・タイプとその原因を示しています。

タイプ	原因	説明
VF_ERROR	致命的なエラー。	何らかの理由で、要求の 実行の処理に失敗しまし た。 たとえば、失敗したナビ ゲーション要求などが原 因です。
VF_INFO	5250アプリケーショ ンからのメッセー ジ。	背後で実行されている実 際の5250プログラムに よって送信されるメッ セージ。 たとえば、失敗した妥当 性検査などです。
VF_INIT_ERROR	初期化に失敗した画 面ラッパー。	これは通常、指定された セッション・ユーザーの オブジェクト・タイプに より定義済みセッション が提供されないときに発 生します。 または、uCommandプロ パティを設定していない

		ときに発生します(「画 面ラッパーの基本」を参 照)。
VF_UNKNOWN_FORM	ナビゲーション中に 未定義のフォームが 見つかりました。	

### 備考

異なるタイプのエラーを処理する方法は、開発者にかかっています。 メッセージを自動的にポップアップで表示するに

は、#com\_owner.avshowmessagesメソッドを使用します。致命的なエラーが発生した場合にベースとなる5250画面を表示すると、開発時に役立ちます。このように表示するには、画面ラッパーの表示および表示位置を変更します。

例

Evtroutine Handling(#screen wrapper.uRampMessage)

Umessagetype(#MsgType) Umessagetext(#MsgText)

Case (#msgtype.value)

When Value\_Is('= VF\_ERROR')

\* Optional.In design mode, making the screen wrapper visible allows you to show the 5250 screen.Set Com(#myscreen\_wrapper) Visible(True)

When Value\_Is('= VF\_INFO')

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value) When Value\_Is('= VF\_UNKNOWN\_FORM')

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value) When Value\_Is('= VF\_INIT\_ERROR')

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value) Endcase

Endroutine

# RampTSAvailableイベント RAMPは対話中であることを通知します。 パラメータ なし 備考 ここでナビゲーションを開始します。 例 Evtroutine Handling(#myscreen\_wrapper.RampTSAvailable) Invoke Method(#myscreen\_wrapper.navigatetoscreen) Name(EMPLOYEE\_SKILLS)

Endroutine

「MakerampTSavailableメソッド」も参照してください。

# vHandleArriveイベント

### 画面が到着しました。

ArrivedScreen	Char 256	到着した画面の名前を指定する文 字列。
PreviousScreen	Char 256	前の画面の名前を含む文字列。
ArrivedPayload	Char 256	ペイロードを含む文字列。
ForAction	Char 256	NavigateToScreenメソッドにパラ メータとして渡された文字列。

### 備考

このイベントは移動したどの画面でも通知されます。

例

Evtroutine Handling(#myscreen\_wrapper.vHandleArrive) Arrivedscreen(#ArrivedScreen) Previousscreen(#PreviousScreen) Arrivedpayload(#Payload) Case (#ArrivedScreen) When Value\_Is(= EMPLOYEE\_DETAILS) Set Com(#SAVE\_BUTTON) Enabled(false) If (#Payload \*NE UPDATE\_EMPLOYEE) #myscreen\_wrapper.getvalue From('empno') Value(#empno.value) #myscreen\_wrapper.getvalue From('surname') Value(#surname.value) Endif Endcase Endroutine メソッド

画面ラッパーは、コンポーネントVF\_SY122で指定される通常のVLコード・メソッドを使用して5250画面を動かします(対応するRAMP JavaScript ファンクションが機能するしくみと非常によく似ています)。

MakerampTSavailableメソッド NavigateToScreenメソッド SetValueメソッド GetValueメソッド SendKeyメソッド Current\_Formメソッド SetCursorメソッド SetCursorToFieldメソッド Get\_Form\_Messageメソッド Check Field Existsメソッド

### MakerampTSavailableメソッド

RAMPを対話処理できるようにします。

# 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.MakerampTSavailable

# パラメータ

uUserObjectType	Char 32 -任意	RAMPセッションの ユーザー・オブジェク ト・タイプを含む文字 列。 複数のセッションを 扱っているときに必要 です。
uSession_Id	Char 40 -任意	Destinationに割り当てら れたセッション。デ フォルトは*AUTOで す。

### 戻り値

なし

### 備考

uExecuteコマンド・ハンドラー・メソッドでこのメソッドを呼び出しま す。これによるパフォーマンス上のペナルティはありません。また、こ れによりRAMPTSとの対話処理を確実に実行できます。

### 使用例

Invoke Method(#myscreen\_wrapper.MakerampTSavailable) uSession\_Id(SESSION\_A)

Invoke Method(#myscreen\_wrapper.MakerampTSavailable)
uUserObjectType(HumanResources)

関連トピック「MakerampTSavailableメソッド」

### NavigateToScreenメソッド

RAMP-TS画面に移動します。

# 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.NavigateToScreen Name(EMPLOYEE\_SKILLS)

# パラメータ

Name	Char 256 - 必須	移動先の画面の名前を含む文字 列。
------	---------------	----------------------

### 戻り値

なし

## 備考

NavigateToScreenを実行すると、画面ラッパーはvHandleArriveイベント・ ルーチンで処理される画面到着イベント通知を受け取ります。

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER) #myscreen\_wrapper.Unavigatetoscreen) Name(EMPLOYEE\_SKILLS)
### SetValueメソッド

5250画面の入力フィールドの値を設定します。インデックスを渡して、 サブファイルの入力フィールドの値を設定します。

### 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper)
Parent(#COM\_OWNER)

Setting by Name - #myscreen\_wrapper.setvalue Infield(sFieldName) Value(vValue)

# パラメータ

名前による設定:

InField	Char 256 - 必須	入力フィールドの名前を含 む文字列。
Value	Variant -必須	値を含む文字列または数 値。
Index	Integer - 任意	フィールドのサブファイル 行を指定する整数。 注:指定した行インデックス が現在のサブファイル・ ページに存在する必要があ ります。

### 戻り値

なし

#### 備考

値を設定できるのは、名前を付けられている入力フィールドのみです。

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.setvalue Infield(GIVENAME) Value(#Givename)

\* Set the value of a field in the 3<sup>rd</sup> row of the current subfile page #myscreen\_wrapper.setvalue Infield(SFL\_OPTION) Value(#SelOption) Index(3)

## GetValueメソッド

パラメータ

## 画面またはサブファイルのフィールドから値を取得します。 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.getvalue From(sField) Value(sValue)

<b>ر</b>			
	From	Char 256 - 必須	値を取得するフィールド の名前を含む文字列。
	DefaultValue	Char 256 -任意	フィールドが見つからな かった場合に返すデフォ ルト値を含む文字列。
	Index	Integer - 任意	フィールドのサブファイ ル行を指定する整数。 注:指定した行インデック スが現在のサブファイ ル・ページに存在する必 要があります。

# 戻り値

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper)
Parent(#COM\_OWNER)

#myscreen\_wrapper.getvalue From(SURNAME) Value(#surname.value)
#myscreen\_wrapper.getvalue From(SURNAME) Value(#surname.value)
Index(5) DefaultValue(\*Blanks)

### SendKeyメソッド

# ファンクション・キーを押す操作をエミュレートします。 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.SendKey Key(#myscreen\_wrapper.<key property>)

J	<b>パラメー</b>	・タ	
	+-	Property -必須	希望のキーにする #myscreen_wrapperのプロパ ティ。 これらのプロパティのリス トについては、lansa049.chm にある「SENDKEYファンク ションのファンクション・ キー名」のSENDKEY名を参 照してください。
	Payload	Char 256 -任意	ペイロードを含む文字列。 ペイロードはvHandleArrive で返されます。

# 戻り値

なし

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER) #myscreen\_wrapper.Sendkey Key(#myscreen\_wrapper.KeyPageDown) Payload(NEXT\_PAGE)

### Current\_Formメソッド

### 現在の画面ラッパー画面のフォーム名を取得します。

# 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.current\_form Name(sName)

# パラメータ

なし

### 戻り値

Name Char 25	6 - 必須 現在の5250画面ラッパー画面の を含む文字列	の名前
--------------	-----------------------------------	-----

# 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.current\_form Name(#std\_txtl)

### SetCursorメソッド

### 画面の指定行および列にあるカーソルの位置。

# 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.SetCursor RowNum(iRowNum) ColNum(iColNum)

# パラメータ

RowNum	Integer - 必須	カーソルを配置する位置の 行番号を指定する整数。
ColNum	Integer - 任意	任意。カーソルを配置する 位置の列番号を指定する整 数。 デフォルトは1です。

### 戻り値

なし

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.setcursor Rownum(10)

### SetCursorToFieldメソッド

#### 画面またはサブファイルの指定されたフィールドにカーソルを置きま す。

## 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.SetCursorToField Name(SURNAME)

# パラメータ

Name	Char 256 - 必須	カーソルを配置するフィー ルドの名前を指定する文字 列。
Index	Integer - 任意	フィールドのサブファイル 行を指定する整数。 注:指定した行インデック スが現在のサブファイル・ ページに存在する必要があ ります。

### 戻り値

なし

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#myscreen\_wrapper.setcursortofield Name(SURNAME) Index(3)

### Get\_Form\_Messageメソッド

# 指定された画面行番号のメッセージを取得し、ルーティングします。 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#bMoreMsgs := #myscreen\_wrapper.GetFormMessage(iRow)

# パラメータ

RowNumber Integer - 必須	メッセージを取得する 5250画面の行番号を指定 する整数。
------------------------	--------------------------------------

# 戻り値

MoreMessages	Boolean	サブファイルのメッセー ジ:
		true – さらにメッセージ があります。
		false – これ以上メッセー ジはありません。

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#MoreMsgs := #myscreen\_wrapper.Get\_Form\_Messate RowNumber(22)

### Check\_Field\_Existsメソッド

### フィールドが現在の画面またはサブファイルに存在するかどうかを確認 します。

### 構文

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM\_OWNER)

#bFound := #myscreen\_wrapper.Check\_Field\_Exists(sName iInd)

パラメー	タ
------	---

Name	Char 256 - 必須	カーソルを配置する位置の 行番号を指定する整数。
Index	Integer - 任意	フィールドのサブファイル 行を指定する整数。 注:指定した行インデック スが現在のサブファイル・ ページに存在する必要があ ります。

### 戻り値

Found Boolean	true – 現在の画面でフィー ルドが見つかりました。 false – フィールドは見つか りませんでした。
---------------	------------------------------------------------------------------

### 使用例

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Parent(#COM OWNER)

#FldFound := #myscreen\_wrapper.Check\_Field\_Exists("SURNAME")

#FldFound := #myscreen\_wrapper.Check\_Field\_Exists("SURNAME"
#listcount)

Dowhile (#myscreen\_wrapper.check\_field\_exists( "SKILLCODE" #listcount )) #myscreen\_wrapper.getvalue From("skillcode") Value(#skilcode) Index(#listcount)

#listcount += 1

Endwhile

## 使用例

- 例1:従業員の詳細の表示
- 例2:従業員の詳細と資格の表示
- 例3: System iのディスク使用量の表示

### 例1:従業員の詳細の表示

#### この例では、人事管理システムの一部である資格情報の検索 / 保守画面 にナビゲートします。

😸 Basic details 🛛 🛷 Appro	ove 🗌 🗖 Details 🛛 🧇 Example 8 🦳 🦫 Skills 🛛 🥐 Transfer 🛛 🎅 Email 🖉 🖙 Video 🦷	🔍 All Details
Employee Number	A0090	Save
Employee Surname	BLOGG	0
Employee Given Name(s)	FRED JOHN HENRY	
Street No and Name	72 MAIN STREET	
Suburb or Town	NEWTOWN HEIGHTS	
State and Country	AUSTRALIA	
Post / Zip Code	2202	
Home Phone Number	344-2345	
この画面に移動す	するために、RAMPスクリプトは次のステップを	実行し

• サイン・オンする。

ます。

- コマンド行にlansa run pslsys partition(dem)と入力し、Enterキーを押す。
- オプションフィールドに3と入力し、Enterキーを押す。
- 現在選択している従業員の従業員番号を入力し、Enterキーを押す。
- F21キーを押す。

Function Options(\*DIRECT) Begin\_Com Role(\*EXTENDS #VF\_AC010) Height(569) Layoutmanager(#MAIN\_LAYOUT) Width(776) \* Group By Name(#XG HEAD) Fields(#EMPNO #SURNAME #GIVENAME #ADDRESS1 #ADDRESS2 #ADDRESS3 #POSTCODE #PHONEHME **#DEPTMENT #SECTION**) \* Body and Button arrangement panels Define\_Com Class(#PRIM\_PANL) Name(#BUTTON\_PANEL) Displayposition(2) Height(569) Hint(\*MTXTDF\_DET1) Layoutmanager(#BUTTON\_FLOW) Left(688) Parent(#COM\_OWNER) Tabposition(3) Tabstop(False) Top(0) Width(88) Define\_Com Class(#PRIM\_PANL) Name(#BODY\_HEAD) Displayposition(1) Height(569) Hint(\*MTXTDF DET1) Layoutmanager(#BODY HEAD FLOW) Left(0) Parent(#COM OWNER) Tabposition(2) Tabstop(False) Top(0) Verticalscroll(True) Width(688) \* Attachment and flow layout managers Define\_Com Class(#PRIM\_ATLM) Name(#MAIN\_LAYOUT) Define\_Com Class(#PRIM\_FWLM) Name(#BUTTON\_FLOW) Direction(TopToBottom) Flowoperation(Center) Marginbottom(4) Marginleft(4) Marginright(4) Margintop(4) Spacing(4) Spacingitems(4) Define\_Com Class(#PRIM\_FWLM) Name(#BODY\_HEAD\_FLOW) Direction(TopToBottom) Marginbottom(4) Marginleft(4) Marginright(4) Margintop(4) Spacing(4) Spacingitems(4) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_EMPNO) Manage(#EMPNO) Parent(#BODY HEAD FLOW) Define Com Class(#PRIM FWLI) Name(#FWLI SURNAME) Manage(#SURNAME) Parent(#BODY HEAD FLOW) Define Com Class(#PRIM FWLI) Name(#FWLI GIVENAME) Manage(#GIVENAME) Parent(#BODY HEAD FLOW) Define Com Class(#PRIM FWLI) Name(#FWLI ADDRESS1) Manage(#ADDRESS1) Parent(#BODY HEAD FLOW) Define Com Class(#PRIM FWLI) Name(#FWLI ADDRESS2) Manage(#ADDRESS2) Parent(#BODY HEAD FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_ADDRESS3) Manage(#ADDRESS3) Parent(#BODY HEAD FLOW) Define Com Class(#PRIM FWLI) Name(#FWLI POSTCODE) Manage(#POSTCODE) Parent(#BODY HEAD FLOW) Define Com Class(#PRIM FWLI) Name(#FWLI PHONEHME)

Manage(#PHONEHME) Parent(#BODY HEAD FLOW) Define Com Class(#PRIM FWLI) Name(#FWLI SAVE BUTTON) Manage(#SAVE BUTTON) Parent(#BUTTON FLOW) \* The save button Define\_Com Class(#PRIM\_PHBN) Name(#SAVE\_BUTTON) Caption(\*MTXTDF\_SAVE) Displayposition(1) Left(4) Parent(#BUTTON\_PANEL) Tabposition(1) Top(4) \* Collection for detail fields Define\_Com Class(#Prim\_ACol<#prim\_evef>) Name(#PanelFields) \* Fields in the head area Define\_Com Class(#EMPNO.Visual) Displayposition(1) Height(19) Hint(\*MTXTDF\_DET1) Left(4) Parent(#BODY\_HEAD) Readonly(True) Tabposition(1) Top(4) Usepicklist(False) Width(209) Define Com Class(#SURNAME.Visual) Displayposition(2) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(2) Top(27) Usepicklist(False) Width(324) Define\_Com Class(#GIVENAME.Visual) Displayposition(3) Height(19) Hint(\*MTXTDF\_DET1) Left(4) Parent(#BODY\_HEAD) Tabposition(3) Top(50) Usepicklist(False) Width(324) Define\_Com Class(#ADDRESS1.Visual) Displayposition(4) Height(19) Hint(\*MTXTDF\_DET1) Left(4) Parent(#BODY\_HEAD) Tabposition(4) Top(73) Usepicklist(False) Width(363) Define\_Com Class(#ADDRESS2.Visual) Displayposition(5) Height(19) Hint(\*MTXTDF\_DET1) Left(4) Parent(#BODY\_HEAD) Tabposition(5) Top(96) Usepicklist(False) Width(363) Define Com Class(#ADDRESS3.Visual) Displayposition(6) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(6) Top(119) Usepicklist(False) Width(363) Define Com Class(#POSTCODE.Visual) Displayposition(7) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(7) Top(142) Usepicklist(False) Width(216) Define Com Class(#PHONEHME.Visual) Displayposition(8) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(8) Top(165) Usepicklist(False) Width(286) Define\_Com Class(#PRIM\_ATLM) Name(#ATLM 1) Define Com Class(#PRIM ATLI) Name(#ATLI 1) Attachment(Center) Parent(#ATLM 1) Define Com Class(#PRIM ATLI) Name(#ATLI 2) Attachment(Center) Manage(#BODY\_HEAD) Parent(#MAIN LAYOUT)

```
Define_Com Class(#PRIM_ATLI) Name(#ATLI_3) Attachment(Right)
Manage(#BUTTON_PANEL) Parent(#MAIN_LAYOUT)
Define_Com Class(#vf_sy122) Name(#myscreen_wrapper) Displayposition(3)
Height(569) Parent(#COM OWNER) Width(688)
Define_Com Class(#PRIM_ATLI) Name(#ATLI_4) Attachment(Center)
Parent(#MAIN LAYOUT)
Define_Com Class(#PRIM_ATLI) Name(#ATLI_6) Attachment(Center)
Manage(#myscreen wrapper) Parent(#MAIN LAYOUT)
______
* To better understand this example you should be famililar with the shipped
Personnel System demo.
* We expect the following screens to appear as part of this navigation:
* Login -> type in user and password -> press Enter -> (a special screen?<F3>)
* i5 Main Menu -> type lansa run process(pslsys) partition(dem) -> press Enter
* Personnel System -> type option 3 -> press Enter
* Inquire -> type the current instance employee number -> press Enter
* Browse Employee Details and Skills in OUTPUT mode -> press F21
* Browse Employee Details and Skills in INPUT mode
* Handle Initialization
 _____
Mthroutine Name(uInitialize) Options(*REDEFINE)
Define Com Class(#Prim evef) Name(#FormField) Reference(*dynamic)
Invoke Method(#Com Ancestor.uInitialize)
For Each(#Control) In(#Body_Head.ComponentControls)
If Ref Com(#Control) Is(*INSTANCE OF #prim evef)
Set Ref Com(#FormField) To(*dynamic #Control)
Invoke Method(#PanelFields.Insert) Item(#FormField)
Endif
Endfor
* Set the uCommand wrapper property.
Set Com(#myscreen wrapper) Ucommand(#com owner)
Endroutine
```

\* Handle Command Execution \* Always invoke makerampTSavailable to ensure RAMP-TS is up and running before starting a navigation \* You may also disable the entire form to prevent any input while RAMP is navigating Mthroutine Name(uExecute) Options(\*REDEFINE) Invoke Method(#Com\_Ancestor.uExecute) #myscreen\_wrapper.makerampTSavailable Set Com(#Save\_Button) Enabled(False) #com owner.enabled := false Endroutine \_\_\_\_\_ \* Event Handlers \_\_\_\_\_ \* RAMP has signalled it's ready.Invoke your navigation here. \* Once the navigaton starts, processing resumes in the vHandleArrive event handler. Evtroutine Handling(#myscreen wrapper.RampTSAvailable) Invoke Method(#myscreen wrapper.navigatetoscreen) Name('updempskills') Endroutine \* The Payload is a 256 character string sent together with a SENDKEY. Use the Payload in the same way you would use parameters in an event. \* Assuming when 'updempskills' arrives we change some details and press Enter or click on the SAVE button, we expect one of these 2 screens to appear: \* If the SAVE was successfull, the INQUIRE screen appears. \* If the SAVE was NOT successfull, the update employee screen will reappear \* Setting the Payload we determine what caused the screen 'updempskills' to arrive.

\*

= "UPDATE EMPLOYEE":

\* case KeyEnter: \* SENDKEY( KeyEnter, "UPDATE\_EMPLOYEE" ); \* break: \* 2. In the ARRIVE script of INQUIRE, we test payload. If the Payload is UPDATE\_EMPLOYEE this is telling us we have just done an Update and we are most likely to want to go back to the same screen. \* if ( TOSTRING(oPayload) == "UPDATE\_EMPLOYEE") \* { \* NAVIGATE\_TO\_DESTINATION("updempskills"); \* } \* Evtroutine Handling(#myscreen\_wrapper.vHandleArrive) Arrivedscreen(#CurrentScreen) Previousscreen(#PreviousScreen) Arrivedpayload(#Payload) Case (#CurrentScreen) When Value\_Is(= 'updempskills') Set Com(#SAVE\_BUTTON) Enabled(false) \* Payloads are destroyed when the ARRIVE script finishes executing. Therefore, a payload of UPDATE\_EMPLOYEE would most likely mean there was a validation error. Otherwise - if INQUIRE had arrived - the Payload would have been destroyed. If (#Payload = UPDATE EMPLOYEE) Use Builtin(message box show) With Args(ok ok info "Validation Error" "Please correct any errors") Else \* Unlock the framework Set Com(#avFrameworkManager) Ulocked(false) #myscreen wrapper.getvalue From('empno') Value(#empno.value)

\* 1. In the Button script of updempskills for the Enter key, we attach a payload

#myscreen\_wrapper.getvalue From('surname') Value(#surname.value)

#myscreen\_wrapper.getvalue From('givename') Value(#givename.value)

#myscreen\_wrapper.getvalue From('address1') Value(#address1.value)

#myscreen\_wrapper.getvalue From('address2') Value(#address2.value)

#myscreen\_wrapper.getvalue From('address3') Value(#address3.value)

#myscreen\_wrapper.getvalue From('homephone') Value(#phonehme.value)

#myscreen_wrapper.getvalue From('postcode') Value(#POSTCODE.value) #com_owner.enabled := true
Endif
Endcase
Endroutine
*
* Handle Save *
Mthroutine Name(Save)
* Set the 5250 field values to the values from this panel
#myscreen_wrapper.setvalue Infield('surname') Value(#surname.value)
<pre>#myscreen_wrapper.setvalue Infield('givename') Value(#givename.value)</pre>
#myscreen_wrapper.setvalue Infield('address1') Value(#address1.value)
#myscreen_wrapper.setvalue Infield('address2') Value(#address2.value)
#myscreen_wrapper.setvalue Infield('address3') Value(#address3.value)
#myscreen_wrapper.setvalue Infield('homephone') Value(#phonehme.value)
<pre>#myscreen_wrapper.setvalue Infield('postcode') Value(#POSTCODE.value)</pre>
* Send the Enter key with the payload
#myscreen_wrapper.sendkey Key(#myscreen_wrapper.KeyEnter)
Payload(UPDATE_EMPLOYEE)
Endroutine
* Listen to messages from RAMP and the 5250 application
Evtroutine Handling(#myscreen_wrapper.RampMessage)
Umessagetype(#MsgType) Umessagetext(#MsgText)
Case (#msgtype.value)
When Value_Is('= VF_ERROR')
* Fatal messages reported by Ramp (e.g. Navigation request failed, etc). If in
design mode, show the underlying 5250 screen. Otherwise, make the error
Inessage appear in a message box on top of the command
Set Com(#mysereon_wrapper) Visible(True)
Floo
Message Megid(dcm9899) Megf(dc@m01) Megdta(#megtevt value)
#com_owner avshowmessages
Fndif
* Messages sent by the System i application or unknown form was encountered
When Value $Is('= VF INFO' '= VF IINKNOWN FORM')$
Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value)
* Failure to initialize RAMP.Could occur for mainly one of two reasons

When Value\_Is('= VF\_INIT\_ERROR') Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value) #com owner.avshowmessages Otherwise Use Builtin(message\_box\_show) With\_Args(ok ok info \*Component ('Unknown message type ' + #MsgType + 'encountered')) Endcase Endroutine \* \_\_\_\_\_ \* Handle changes in any of the fields on the panel \* \_\_\_\_\_ Evtroutine Handling(#PanelFields<>.Changed) \* Enable the save button Set Com(#SAVE\_BUTTON) Enabled(True) \* Lock the framework and set a message for the user Use Builtin(bconcat) With\_Args('Changes made to employee' #GiveName #Surname 'have not been saved yet."Do you want to save them before continuing?')To\_Get(#sysvar\$av) Set Com(#avFrameworkManager) Ulocked(USER) Ulockedmessage(#sysvar\$av) Endroutine \_\_\_\_\_ \_\_\_\_\_ \* Enter key pressed \* \_\_\_\_\_ Evtroutine Handling(#PanelFields<>.KeyPress) **Options(\*NOCLEARMESSAGES \*NOCLEARERRORS)** Keycode(#KeyCode) If Cond('#KeyCode.Value = Enter') \* If there no changes have been made issue message and ignore enter If Cond('#SAVE BUTTON.Enabled \*EQ True') Invoke Method(#Com Owner.Save) Else \* Issue 'There are no changes to save' message Use Builtin(Message box show) With Args(ok ok Info \*Component \*MTXTDF NO SAVE) Endif Endif Endroutine

\* Handle the save button
\* \_\_\_\_\_

Evtroutine Handling(#SAVE\_BUTTON.Click)

#com\_owner.Save

Endroutine

\* \_\_\_\_\_

\* Handle Termination

Mthroutine Name(uTerminate) Options(\*REDEFINE)

\* Clean up the colelction of fields on the panel

Invoke Method(#PanelFields.RemoveAll)

\* Do any termination defined in the ancestor

Invoke Method(#Com\_Ancestor.uTerminate)

Endroutine

End\_Com

### 例2:従業員の詳細と資格の表示

# この例では、前の例を拡張したものです。同じ詳細を表示していますが、Visual LANSAリストビューに資格も表示しています。

🔗 Basic details	Approv	ve 📃 Details 😒 Example 8	🖐 Skills 🛛 🦰 Transfer	Email	🖘 Video	All Details
Employee Numbe	er	A0090				Save
Employee Surname		BLOGG				
Employee Given Name(s)		FRED JOHN HENRY				
Street No and Name		72 MAIN STREET	2			
Suburb or Town		NEWTOWN HEIGHTS				
State and Count	ry	AUSTRALIA				
Post / Zip Code		2202				
Home Phone Nur	mber	344-2345				
Acquired	Skill Code	Skill	Comment	Gra		
21/01/87	CL	CL PROGRAMMING2	1	D		
0/00/01	COBOL	COBOL PROGRAMMING	MARKED IMPROVE	P		
7/07/88	ECD	ECONOMICS DEGREE	2	P		
0/00/01	INTRO	COMPANY INTRODUCTION	3	M		
1/01/89	MANAGE1	MANAGEMENT COURSE 1	4	P		
25/09/88	MARKET2	MARKETING COURSE 2		F		
10/06/89	MARKET3	MARKETING COURSE 3		P		
2/08/89	MBA	MASTER BUSINESS		P		
1/01/92	RPG	RPG PROGRAMMING	6	P		
L						

この例では、サブファイル/ブラウズリストにアクセスする方法がわかり ます。

Function Options(\*DIRECT)

Begin\_Com Role(\*EXTENDS #VF\_AC010) Height(569) Layoutmanager(#MAIN\_LAYOUT) Width(776)

\*

\* Simple Field and Group Definitions

\*

Group\_By Name(#XG\_HEAD) Fields(#EMPNO #SURNAME #GIVENAME

\_\_\_\_\_

#ADDRESS1 #ADDRESS2 #ADDRESS3 #POSTCODE #PHONEHME #DEPTMENT #SECTION)

\* Body and Button arrangement panels Define\_Com Class(#PRIM\_PANL) Name(#BUTTON\_PANEL) Displayposition(2) Height(569) Hint(\*MTXTDF\_DET1) Layoutmanager(#BUTTON\_FLOW) Left(688) Parent(#COM\_OWNER) Tabposition(3) Tabstop(False) Top(0) Width(88)

Define\_Com Class(#PRIM\_PANL) Name(#BODY\_HEAD) Displayposition(1) Height(569) Hint(\*MTXTDF\_DET1) Layoutmanager(#BODY\_HEAD\_FLOW) Left(0) Parent(#COM\_OWNER) Tabposition(2) Tabstop(False) Top(0) Verticalscroll(True) Width(688)

\* Attachment and flow layout managers Define\_Com Class(#PRIM\_ATLM) Name(#MAIN\_LAYOUT) Define\_Com Class(#PRIM\_FWLM) Name(#BUTTON\_FLOW) Direction(TopToBottom) Flowoperation(Center) Marginbottom(4) Marginleft(4) Marginright(4) Margintop(4) Spacing(4) Spacingitems(4)

Define\_Com Class(#PRIM\_FWLM) Name(#BODY\_HEAD\_FLOW) Direction(TopToBottom) Marginbottom(4) Marginleft(4) Marginright(4) Margintop(4) Spacing(4) Spacingitems(4)

Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_EMPNO) Manage(#EMPNO) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_SURNAME) Manage(#SURNAME) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_GIVENAME) Manage(#GIVENAME) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_ADDRESS1) Manage(#ADDRESS1) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_ADDRESS2) Manage(#ADDRESS2) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_ADDRESS3) Manage(#ADDRESS3) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_ADDRESS3) Manage(#ADDRESS3) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_POSTCODE) Manage(#POSTCODE) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_PHONEHME) Manage(#PHONEHME) Parent(#BODY\_HEAD\_FLOW) Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_SAVE\_BUTTON) Manage(#SAVE\_BUTTON) Parent(#BUTTON\_FLOW)

\* The save button

Define\_Com Class(#PRIM\_PHBN) Name(#SAVE\_BUTTON) Caption(\*MTXTDF\_SAVE) Displayposition(1) Left(4) Parent(#BUTTON\_PANEL) Tabposition(1) Top(4)

\* Collection for detail fields

Define\_Com Class(#Prim\_ACol<#prim\_evef>) Name(#PanelFields)

\* Fields in the head area

Define Com Class(#EMPNO.Visual) Displayposition(1) Height(19) Hint(\*MTXTDF\_DET1) Left(4) Parent(#BODY\_HEAD) Tabposition(1) Top(4) Usepicklist(False) Width(209) Define\_Com Class(#SURNAME.Visual) Displayposition(2) Height(19) Hint(\*MTXTDF\_DET1) Left(4) Parent(#BODY\_HEAD) Tabposition(2) Top(27) Usepicklist(False) Width(324) Define\_Com Class(#GIVENAME.Visual) Displayposition(3) Height(19) Hint(\*MTXTDF\_DET1) Left(4) Parent(#BODY\_HEAD) Tabposition(3) Top(50) Usepicklist(False) Width(324) Define Com Class(#ADDRESS1.Visual) Displayposition(4) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(4) Top(73) Usepicklist(False) Width(363) Define Com Class(#ADDRESS2.Visual) Displayposition(5) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(5) Top(96) Usepicklist(False) Width(363) Define Com Class(#ADDRESS3.Visual) Displayposition(6) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(6) Top(119) Usepicklist(False) Width(363) Define\_Com Class(#POSTCODE.Visual) Displayposition(7) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(7) Top(142) Usepicklist(False) Width(216) Define Com Class(#PHONEHME.Visual) Displayposition(8) Height(19) Hint(\*MTXTDF DET1) Left(4) Parent(#BODY HEAD) Tabposition(8)

Top(165) Usepicklist(False) Width(286)

Define\_Com Class(#PRIM\_ATLM) Name(#ATLM\_1) Define\_Com Class(#PRIM\_ATLI) Name(#ATLI\_1) Attachment(Center) Parent(#ATLM\_1) Define\_Com Class(#PRIM\_ATLI) Name(#ATLI\_2) Attachment(Center) Manage(#BODY\_HEAD) Parent(#MAIN\_LAYOUT) Define\_Com Class(#PRIM\_ATLI) Name(#ATLI\_3) Attachment(Right) Manage(#BUTTON\_PANEL) Parent(#MAIN\_LAYOUT)

Define\_Com Class(#PRIM\_ATLI) Name(#ATLI\_4) Attachment(Center) Parent(#MAIN\_LAYOUT) Define\_Com Class(#PRIM\_ATLI) Name(#ATLI\_6) Attachment(Center) Parent(#MAIN\_LAYOUT)

Define\_Com Class(#PRIM\_FWLI) Name(#FWLI\_1) Parent(#BODY\_HEAD\_FLOW)

Define\_Com Class(#PRIM\_LTVW) Name(#skills) Componentversion(2) Displayposition(9) Fullrowselect(True) Height(229) Left(4) Parent(#BODY\_HEAD) Showsortarrow(True) Tabposition(9) Top(188) Width(485) Define\_Com Class(#PRIM\_LVCL) Name(#LVCL\_1) Caption('Acquired') Captiontype(Caption) Displayposition(1) Parent(#skills) Source(#VF\_ELTXTS) Width(18) Widthtype(Fixed) Define Com Class(#PRIM LVCL) Name(#LVCL 2) Displayposition(2) Parent(#skills) Source(#SKILCODE) Width(17) Define Com Class(#PRIM LVCL) Name(#LVCL 3) Captiontype(ColumnHeadings) Displayposition(3) Parent(#skills) Source(#SKILDESC) Width(32) Define Com Class(#PRIM LVCL) Name(#LVCL 4) Captiontype(ColumnHeadings) Displayposition(4) Parent(#skills) Source(#COMMENT) Width(24) Define Com Class(#PRIM LVCL) Name(#LVCL 5) Displayposition(5) Parent(#skills) Source(#GRADE) Width(8) Widthtype(Characters)

Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Displayposition(3) Height(569) Parent(#COM\_OWNER) Width(688)

```
*
* To better understand this example you should be famililar with the shipped
Personnel System demo.
* We expect the following screens to appear as part of this navigation:
* Login -> type in user and password -> press Enter -> (a special screen? <F3>)
* i5 Main Menu -> type lansa run process(pslsys) partition(dem) -> press Enter
* Personnel System -> type option 3 -> press Enter
* Inquire -> type the current instance employee number -> press Enter
* Browse Employee Details and Skills in OUTPUT mode -> press F21
* Browse Employee Details and Skills in INPUT mode
* Handle Initialization
Mthroutine Name(uInitialize) Options(*REDEFINE)
Define_Com Class(#Prim_evef) Name(#FormField) Reference(*dynamic)
Invoke Method(#Com_Ancestor.uInitialize)
For Each(#Control) In(#Body Head.ComponentControls)
If Ref Com(#Control) Is(*INSTANCE OF #prim evef)
Set Ref Com(#FormField) To(*dynamic #Control)
Invoke Method(#PanelFields.Insert) Item(#FormField)
Endif
Endfor
* Set the uCommand wrapper property.
Set Com(#myscreen wrapper) Ucommand(#com owner)
```

#### Endroutine

-----\* Handle Command Execution

\*

\* Always invoke makerampTSavailable to ensure RAMP-TS is up and running before starting a navigation

\*

\* You may also disable the entire form to prevent any input while RAMP is navigating

۰ <u>-----</u>

\_\_\_\_\_

Mthroutine Name(uExecute) Options(\*REDEFINE)

Invoke Method(#Com\_Ancestor.uExecute)

Invoke Method(#myscreen\_wrapper.MakeRampTSAvailable)

Set Com(#Save\_Button) Enabled(False)

#com\_owner.enabled := false

Endroutine

\*

\*

\*

\* RAMP has signalled it's ready. Invoke your navigation here. Also, clear the list and panel fields.

\*

\* Once the navigaton starts, processing resumes in the vHandleArrive event handler.

\*

### Evtroutine Handling(#myscreen\_wrapper.RampTSAvailable)

```
Clr_List Named(#skills)
#XG_HEAD := *null
Invoke Method(#myscreen_wrapper.navigatetoscreen) Name('updempskills')
```

Endroutine

\*

\* The Payload is a 256 character string sent together with a SENDKEY. Use the Payload in the same way you would use parameters in an event.

\* Assuming when 'updempskills' arrives we change some details and press Enter or click on the SAVE button, we expect one of these 2 screens to appear: \* If the SAVE was successfull, the INQUIRE screen appears.

\* If the SAVE was NOT successfull, the update employee screen will reappear \* Setting the Payload we determine what caused the screen 'updempskills' to arrive.

\*

```
* 1. In the Button script of updempskills for the Enter key, we attach a payload = "UPDATE_EMPLOYEE":
```

\*

```
* case KeyEnter:
```

```
* SENDKEY( KeyEnter, "UPDATE_EMPLOYEE" )
```

\* break

\* 2. In the ARRIVE script of INQUIRE, we test payload. If the Payload is UPDATE\_EMPLOYEE this is telling us we have just done an Update and we are most likely to want to go back to the same screen.

```
* if ( TOSTRING(oPayload) == "UPDATE_EMPLOYEE")
```

```
* {
```

```
* NAVIGATE_TO_DESTINATION("updempskills")
```

\* }

\*

Evtroutine Handling(#myscreen\_wrapper.vHandleArrive) Arrivedscreen(#CurrentScreen) Previousscreen(#PreviousScreen) Arrivedpayload(#Payload) Define\_Com Class(#prim\_boln) Name(#MoreRecords)

Case (#CurrentScreen)

When Value\_Is(= 'updempskills')

Set Com(#SAVE\_BUTTON) Enabled(false)

\* Payloads are destroyed when the ARRIVE script finishes executing. Therefore, a payload of UPDATE\_EMPLOYEE would most likely mean there was a validation error. Otherwise - if INQUIRE had arrived - the Payload would have been destroyed. If (#Payload = UPDATE\_EMPLOYEE)

Use Builtin(message\_box\_show) With\_Args(ok ok info "Validation Error" "Please correct any errors")

Else \* Unlock the framework - no harm done if it wasn't locked Set Com(#avFrameworkManager) Ulocked(false)

\* Get the current instance and the values of the panel fields Invoke Method(#avListManager.GetCurrentInstance) Akey1(#deptment) Akey2(#section)

\* If there are no entries in the list or have sent a PageDown key, get the subfile page currently on the hidden 5250 If ((#skills.entries \*LE 0) Or (#Payload = NEXT\_PAGE))

#MoreRecords := #com\_owner.uGetSubfilePage

\* There are more records in teh subfile, send a pagedown. Processing will continue in this same event handler once the same screen with the next subfile page has arrived If (#MoreRecords)

#myscreen\_wrapper.sendkey Key(#myscreen\_wrapper.KeyPageDown)
Payload(NEXT\_PAGE)

\* When all the records have been added to the list view we can re enable the panel

Else

#com\_owner.enabled := true

#myscreen\_wrapper.getvalue From('empno') Value(#empno.value) #myscreen\_wrapper.getvalue From('surname') Value(#surname.value) #myscreen\_wrapper.getvalue From('givename') Value(#givename.value) #myscreen\_wrapper.getvalue From('address1') Value(#address1.value) #myscreen\_wrapper.getvalue From('address2') Value(#address2.value) #myscreen\_wrapper.getvalue From('address3') Value(#address3.value) #myscreen\_wrapper.getvalue From('homephone') Value(#phonehme.value) #myscreen\_wrapper.getvalue From('postcode') Value(#POSTCODE.value) Endif

Else

\* We can assume that there has been a successfull update so update the instance list.

Invoke Method(#avListManager.UpdateListEntryData) Akey1(#Deptment) Akey2(#Section) Akey3(#Empno.value) Visualid2(#surname + " " + #givename) Businessobjecttype(EMPLOYEE)

Endif

Endif

Endcase Set Com(#myscreen\_wrapper) Visible(False) Endroutine

\* Traverse the skills subfile/browselist

\* The technique used here consists of picking one field we know it's in the subfile, in this case "dateacq" and while there it is present on the screen, get all teh fields.

\* Fields in RAMP-TS subfiles are indexed starting from 1. A subfile page with 7 rows will have 7 instances of each of the fields in the subfile. Here we increase #listcount and use it to get the value of a field.

\* Note that you can specify a default value when using the getvalue method. Mthroutine Name(uGetSubfilePage)

Define\_Map For(\*result) Class(#prim\_boln) Name(#NextPage) Define Field(#MoreVal) Type(\*char) Length(1)

#listcount := 1 #NextPage := false

```
Dowhile (#myscreen_wrapper.check_field_exists( "dateacq" #listcount ))
```

```
#myscreen_wrapper.getvalue From("dateacq") Value(#vf_eltxts)
Defaultvalue(#ddmmyy) Index(#listcount)
#myscreen_wrapper.getvalue From("skillcode") Value(#skilcode)
Index(#listcount)
#myscreen_wrapper.getvalue From("skilldesc") Value(#skildesc)
Index(#listcount) Defaultvalue("Defalt value")
#myscreen_wrapper.getvalue From("comment") Value(#comment)
Index(#listcount)
#myscreen_wrapper.getvalue From("grade") Value(#grade) Index(#listcount)
```

```
* You can put some tracing
#com_owner.avframeworkmanager.avRecordTrace Component(#com_owner)
Event("Adding entry = " + #vf_eltxts + ", " + #skilcode + ", " + #skildesc + ", "
+ #grade)
```

```
Add_Entry To_List(#skills)
```

#listcount += 1

Endwhile

\* when identifying this screen we set the name of the "+" sign = "moreindicator". The presence of that field in the last row of the subfile tells us whether there is another page. The last row is one less than the current value of #listcount.

```
#listcount -= 1
```

If (#myscreen\_wrapper.check\_field\_exists( "moreindicator" #listcount )) #myscreen\_wrapper.getvalue From("moreindicator") Value(#MoreVal) Index(#listcount) #NextPage := (#MoreVal.trim \*NE "") Endif

Endroutine

\* Listen to messages from RAMP and the 5250 application

Evtroutine Handling(#myscreen\_wrapper.RampMessage) Umessagetype(#MsgType) Umessagetext(#MsgText)

Case (#msgtype.value)

```
When Value_Is('= VF_ERROR')
```

\* Fatal messages reported by Ramp (e.g. Navigation request failed, etc). If in design mode, show the underlying 5250 screen. Otherwise, make the error message

\* appear in a message box on top of the command

```
If (#usystem.iDesignMode = true)
```

Set Com(#myscreen\_wrapper) Visible(True)

Else

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value)

#com\_owner.avshowmessages

Endif

\* Messages sent by the System i application or unknown form was encountered

When Value\_Is('= VF\_INFO' '= VF\_UNKNOWN\_FORM')

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value)

\* Failure to initialize RAMP. Could occur for mainly one of two reasons

When Value\_Is('= VF\_INIT\_ERROR')

Message Msgid(dcm9899) Msgf(dc@m01) Msgdta(#msgtext.value)

#com\_owner.avshowmessages

Otherwise

Use Builtin(message\_box\_show) With\_Args(ok ok info \*Component ('Unknown message type ' + #MsgType + 'encountered'))

Endcase

Endroutine

\* \_\_\_\_\_

\* Handle changes in any of the fields on the panel \* \_\_\_\_\_

Evtroutine Handling(#PanelFields<>.Changed)

\* Enable the save button Set Com(#SAVE\_BUTTON) Enabled(True)

\* Lock the framework and set a message for the user Use Builtin(bconcat) With\_Args('Changes made to employee' #GiveName #Surname 'have not been saved yet.' 'Do you want to save them before continuing?') To\_Get(#sysvar\$av)

Set Com(#avFrameworkManager) Ulocked(USER) Ulockedmessage(#sysvar\$av)

Endroutine

\* \_\_\_\_\_

*	Enter	key	pressed
---	-------	-----	---------

Evtroutine Handling(#PanelFields<>.KeyPress) Options(\*NOCLEARMESSAGES \*NOCLEARERRORS) Keycode(#KeyCode)

If Cond('#KeyCode.Value = Enter')

\* If there no changes have been made issue message and ignore enter

\_\_\_\_\_

If Cond('#SAVE\_BUTTON.Enabled \*EQ True')

Invoke Method(#Com\_Owner.Save)

Else

\* Issue 'There are no changes to save' message Use Builtin(Message\_box\_show) With\_Args(ok ok Info \*Component \*MTXTDF\_NO\_SAVE)

Endif

Endif

Endroutine

\* \_\_\_\_\_ \* Handle the save button \* \_\_\_\_\_

Evtroutine Handling(#SAVE\_BUTTON.Click)

\* Call the Save method

Invoke Method(#Com\_Owner.Save)

Endroutine

\* \_\_\_\_\_

\* Handle Save

Mthroutine Name(Save)

```
* Set the 5250 field values to the values from this panel
#myscreen_wrapper.setvalue Infield('surname') Value(#surname.value)
#myscreen_wrapper.setvalue Infield('givename') Value(#givename.value)
#myscreen_wrapper.setvalue Infield('address1') Value(#address1.value)
#myscreen_wrapper.setvalue Infield('address2') Value(#address2.value)
#myscreen_wrapper.setvalue Infield('address3') Value(#address3.value)
#myscreen_wrapper.setvalue Infield('homephone') Value(#phonehme.value)
#myscreen_wrapper.setvalue Infield('postcode') Value(#POSTCODE.value)
```

```
* Send the Enter key with the payload
#myscreen_wrapper.sendkey Key(#myscreen_wrapper.KeyEnter)
Payload(UPDATE_EMPLOYEE)
```

Endroutine

*	
*	Handle Termination
*	

Mthroutine Name(uTerminate) Options(\*REDEFINE)

\* Clean up the colelction of fields on the panel Invoke Method(#PanelFields.RemoveAll)

\* Do any termination defined in the ancestor Invoke Method(#Com\_Ancestor.uTerminate)

Endroutine

End\_Com

### 例3: System iのディスク使用量の表示

画面ラッパーは、非表示の5250画面から値を取得し、まったく異なる方 法でその値を表示します。この例では、System iのディスク使用量をグラ フィックで表示します。



Work with Disk Status画面にアクセスするには、コマンド行に*wrkdskstsと*入力します。この例で、Work with Disk Status画面に指定された名前は「DiskStatus」です。

Disk Status画面では、サブファイルの%Use列を読み取り、グラフにその データを入力します。

Function Options(\*DIRECT)

Begin\_Com Role(\*EXTENDS #VF\_AC010) Height(559) Hint(\*MTXTDF\_DET1) Layoutmanager(#ATLM\_1) Width(557) Define\_Com Class(#PRIM\_GRID) Name(#DiskSts) Displayposition(1) Height(150) Left(109) Parent(#PANL 2) Rowheight(19) Tabposition(1) Top(15) Width(212) Define Com Class(#PRIM GDCL) Name(#GDCL 1) Caption('Disk Unit') Captiontype(Caption) Displayposition(1) Parent(#DiskSts) Source(#VF\_ELTYP) Width(29) Define\_Com Class(#PRIM\_GDCL) Name(#GDCL\_2) Caption('% Use') Captiontype(Caption) Displayposition(2) Parent(#DiskSts) Readonly(False) Source(#VF\_ELTXTS) Width(30) Widthtype(Remainder) Define\_Com Class(#PRIM\_GRPH) Name(#GRPH\_1) Displayposition(1) Height(370) Left(0) Parent(#PANL\_3) Scatterstyle(SymbolAtPoints+Solid) Surfacestyle(ConnectLinesInBlack) Tabposition(1) Top(0) Width(557) Xcaption('Disk Units') Ycaption('% Use') Define\_Com Class(#PRIM\_GRCL) Name(#GRCL\_1) Columnrole(Label) Displayposition(1) Parent(#GRPH 1) Source(#VF ELTYP) Define Com Class(#PRIM GRCL) Name(#GRCL 2) Columnsymbol(HollowUpTriangle) Displayposition(2) Parent(#GRPH\_1) Source(#VF ELWIDP) Define\_Com Class(#vf\_sy122) Name(#myscreen\_wrapper) Displayposition(3) Height(513) Left(144) Parent(#PANL\_1) Top(24) Visible(False) Width(593) Define\_Com Class(#PRIM\_PANL) Name(#PANL\_1) Displayposition(1) Height(559) Layoutmanager(#SPLM\_1) Left(0) Parent(#COM\_OWNER) Tabposition(1) Tabstop(False) Top(0) Width(557) Define\_Com Class(#PRIM\_ATLM) Name(#ATLM\_1) Define\_Com Class(#PRIM\_ATLI) Name(#ATLI\_1) Attachment(Center) Manage(#PANL\_1) Parent(#ATLM\_1) Define Com Class(#PRIM SPLM) Name(#SPLM 1) Define Com Class(#PRIM PANL) Name(#PANL 2) Displayposition(1) Height(181) Layoutmanager(#FWLM 1) Left(0) Parent(#PANL 1)

Tabposition(2) Tabstop(False) Top(0) Width(557)

Define\_Com Class(#PRIM\_PANL) Name(#PANL\_3) Displayposition(2) Height(370) Layoutmanager(#ATLM\_2) Left(0) Parent(#PANL\_1)

Tabposition(3) Tabstop(False) Top(189) Width(557)

Define\_Com Class(#PRIM\_SPLI) Name(#SPLI\_1) Manage(#PANL\_2) Parent(#SPLM\_1) Weight(1)

Define\_Com Class(#PRIM\_SPLI) Name(#SPLI\_2) Manage(#PANL\_3) Parent(#SPLM\_1)

Define\_Com Class(#PRIM\_ATLM) Name(#ATLM\_2)

Define\_Com Class(#PRIM\_ATLI) Name(#ATLI\_2) Attachment(Center) Manage(#GRPH 1) Parent(#ATLM 2)
```
Define Com Class(#PRIM FWLM) Name(#FWLM 1)
Direction(TopToBottom) Flowoperation(Center) Margintop(15)
Spacingitems(2)
Define_Com Class(#PRIM_FWLI) Name(#FWLI_3) Manage(#DiskSts)
Parent(#FWLM 1)
Define Com Class(#PRIM PHBN) Name(#PHBN 1) Caption('Refresh
Statistics') Displayposition(2) Left(331) Parent(#PANL_2) Tabposition(2)
Top(15) Width(117)
Define_Com Class(#PRIM_FWLI) Name(#FWLI_6) Manage(#PHBN_1)
Parent(#FWLM_1)
* This example shows a graph with the percentage of activity of each of the
System i disk units
Mthroutine Name(uInitialize) Options(*REDEFINE)
* Do any initialization defined in the ancestor
Invoke Method(#Com_Ancestor.uInitialize)
Set Com(#grph_1) Graphtype(Bar)
Set Com(#myscreen_wrapper) Ucommand(#com_owner)
Endroutine
Mthroutine Name(uExecute) Options(*REDEFINE)
* Do any execution logic defined in the ancestor
Invoke Method(#Com_Ancestor.uExecute)
Invoke Method(#myscreen wrapper.MakeRampTSAvailable)
Endroutine
Evtroutine Handling(#myscreen wrapper.RampTSAvailable)
* Clear the lists
Clr List Named(#DiskSts)
Clr List Named(#grph 1)
* Run teh scripts to navigate to work with disk status
* The navigation only involves 3 screens:Login -> System i main menu, type
wrkdsts in teh command line -> Destination Work with disk status.
Invoke Method(#myscreen wrapper.navigatetoscreen) Name(wrkdsksts)
```

Endroutine

Evtroutine Handling(#myscreen\_wrapper.vHandleArrive)

Arrivedscreen(#CurrentScreen) Previousscreen(#PreviousScreen)

Arrivedpayload(#Payload)

Define\_Com Class(#prim\_boln) Name(#MoreRecords)

Case (#CurrentScreen)

When Value\_Is(= wrkdsksts)

\* Payloads are destroyed when the ARRIVE script finishes executing.

\* The payload is set to FIRST\_ARRIVAL when teh enter key is sent in System i main menu to differentiate what caused the screen to arrive:

\* SETVALUE("cmdline","wrkdsksts")

\* SENDKEY(KeyEnter, "FIRST\_ARRIVAL");

\* The same screen can arrive after sending the F10 or an F5, so we set payload to indicate so

If (#Payload = RESTART\_STATS)

#myscreen\_wrapper.sendkey Key(#myscreen\_wrapper.KeyF5)

Payload(REFRESH\_STATS)

Else

If (#Payload = FIRST\_ARRIVAL)

#myscreen\_wrapper.sendkey Key(#myscreen\_wrapper.KeyF10)

Payload(RESTART\_STATS)

Else

#com\_owner.uGetSubfilePage

Endif

Endif

Endcase

```
Set Com(#myscreen_wrapper) Visible(False)
```

Endroutine

\* When RAMP-TS encounters screens like WRKDSKSTS or WRKACTJOB, it does not recognise the lists on them as subfiles (because they are not actually subfiles).

\* Instead they are recognised as multiple fields.

\* However, your scripts can still process them similarly to lists, as long as you follow a naming standard when naming the fields.

\* In our WRKDSKSTS screen we identified the first column as UNITn and the fourth column as USEDn

Mthroutine Name(uGetSubfilePage)

#listcount := 1

Dowhile (#myscreen\_wrapper.check\_field\_exists( ("USED" +

#listcount.asstring) ))

#myscreen\_wrapper.getvalue From("UNIT" + #listcount.asstring)

Value(#vf\_eltyp)

#myscreen\_wrapper.getvalue From("USED" + #listcount.asstring)

Value(#vf_eltxts)
#VF_ELWIDP := #vf_eltxts.trim.asnumber
Add_Entry To_List(#DiskSts)
Add_Entry To_List(#grph_1)
#listcount += 1
Endwhile
Endroutine
Mthroutine Name(uTerminate) Options(*REDEFINE)
* Do any termination defined in the ancestor
Invoke Method(#Com_Ancestor.uTerminate)
Endroutine
* Refresh statistics
Evtroutine Handling(#PHBN_1.Click)
Invoke Method(#myscreen_wrapper.MakeRampAvailable)
Foraction(RampTSAvailable)
Endroutine
End_Com

## プログラミング・テクニック

ここでは、アプリケーション設計に関する一般的な問題の解決に役立ち、RAMPアプリケーションに高度な機能を簡単に統合するためのプロ グラミング・テクニックについて説明します。

画面の定義

• 1つの画面で複数のモードを表示する場合の処理 プログラミング

- ファンクション・キーの記述の使用によるRAMPボタンの条件設定
- コマンドハンドラータブと多数の5250 Destination画面
- 高度なプロンプト
- シングルJunctionポイント(SJP)を使用するRAMP設計手法
- HIDE\_CURRENT\_FORMを使用したコマンド・ハンドラー・タブへのアクセスの管理

## ファンクション・キーの記述の使用によるRAMPボタンの条件 設定

この例では、5250画面上で以下のファンクション・キーの記述を

F3=Exit F4=Prompt F9=Retrieve F12=Cancel F13=Information Assistant F16=System Main menu

RAMPボタンおよびファンクション・キーと一致させる方法について説 明します。

この例では、JavaScriptファンクション

SHARED.apply5250FunctionKeyPatternsを使用します。このファンクションは指定した画面の行でFn=xxxxxパターンを検索し、これらのパターンを使用して、一致するRAMP-TSボタンおよびファンクション・キーの条件を設定します。

一致にはパターンのテキスト部分は含まれないことに注意してください。通常、RAMP-TSでは、ボタンのキャプションはRAMPツールの5250 画面とは無関係に定義されます。

このサンプルを実行するには、SHARED.apply5250FunctionKeyPatterns内 のファンクションをコピーしてSHAREDスクリプト・オブジェクト、 UF\_SY420\_RTS.JSファイルに貼り付けます。ファンクションのロジック は以下のとおりです。

- すべてのRAMP-TSファンクション・キーおよびボタンは、初期時 に無効になっています。
- 5250画面の指定された行でFn=XXXXXパターンを検索します。関連するファンクション・キーおよびボタンは、標準の SETKEYENABLEDファンクションを呼び出して有効にします。
- 特定のファンクション・キーを強制的に常に有効にしていた場合、 これはSETKEYENABLEDを使用して実行されています。
- 特定のファンクション・キーを強制的に常に無効にしていた場合、 これはSETKEYENABLEDを使用して実行されています。

ファンクションの呼び出し

以下の例のように、Destination画面の到着スクリプトからファンクションを呼び出します。

vHandle\_ARRIVE:function(oPayload, oPreviousForm)
{

var bReturn = true;

SHARED.apply5250FunctionKeyPatterns(22,23); /\* Must be before SHOW CURRENT FORM() \*/

SHOW\_CURRENT\_FORM(true); /\* Show the form in the framework and show VLF buttons \*/

HIDE\_5250\_BUTTONS(); /\* Hide any 5250 style buttons displayed \*/

GET\_FORM\_MESSAGE(22); /\* Extract messages and hide the message line \*/ SETBUSY(false); /\* Last thing done - turn off the busy state \*/

ここでは、到着画面の行22および23でFn=xxxxテキスト・パターンを検 索し、RAMP-TSボタンおよびファンクション・キーをこれらのパターン と一致させます。これらは単にテキスト・パターンであり、5250画面で どのファンクション・キーが実際に有効であるかについて、プログラム として厳密には何も実行しません。テキスト・パターンは5250画面上に 配置され、コンピュータ・プログラムではなくユーザーにより読み取ら れます。

#### パラメータ

このファンクションには以下のパラメータがあります。

開始行番号	必須。整数。Fn=xxxxパターンを検索する最 初の行。
終了行番号	省略可能。整数。Fn=xxxxパターンを検索す る最後の行。デフォルト値は開始行番号と 同じ値です。
常に有効にする	省略可能。ファンクション・キー識別子の
キー/ボタン	配列。例:[KeyEnter,KeyF11]
常に無効にする	省略可能。ファンクション・キー識別子の
キー/ボタン	配列。例 : [ KeyF12, KeyF3 ]

#### 例:

SHARED.apply5250FunctionKeyPatterns(22, 23, [KeyEnter,KeyF1], [KeyF12] );

これは、画面の行22から23をチェックし、EnterおよびF1キーを常に有効にし、F12キーを常に無効にするように指定しています。

無条件にPageUpキーとPageDownキーを有効にする場合、これらのキー を有効にするための一致するテキスト・パターンはありません。次のよ うに記述します。

SHARED.apply5250FunctionKeyPatterns(22, 23, [ KeyPageUp, KeyPageDown ] );

SHAREDファンクションでは、これを継続的に実行することもできます。

「ファンクションに関する質問」も参照してください。

ファンクションに関する質問

SHARED.apply5250FunctionKeyPatternsが予想どおりに機能しない場合、どうしたらよいですか?

アプリケーション・レベル追跡機能を常に有効にして画面を実行し、結 果を検証してください。

デザイン・モード(検証画面など)で実行中に表示されるその他のボタン によりこの問題が明確になっていないかも確認します。実際のテストで は、UF\_EXEC (エンド・ユーザー)モードでロジックを実行します。

Destination画面のボタンのキャプションにより問題が明確になっていな い可能性もあります。5250画面のテキスト・パターンはF7=ホップを指 定しているが、RAMPツールの画面の定義はF7キーに「スキップ」とい うキャプションのボタンを使用するように指定していることがありま す。

SHARED.apply5250FunctionKeyPatternsのロジックを変更する ことはできますか?

はい。このコードは、SHAREDオブジェクトで使用するサンプルとして 提供されています。

これは、必要に応じてユーザーが変更および管理できるコードです。 例えば、最初にすべてのファンクション・キーを有効にしたい場合があ ります。どのファンクション・キーが実際に有効になっているかを現在 の画面のみで確認するのは不可能であるため、F24=その他のキー・スタ イルの画面を使用することは、可能な解決法です。これを実行するに は、以下の行を変更する必要があります。

for (var i = 0; i < oForm.vFKERTS.length; i++) { if (oForm.vFKERTS.charAt(i)
!= "0") oForm.vFKERTS = InsertString(oForm.vFKERTS,"0",i); }</pre>

#### 変更後:

for (var i = 0; i < oForm.vFKERTS.length; i++) { if (oForm.vFKERTS.charAt(i)
!= "1") oForm.vFKERTS = InsertString(oForm.vFKERTS,"1",i); }</pre>

呼び出し側で常に指定せずに、EnterキーおよびF1キーを常に有効にしたい場合があります。このため、この行を

SHARED.apply5250FunctionKeyPatternsの最後に追加できます。

SETKEYENABLED(GLOBAL\_oCurrentTSform.symbolicName,KeyEnter,true,SETKEYENABLED(GLOBAL\_oCurrentTSform.symbolicName,KeyF1,true,tru

また、Fn=xxxxx文字列のxxxxテキスト部分に何らかの方法によりボタン・キャプションを連結したい場合もあります。2番目のキー一致要素、fkeyMatch[2]および標準のRAMP-TS

OVERRIDE\_KEY\_CAPTION\_SCREENファンクションを使用してこれを 実行することもできます。

特殊なRAMPツール・オプションを使用して、すべてのオプションや組み合わせを自動的かつ効率的に扱うことができないことが理解されたかもしれません。多数のチェック・ボックスやオプションがあるため、これらの意味や動作を理解することはできないかもしれません。このような汎用スクリプト・アプローチの使用は簡易的で、ニーズに合わせてアプローチをカスタマイズすることができます。

SHARED.apply5250FunctionKeyPatternsをすべての到着スクリプト内に配置するにはどうすればよいですか?

挿入先のSHAREDオブジェクトですでに何かを呼び出していない限り、 既存のスクリプトに個別に追加する必要があります。

また、SHARED.apply5250FunctionKeyPatternsがすべての新しいスクリプ ト内に自動的に生成されるように、到着スクリプトのスケルトンを変更 することもできます。『RAMP-TSガイド』(lansa050.chm)の「スクリプ トのスケルトン」を参照してください。

### SHARED.apply5250FunctionKeyPatterns

このサンプルJavaScriptファンクションは、指定した画面の行で Fn=xxxxxパターンを検索するように設計されています。下記の追加さ れたファンクションをコピーし、SHAREDスクリプト・オブジェクト、 UF\_SY420\_RTS.JSファイルに貼り付けます。

// Apply Fn=xxxxxx function key patterns to buttons and function keys enabled on the current RAMP-TS screen

apply5250FunctionKeyPatterns

:function(iLowRow,iHighRow,aForceEnable,aForceDisable)

{

if (GLOBAL\_oAXESInterface == null) return; // No AXES interface

if (GLOBAL\_oCurrentTSform == null) return; // No current AXES form

if (oGLOBAL\_CurrentFORM == null) return; // No RAMP-TS definition for the form

TRACE("SHARED.applyFunctionKeyPatterns started");

if (iHighRow == null) iHighRow = iLowRow; // default is same as low row
var allkeys = "";

var typeOUTPUT = GLOBAL\_oAXESInterface.Element.TYPE\_OUTPUT; var oForm = oGLOBAL\_CurrentFORM;

// Disable all function keys and buttons to start with.

// Note that the function keys (oForm.vFKERTS) and the buttons

(oForm.vFKEVLF) are BOTH disabled here

TRACE("SHARED.applyFunctionKeyPatterns is disabling all function keys and all buttons");

```
for (var i = 0; i < oForm.vFKEVLF.length; i++) { if
```

```
(oForm.vFKEVLF.charAt(i) != "0") oForm.vFKEVLF =
```

InsertString(oForm.vFKEVLF,"0",i); }

for (var i = 0; i < oForm.vFKERTS.length; i++) { if (oForm.vFKERTS.charAt(i)
!= "0") oForm.vFKERTS = InsertString(oForm.vFKERTS,"0",i); }</pre>

// Strip all output fields on the specified lines to create a long string of function
keys strings

```
for (var iRow = iLowRow; iRow <= iHighRow; iRow++)
```

```
{
```

```
var aAElement = GLOBAL_oCurrentTSform.getElementsByRow(iRow);
for (var i = 0; i < aAElement.length; i++)</pre>
```

{

```
var oAXESElement = aAElement[i];
if (oAXESElement.type == typeOUTPUT) { allkeys += " " +
oAXESElement.getValue(); }
// This RegExp looks for strings of the form F1=XXXX (where "F" can be F,
PF, FP, CF
// or Cmd) XXXX can be a string of any length terminating at more than one
space,
// the end of the line or another instance of "F1=" (thats the ?= look ahead
group).
// All groups are forgotten (that's the ?:)except the function number and the
XXXX text.
var reFKey = (:F|PF|FP|CF|Cmd)((d+)[=-](.*?))
(?:\b(?:F|PF|FP|CF|Cmd)\d+[=-])|\s{2,}|$)/gi;
var fkeyMatch = reFKey.exec(allkeys);
while (fkeyMatch != null)
var key = "F" + TRIM_RIGHT(fkeyMatch[1]);
// Note that the function key and the button are both being enabled here
SETKEYENABLED(GLOBAL_oCurrentTSform.symbolicName,key,true,true);
fkeyMatch = reFKey.exec(allkeys);
// Enable any forced buttons.Note that the function key and the button are
BOTH enabled
if (aForceEnable != null)
TRACE("SHARED.applyFunctionKeyPatterns is forcing the enablement of
specified keys/buttons");
for (var i = 0; i < aForceEnable.length; i++) {
SETKEYENABLED(GLOBAL oCurrentTSform.symbolicName,aForceEnable)
// Disable any forced buttons.Note that the function key and the button are
BOTH disabled
if (aForceDisable != null)
TRACE("SHARED.applyFunctionKeyPatterns is forcing the disablement of
specified keys/buttons");
```

for (var i = 0; i < aForceDisable.length; i++) {
 SETKEYENABLED(GLOBAL\_oCurrentTSform.symbolicName,aForceDisable
 }
 // Finished
 TRACE("SHARED.applyFunctionKeyPatterns ended");
 }, // <--- Note the comma</pre>

1つの画面で複数のモードを表示する場合の処理

System iアプリケーションでは、1つの画面で複数のモードを処理できます。

例えば、ADD、CHANGE、DISPLAY および DELETEが可能な1つの画面 をアプリケーションに作成できます。

これを処理する方法の例については、チュートリアル「RAMP-TS008: 照会画面の更新モードへの変更」を参照してください。

## コマンドハンドラータブと多数の5250 Destination画面

多数の5250 Destination画面を1つのコマンドハンドラータブに関連付ける ことができます。この機能にはさまざまな用途があり、いくつかのアプ リケーション設計に関する一般的な問題を解決できる場合があります。 例えば、5つの異なる5250 Destinationフォームがあり、各フォームがリ ポート作成基準を要求し、リポートをバッチに送信するとします。この 5つの5250画面をuReport1、uReport2、...uReport5と呼ぶことにします。 このアプリケーションのプロトタイプを作成するとき、複数の方法でこ の5つの異なるリポートを処理できます。

#### ビジネスオブジェクトの数が多すぎる

各リポートを"Report 1"から"Report 5"までの固有のビジネスオブジェクトとして定義します。この場合、アプリケーションのナビゲーション・ ツリーは次のような構造になります。



ユーザーがいずれかのリポートをクリックすると、フォームの右側全体 にリポートの関連5250フォームが表示されます。

#### コマンドタブの数が多すぎる

Report 1からReport 5までの5つの関連するコマンドまたはアクションを持つ"Reports"という名前の1つのビジネス・オブジェクトを定義します。この場合、アプリケーションのナビゲーションとコマンド・ハンドラー・ タブは次のような構造になります。



ユーザーがリポートタブをクリックすると、関連する5250フォームがタ ブに表示されます。いずれかのタブがデフォルトになります。

これらの方法にはいくつかの問題があります。

- 1番目の方法は、使用するビジネスオブジェクトの数が多すぎます。
- 2番目の方法は、使用するコマンド(またはアクション)の数が多す ぎます。

50個または500個の異なるタイプのリポートがある場合、どうなるでしょうか。

対策:ダイナミックコマンドタブ

Reportsという名前の1つのビジネスオブジェクトを作成し、それにSubmit Report Requestという1つのダイナミックコマンドハンドラータブを設定 することで解決できます。

例えば、2つのタブを表示するように設定されたReportsビジネスオブ ジェクトがあるとします。1番目のタブは"Submit Report Request"で、2番 目のタブは、複数の異なる方法でリポートバッチジョブの出力を表示で きる"View Spool Files"です。



この例では、実際に表示される5250 Destination画面を実行時に動的に変 更する必要があるので、"Submit Report Request"コマンドハンドラータブ だけに注目します。

この1つのタブに表示される5250画面を変更するには、主に次の2つの方法を使用します。

- ユーザー制御コマンドタブと多数のDestination画面
- プログラム制御コマンドタブと多数のDestination画面

#### 制限事項

- Framework SWITCH機能を使用して、多数の5250 Destination画面が 定義されたコマンドハンドラーに切り替えることはできません。
- このオプションを使用するコマンドハンドラーは、個別のポップ アップウィンドウではなく、メインのフレームワークウィンドウに配 置する必要があります。

### ユーザー制御コマンドタブと多数のDestination画面

複数のDestination画面を1つのコマンドハンドラーに関連付けることができます。この場合、フレームワークは、エンドユーザーが使用する画面を決定するためのウィンドウを自動的に表示します。

- Reportsビジネスオブジェクトを作成します。
- Reportsビジネスオブジェクトにフィルターがないこと、メイン・ フォームの右側の表示エリア全体を使用するようにビジネスオブジェ クトが設定されていることを確認します。
- Reportsに"Submit Report Request"という1つのビジネスオブジェクト レベルのコマンドハンドラーを設定します。このコマンドをデフォル トにします。
- 通常の方法で5つの5250 Destinationフォームを定義します。
- 5つすべての5250 DestinationフォームをSubmit Report Requestコマン ドハンドラータブに関連付けます。この操作を実行すると、RAMP ツールは、複数のDestinationフォームを1つのコマンドハンドラータブ に関連付けていることを通知します。
- アプリケーションを実行します。

Submit Report Requestコマンドタブを表示する必要がある場合は、このコマンドタブは複数の5250 Destinationフォームがあることを検出し、ユーザーに使用するフォームを選択するように指示します。



## プログラム制御コマンドタブと多数のDestination画面

コマンドタブに表示する画面を制御するプログラムを作成できます。少 し設定が難しいですが、拡張は簡単です。

Reportsビジネスオブジェクトを作成します。

Reportsに"Submit Report Request"という1つのインスタンス・レベルのコ マンドハンドラーを設定します。このコマンドをデフォルトにします。 Reportsビジネスオブジェクトで、非表示のフィルターを作成し、インス タンスリストに5つのリポート名を入れます。関連するリポートを識別す

るためのAKeyN値またはNKeyN値を必ず追加してください。例:

BEGIN\_COM ROLE(\*EXTENDS #VF\_AC007) HEIGHT(182) WIDTH(326) Mthroutine uInitialize Options(\*Redefine)

#Com\_Owner.avHiddenFilter := TRUE

#avListManager.ClearList

Invoke #avListManager.AddtoList Visualid1('Report 1') Visualid2('Daily production report') AKey1('uReport1') NKey1(1)

Invoke #avListManager.AddtoList Visualid1('Report 2') Visualid2('Monthly production report') AKey1('uReport2') NKey1(2)

Invoke #avListManager.AddtoList Visualid1('Report 3') Visualid2('Overloaded production report') AKey1('uReport3') NKey1(3)

Invoke #avListManager.AddtoList Visualid1('Report 4') Visualid2('Monday Morning Management Report') AKey1('uReport4') NKey1(4)

Invoke #avListManager.AddtoList Visualid1('Report 5') Visualid2('Daily production report') AKey1('uReport5') NKey1(5)

\* Instance list updating has been completed

INVOKE METHOD(#avListManager.EndListUpdate)

Endroutine

End\_Com

インスタンスリストとコマンドハンドラータブは、次のようにユーザー に表示されます。

On Tool Bar	Report - Description
Demo Application     So Ho	Report 1 Daily production report Report 2 Monthly production report
Reports     Departments     Employees	Report 3         Overloaded production report           Report 4         Monday Morning Management Report           Report 5         Daily production report
Sections     Monthly Reports     Online Reports     Annual Reports	Report : Submit Report Request (Report 2-Monthly production report)

ユーザーがインスタンスリストのいずれかのリポートをクリックする

```
と、関連する5250 Destination画面がタブに表示されます。
```

通常の方法で5つの5250 Destinationフォームを定義します。

1番目の5250 Destinationフォーム(例:uReport1)だけを"Submit Report Request"コマンド・ハンドラー・タブに関連付けます。

インスタンス・リストの数値キー値NKey1に要求したリポート番号が含まれている場合は、uReport1に移動するスクリプトを次のように変更できます。

/\* See is the report number in the instance list is for some other report \*/ /\* If it is then "reroute" this request to correct 5250 destination form \*/ switch ( objListManager.NKey1[0] )

{

case 2:NAVIGATE\_TO\_SCREEN("uReport2"); return; case 3:NAVIGATE\_TO\_SCREEN("uReport3"); return; case 4:NAVIGATE\_TO\_SCREEN("uReport4"); return; case 5:NAVIGATE\_TO\_SCREEN("uReport5"); return;

/\* Normal navigation logic to handle report number 1 \*/
NAVIGATE\_TO\_JUNCTION("whatever");

Etc, etc .....

インスタンス・リストの英数字キー値AKey1に要求した5250 Destination 画面の名前が含まれている場合は、スクリプトを次のように変更できま す。

/\* See is the 5250 screen name is this screen's name \*/ /\* If it is then "reroute" this request to correct 5250 destination form \*/ if (objListManager.AKey1[0] != "uReport1")

```
NAVIGATE_TO_SCREEN(objListManager.AKey1[0]); return;
```

}

```
/* Normal navigation logic to handle this screen */
NAVIGATE TO JUNCTION("whatever");
```

```
Etc, etc .....
```

他の場面での使用

この例は、5つの異なる5250リポート画面を1つのコマンドハンドラータ ブに表示することを動的に選択する方法を示します。

選択は、ユーザーまたはスクリプトに記述したロジックによって行うこ とができます。

以下を理解している必要があります。

- "ERP"アプリケーションに3つの異なるタイプの"Orders"、例えば International、National、およびLocalが存在する場合、この方法を使用 して、3つの異なる5250 Destination画面を"Details"という1つのコマン ドハンドラータブに表示できます。
- インスタンスリストを使用して、5250 Destinationフォームの「メニュー」を動的に作成できます。
- すべての5250 Destination画面を正式にコマンドハンドラータブに接続する必要があるわけではありません。5250 Destination画面は、 NAVIGATE\_TO\_SCREEN()ファンクションを使用してナビゲーション・スクリプトに組み込まれたロジックによって動的にタブに添付 (表示)することができます。

## 高度なプロンプト

簡単なVisual LANSAフォームをフィールドに関連付けることによって、 5250 RAMP画面で高度なプロンプトを簡単に実行できます。

例えば、複数の異なる品目サイズをラジオボタンとして表示するVisual LANSAフォームを作成し、そのフォームをRAMP画面のItem Sizeフィールドに関連付けてその値を返すことができます。



プロンプタ・フォームでは、ラジオ・ボタン、ソート可能なツリー、リ スト・ビューなどすべての高度なVisual LANSA機能にアクセスできま す。

System iのプロンプトとは異なり、Visual LANSAのプロンプタ・フォームでは、必ずSystem iサーバーとの対話処理が行われるわけではありません。このため、処理速度が向上します。

さらに、高度なプロンプトを使用して、5250デバイス上では実現できな い機能を提供できます。例えば、電話番号のプロンプタは、電話番号検 索Webサイトを表示し、ユーザーがいずれかの電話番号を選択すると、 その値を5250画面に反映します。

プロンプタ・フォームのその他の使用法

プロンプタ・フォームは、さまざまな方法で高度なWindowsデスクトッ プ統合に使用できます。例えば、プロンプタ・フォームは以下の動作を 実行できます。

- 支払い遅延e-mailを準備して、送信します。
- インターネット・サイトまたはWebサービスを経由してクレジット 参照チェックを実行します。
- System iサーバーから情報を抽出し、MS-Excel表計算を作成し、 MS-Excelを起動して表計算の情報を表示します。
- リンク先のWebページまたは関連するWebページを表示します。
- リンク先のPDFドキュメントまたは関連するPDFドキュメントを表

示します。

その他の高度なWindowsデスクトップ統合のフォームも作成できます。

高度なプロンプタ・フォームは、5250画面上の情報のサブセットに高度 な機能を統合するための簡単な方法として用意されています。 すべての デスクトップ統合要件を満たすために新しいRAMP画面全体をRAMPア プリケーションに追加することも可能です。 プロンプタ・フォームの使用

プロンプタ・フォームの作成

プロンプタ・フォームを通常のVLフォームとして作成します。

標準の動作を継承するようにそのAncestorプロパティをVF\_AC017に設定 する必要があります。

プロンプタ・フォームのフィールドへの関連付け

プロンプタ・フォームをフィールドに関連付けるには、RAMPウィンド ウを開いてナビゲーションツリーのセッションオブジェクトをクリック します。 セッションプロパティが表示されます。

27.7	コン ー デフォルト セッシ	/aン	
ŧ+7'	ション デフォル	木 セッジョン	
2-#	~ オフジジェウト タイフ	* RAMP	*_DEFAULT_SESSION 名前の確認
RA	MPの既定値のレ	イアウト サイズーーー	
高	\$ 300	中国 700	上左
Top	p マスクの高さ	Bo	ottom マスクの幅
RA	MP スクリーン レイア	ウト スタイルー・・・・	
	● 固定レイア	ウト	つ フローレイアウト
ースウ	0-ルパー		
	■ 水平ス加・	ールパーを表示	■ 垂直ス如ールパーを表示
7 7	「時月な5250フォーム」	の表示時にフレームの	ワークをロックする
- F	既存のコネクションのこ	ユーザー/パスワードを	『再利用する
27.7	ンを常にユーザーン	わジェクト名/みつ	を持ったサーバーヘリンクする
AM	IP-TS 是大nがオー	(法规制去限制(制力)	10
44.5	日本ロスールトの思います		10
1412	100011-101 200A		
	52507~小小*名	ファンクジョンキー	VL NDF 5- (25X VF_ACU17 7759 ±2F)
1	txtSTATE	F4	P_STATE
2	tetPHONE	F5	P_PHONE
3		F4	

特別なフィールド受け渡しエリアを使用して、フィールドに関連付ける フォームを定義します。

この例の2つのエントリーは、以下を示しています。

txtSTATEという入力フィールドが5250 Destinationフォーム上にあり、そのフォームにカーソルまたはフォーカスがある場合にユーザーがファンクション・キーF4(またはそれと同等のボタン)を押すと、要求を処理するためにP\_STATEというVLフォームが呼び出されます。

txtPHONEという入力フィールドが5250宛先フォーム上にあり、その フォームにカーソルまたはフォーカスがある場合にユーザーがファンク ション・キーF5(またはそれと同等のボタン)を押すと、要求を処理する ためにP\_PHONEというVLフォームが呼び出されます。

高度なプロンプタ・フォームの機能

ユーザーがいずれかのVLフォームを呼び出すために必要な操作を実行すると、次の動作が発生します。

- スクリプトのHANDLE\_PROMPTファンクションが、そのフィール ドに関連付けられているプロンプタ・フォームを表示するために呼び 出されます。必要に応じて、このファンクションを使用して追加情報 をフォームに渡すことができます。
- フォームを初めてセッションで使用する場合は、uInitializeメソッドが呼び出されます。これにより、初回処理を実行できます。
- 現在の5250 Destinationフォーム上のすべての名前付きフィールドの 値が抽出され、VLフォームに使用可能になります。
- ユーザーに表示する項目の準備と配置を実行するためにVLフォームのuShowメソッドが呼び出されます。
- ユーザーが選択すると、VLフォームは現在の5250 Destination フォーム上の名前付きフィールドの値を変更できます。

このトピックを詳しく学習するための例 リポジトリには、次のVisual LANSAフォームが用意されています。 コンボボックス

DF\_PRM01では、次のように米国の州のコンボボックスを使用できます。

ALABAMA - AL	^
ALASKA - AK	
AMERICAN SAMOA - AS	
ARIZONA - AZ	
 ARKANSAS - AR	
CALIFORNIA - CA	1

#### ラジオボタン

DF\_PRM02では、製品サイズのラジオボタンを使用できます。



列のリスト DF PRM03では通常、名前順に従業員が表示されます。

LGE		Pror	npt	
GARY	DE D52503 - Promoting Example 3			
14 Hutchins Crescent,				
PADSTOW HEIGHTS.	Employee Surname B Search			
NSW.				
2016	Employee Number	Employee Surname	Employee Given	Post / Zip Code
456 1524	A1031	BLAKE	JOHN	0
151 4583	A0090	BLOGGS	FRED JOHN	0
92/02/01	A3564	BROWN	FREDDY	0
	A0070	BROWN	VERONICA	U
0/00/00				
INF 💭				
02				
25,600.04				
2.133.33	Ľ		I	

## ツリー

DF\_PRM04では、ツリーを使用して部門および部課の情報を表示できます。

Post / Zip Code	2016
Home Phone Number	456 1524
Business Phone Number	151 4583
Start date (YYMMDD)	83/03/01
Termination Date (YYMMDD)	0/00/00
Department Code Section Code Employee Salary Monthly Salary Start Date (DDMMYY) Termination Date (DDMMYY)	INF 02 INF O2 INTERNAL AUDITING FLEET ADMINISTRATION GROUP ACCOUNTS I/03/83 I/03/83 O/00/00 PURCHASING ACCOUNTING DEVELOPMENT ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING ICCOUNTING

シングルJunctionポイント(SJP)を使用するRAMP設計手法

RAMPを適用する複雑な5250アプリケーションを図で表すと次のようになる場合があります。



5250ユーザーはサイン・オンした後、大量のメニュー/Junctionをナビ ゲートして「作業画面」(5250 Destination画面)に到達し、そこで必要な 作業を実行します。

RAMPコレオグラファはこのナビゲーションに従うことができ、それを 操作しながら大量の画面内を移動するために必要なさまざまなナビゲー ションを定義できます。

RAMP開発者にとって、Junctionの識別とそのナビゲーション・スクリプトの生成は、時間を要する平凡な作業となる場合があります。

RAMP開発者の視点から考えると、5250アプリケーションが実際に次のような構造であれば、プロセス全体は処理しやすくなります。



ここでは、シングルJunctionポイント(またはプログラム)が5250 Destination画面へのすべてのアクセスを制御します。

5250アプリケーションがこのような構造になっている場合、RAMPアプリケーションの開発は簡略化され期間も短縮されます。その理由は次のとおりです。

• 1 つのJunctionだけを定義してスクリプト化すればよい。

• 宛先画面の呼び出しスクリプトが簡素化され標準化される。

残りのセクションでは、5250アプリケーションのこのタイプのビューを 設定できる方法を説明します。

この手法を、シングルJunctionポイント(SJP)モデルと呼びます。 SJPモデルは必ずしもすべての種類のアプリケーションに適用できるとは 限りませんが、このモデルが適用できれば、RAMPアプリケーションの 開発にかかる期間を短縮できます。

基本的に、SJP手法では2つの異なるアプリケーション・ビューが存在します。



このプログラム的なビューの世界を作成するには、System i 5250プログ ラム<sup>IIK</sup>がすでに存在しているか、System i 5250プログラムを作成する必 要があります。

この特別なプログラムをSJP(シングルJunctionポイント)プログラム SJP く と呼ぶことにします。

ある種の SJP は、すべてのSystem iシステムにすでに存在します。 それは、QCMD(またはコマンド入力画面)と呼ばれるプログラムで、こ のプログラムからほとんどすべての5250アプリケーションを直接的また は間接的に呼び出すことができます。ただし、セキュリティ上の理由で QCMDの使用は多くのサイトで禁じられています。したがって、この資 料の残りでは、独自の特別な SJP を作成できるさまざまな方法や、生 じる可能性のあるいくつかの問題と追加のメリットについて説明しま す。

SJPはどのように動作しますか?

実際のアプリケーションで、SJPは本当に単純なのですか?

SJPを使って他の役に立つ処理を行えますか? SJPはCL(制御言語)プログラムにする必要がありますか? SJP手法の使用に影響を与える可能性のある他の問題は何ですか?

#### SJPはどのように動作しますか?

SJPプログラムを使用すれば、RAMPアプリケーションが使用できる目的 の場所に汎用的にアクセスできます。SJPはユーザーと対話するように設 計されていません。SJPはRAMPスクリプトと対話するように設計されて います。

簡単なSJPとRAMPスクリプトは、次のように連携して動作できます。

クライアントでのRAMP Destination呼び出しスクリプトの実行



ここで使用されているRAMPスクリプトの例は、CUSTINQという5250プ ログラムを使用して顧客の詳細を表示するコマンド・ハンドラーに関連 付けられています。

実行が開始されると、このスクリプトはまずSJPというJunction画面にナ ビゲートします。

これにより、SJPプログラムがその5250画面を表示します。

次に、フィールドPGMNAMEを"CUSTINQ"という値に設定し、Enter キーを送信します。

これにより、SJPプログラムは画面を受け取ります。

SJPプログラムのCLフィールド&PGMNAME には現在"CUSTINQ"という 名前が含まれています。

次に汎用呼び出しを使用してプログラムCUSTINQを呼び出します。

RAMPスクリプトはプログラムCUSTINQを取得し、顧客番号123456を表示します。

この簡単なSJPを使用して、何百もの Destination画面のスクリプト(それ ぞれに簡単な1つのCALLインターフェースが含まれるのみ)を作成し、あ らゆる種類のSystem i 5250プログラムにアクセスできます。 実際のアプリケーションで、SJPは本当に単純なのですか?

おそらく単純ではありません。多くの場合、呼び出すプログラムに対してまたは呼び出すプログラム間で簡単なパラメータ(複雑な場合もある)を渡す必要があります。

ただし、このスタイルのアプリケーション設計では、通常、プログラム・グループは共通パラメータ・プロトコルを共有する大規模なアプリケーション・グループに分類されます。

たとえば、REQUEST\_TYPEフィールドを、RAMPスクリプト間で交換される情報に追加することで、異なるプログラム・パラメータ・プロトコルを次のように簡単に処理することができます(ロジックは擬似コード)。

WRITE and READ the 5250 screen containing PGMNAME and REQUEST\_TYPE

DOWHILE (REQUEST\_TYPE not equal to "SIGNOFF")

CASE of REQUEST\_TYPE

WHEN = "CALLP1" CALL PGM\_NAME using calling protocol 1 for parameters

WHEN = "CALLP2" CALL PGM\_NAME using calling protocol 2 for parameters

WHEN = "CALLP3" CALL PGM\_NAME using calling protocol 3 for parameters

< etc >

ENDCASE

WRITE and READ the 5250 screen containing PGMNAME and REQUEST\_TYPE ENDWHILE

RPGやCLプログラムを使用していた場合、IBM i のプログラム呼び出し インターフェースの高い柔軟性に気づいていないかもしれません。プロ グラム・パラメータは単なる記憶領域であり、プログラム間をポイン ターとして渡されます。プログラム・パラメータには次の特徴がありま す。

- パラメータの長さは、呼び出されるプログラムで定義されている長 さと同一である必要はありません。それ以上の長さがあれば大丈夫で す。これにより、SJP内でパラメータ変数の小さなセットを共有した り再利用することが非常に簡単になります。
- 実際に要求されているよりも多くのパラメータをプログラムに渡す ことができます。余分なパラメータは通常無視されます。つまり、プ ログラム内で、実際のCALLコマンドのほとんどをなくすことができ ます。
- パラメータ値はRAMPスクリプトからSJPに直接渡して、さらに呼び出し先のプログラムに渡すことができます。またこの手法を使用して、スクリプトに再び返されたパラメータ値を取得することもできます。つまり、RAMPスクリプトを使用してバッチ・スタイル・プログラムを呼び出すこともできます。

#### SJPを使って他の役に立つ処理を行えますか?

# SJPは、ほとんどすべての処理を設計できます。たとえば、次のように柔軟性が高く汎用的なIBM i コマンドへのインターフェースを提供できます。

WRITE and READ the 5250 screen containing PGMNAME, REQUEST\_TYPE and COMMAND

DOWHILE (REQUEST\_TYPE not equal "SIGNOFF")

CASE of REQUEST\_TYPE WHEN = "CMD" CALL QCMDEXEC (COMMAND 256) WHEN = "CALLP3" CALL PGM\_NAME using calling protocol 3 for parameters < etc >

#### RAMPスクリプトでCLコマンドを実行できます。

NAVIGATE\_TO\_JUNCTION("SJP"); SETVALUE("REQUEST\_TYPE","CMD"); SETVALUE("COMMAND","WRKSBMJOB \*JOB") SENDKEY(KeyEnter);

または

NAVIGATE\_TO\_JUNCTION("SJP"); SETVALUE("REQUEST\_TYPE","CMD"); SETVALUE("COMMAND","SBMJOB(BATCH) CMD("CALL PRINTORDER")") SENDKEY(KeyEnter);

RAMPスクリプトとSJP間での通信に使用される5250画面は、ユーザーに 表示される実際の5250画面というよりも、実際はプログラム・データ構 造です。 SJPはCL(制御言語)プログラムにする必要がありますか?

いいえ、SJPは5250画面の読み書きをサポートするあらゆるプログラム言 語で書くことができます。たとえば、RPG、COBOL、C、RDML(実際は RPG)などを使えます。

LANSAプログラムがある場合はRDMLを選択することをお勧めします。 RDMLを使用すれば、LANSAのプロセスやファンクションを簡単に呼び 出すことができ、また交換リストやデータ構造などの共通のプログラム 間通信メカニズムにアクセスできます。 SJP手法の使用に影響を与える可能性のある他の問題は何ですか?

主な問題の1つは、ユーザー・プロファイルとサイトのセキュリティ要件 に関連します。

ユーザーが通常の5250画面を使用するときに、たとえばUSERAがSJPプログラムにアクセスできることが望ましくない場合があります。

さらに、USERAが自身のIBM i ジョブをプロファイルUSERA下で実行 し、監査、ログ、およびセキュリティ情報に「実際」のユーザーが表示 されるよう、多くのサイトから強く要求されています(ただし、多くの同 時実行ユーザーに機能を提供するHTTP Webサーバーなどの「スレッド 化された」プロセスがSystem iサーバーで次々に使用されるため、この情 報は消失します)。

では、どのようにすれば、1つのユーザー・プロファイルUSERAでこの ような世界の異なるビューをサポートできるのでしょうか?

- 実際の5250セッションにサイン・オンした場合、通常のサインオン メニューが表示されます。
- RAMPスクリプト経由でサイン・オンした場合、SJPプログラムが メインの「メニュー」として表示されますか?

この問題にはいくつかの解決策があります。

- RAMPスクリプト経由でログインする場合、IBM i サイン・オン画 面のプログラム / プロシージャーオプションを使用してSJPプログラ ムを指定します。SJPにセキュリティ・ロジックを追加して、ユー ザーが実際の5250インターフェースを使用して同じ操作を行えないよ うにすることができます(ポイント2を参照)。
- 共通のメニュー・プログラムを使用する場合、自身がRAMPスクリ プトから呼び出されていることを検出してからSJPプログラムを呼び 出すよう、そのプログラムを変更できます。同様に、初めに共通メ ニューを表示し、「非表示」の特別メニュー・オプションを使用して SJPプログラムを呼び出すこともできます。SJPプログラムは、たとえ ば実際の人間のユーザーには実行不可能なRAMPスクリプトとの暗号 化交換を実行して、自身がRAMPスクリプトからアクセスされている ことを確認できます。
- RAMPスクリプトは、初期プログラムとしてSJPプログラムが指定 された、汎用の"USERX"として初期サイン・オンできます。SJPプロ グラムは画面を表示して実際のユーザー・プロファイルとパスワード

を要求します。RAMPログオン・スクリプトはこれを埋め込んで戻し ます。次にIBM APIを呼び出して、現在のジョブのユーザー・プロ ファイルを汎用のUSERXから実際のユーザーに変更します。再度、 実際のユーザーには実行不可能な暗号化交換を使用して、RAMPスク リプトからのアクセスであることを確認できます。
HIDE\_CURRENT\_FORMを使用したコマンド・ハンドラー・ タブへのアクセスの管理

このシナリオでは、RAMPアプリケーションが受注処理システムで作成 されています。

一部のコマンド・ハンドラー・タブ(および基本となる5250 Destinationス クリプト)で、ユーザーがキャンセルまたは完了した注文に関して、アク ションを実行できないようにする必要があるとします。

ステップ1-各インスタンス・リスト項目に一種の「コード」 列または「ステータス」列を挿入します。

ここでは、フィールド#ORDSTATUSが、インスタンス・リストの列 Acolumn9()にマッピングされています。

値には、「CAN」(キャンセル済み)、「OPN」(開いている)、「WIP」 (作業中)、または「COM」(完了)があります。

Invoke Method(#avListManager.AddtoList) Visualid1(#OrdNo) Visualid2(#CustlName) Akey1(#OrderNumber) AColumn9(#ORDSTATUS)

注:AColumn9()は、ユーザーが指定した通りに表示されるときとされないときがあります。

ステップ2-該当するスクリプトに確認コードを挿入します。 ここで、注文を変更できる5250画面のスクリプトは、キャンセルした注 文または完了した注文を表示しないように追加された確認を行っていま す。

/\* Get the order status from additional column 9 in the current order instance list entry \*/

var ORDSTATUS = objListManager.AColumn9[0];

/\* If the order is cancelled or closed, prevent the 5250 screen from being displayed, and show a message as to why \*/

if ((ORDSTATUS == "CAN") || (ORDSTATUS == "COM"))

{

HIDE\_CURRENT\_FORM("Sorry, but you are not allowed to display this order because it is cancelled or completed.");

return;

}

/\* If we reach here then it's okay to proceed to the order display screen \*/

< etc > < etc >

HIDE\_CURRENT\_FORM("message")ファンクションにより、コマンド・ タブに表示されている現在の5250画面が表示されなくなり、代わりにタ ブの中央に「Sorry, but you are not allowed to display this order because it is cancelled or completed.」というメッセージが表示されます。 AColumn9の内容(例:「CAN」、「OPN」、「WIP」、「COM」)は、ス クリプトの任意の場所で使用して、ユーザーのアクティビティを制限ま たは制御できます。

# 複数言語対応のRAMPアプリケーション

文字列

「ADD\_STRINGファンクション」と「STRINGファンクション」を参照 してください。

RAMPボタンに表示されるキャプションは、

OVERRIDE\_KEY\_CAPTION\_SCREENファンクションと OVERRIDE\_KEY\_CAPTION\_ALLファンクションを使用して複数言語対 応に変更できます。

# トラブルシューティング

エラー・メッセージ

xxxxxxは親のないスクリプトなので削除してください エンド・ユーザー・モード(UF\_EXEC)で発生し、 デザイン・モード(UF\_DESGN)では発生しない RAMP実行エラー

# 問題

RAMPアプリケーションの実行時: スクリプトの不明な動作 インスタンスリストで選択内容を変更したときに画面 が反応しない xxxxxxは親のないスクリプトなので削除してください スクリプトが親のないスクリプトで削除する必要があることを示す RAMP警告メッセージが表示されます。

このメッセージの意味

- スクリプトがDestination、Junction、またはSpecialな画面で使用されていないことを意味します。
- スクリプトは使用されていないため、削除する必要があります。
- このメッセージはRAMPの操作には影響を与えず、単に警告するだけです。
- これらの警告が多数表示される場合、複数開発者環境でマージ・ ツールの誤って使用している可能性があります。

このメッセージが表示された場合のスクリプトの削除方法

- RAMPツールを起動します。
- スクリプト・ツリー・ノードを展開し、スクリプトを探します。
- スクリプトを選択し、削除ボタンを押します。
- 複製スクリプト名(複数開発者環境で発生します)に注意してください。正しいスクリプトであることを確認します。

親のないスクリプトが発生する理由

最も可能性があるのは、親のDestination、JunctionまたはSpecialな画面でもマージせずに、最新のスクリプトすべてでマージ・ツールを使用してマージを行った場合です。

親のDestination、JunctionまたはSpecialな画面でもマージを行 なわずに、単一スクリプトでだけマージ・ツールを使用して マージする場合

通常、以前に親オブジェクトおよび関連スクリプトでマージを行い、単一の更新済みスクリプトでのみマージを行いたい場合にだけこれを実行します。最初のマージでこれを実行しないでください。親のないスクリプトを作成する危険があります。

RAMPのマージ画面および複数開発者が作成した関連スクリプトにアプローチする方法

高レベルフレームワーク設計オブジェクト、つまりアプリケーション、 ビジネス・オブジェクト、コマンドおよびコマンド・ハンドラー(タブ) が、マスター開発者によってセットアップされ、すべての開発者が同じ モデルから作業しているとします(つまり、開発者はRAMP画面およびス クリプトを定義し、事前定義されたコマンド・ハンドラー・タブにリン クするだけとします)。

- 送信者は、マージ・リストに対して作成した画面を追加する必要が あります。これにより、自動的に関連スクリプトが含められます。
- また、送信者は同じマージ・リストを、RAMP Destination画面での リンクで変更されたすべてのコマンド・ハンドラーに追加します。
- 受信者は、すべてをマスター・フレームワークにマージする必要が あります。コマンド・ハンドラーは、更新/交換として処理される必 要があり、RAMPオブジェクトは新しいオブジェクトである必要があ ります。新しいGUIDが割り当てられる必要があるケースはありませ ん。

### 同じ名前のスクリプトが存在する理由

複数開発者環境では、2名の開発者がスクリプトを作成する場合、例えば FORMSCRIPT\_137という名前の2つのスクリプトができあがってしまう ことがあります。

この2名の開発者の作業内容がマージされる場合、開発者にとっては混乱 する状況ですが、RAMPはスクリプト名を単なるキャプションとして扱 うため、RAMPが混乱することはありません。内部的には、RAMPは一 意のGUIDでスクリプトを認識して実行します。

開発者は、こういった混乱を避けるため、デフォルトのスクリプト名を 変更することができます。 インスタンスリストで選択内容を変更したときに画面が反応し ない

インスタンスリストでエントリーを選択したとき、RAMP画面がこの変 更を反映せず、最初に選択したエントリーのデータを表示します。

この問題が発生する状況

Destination画面のスクリプトを記録したが、SETVALUEファンクション の値パラメータを記録済みのハードコードされた値から代替値に変更し ていない可能性があります。

また、SETVALUEファンクションの値が引用符で囲まれていることが考 えられます。この場合、その値は代替値ではなく、リテラルとして解釈 されます。

例えば、次の例は間違いです。

SETVALUE("UtxtMachine","objListManager.AKey1[0]");

#### 対策

SETVALUEファンクションの値パラメータを代替値に指定し、引用符で 囲んでいないことを確認します。

SETVALUE("UtxtMachine", objListManager.AKey1[0]);

詳細は、以下を参照してください。

- 「スクリプトを使用したインスタンスリストとの対話」を参照して ください。
- 「ハードコードされた従業員番号を現在のインスタンスリストのエントリーに置き換える」を参照してください。

エンド・ユーザー・モード(UF\_EXEC)で発生し、デザイン・モード(UF\_DESGN)では発生しないRAMP実行エラー RAMPアプリケーションはデザイン・モードで実行できても、エンド・ユーザー・モードでは次のようなエラーが表示される場合があります。

🔊 Internet	Explorer スクリプト エラー	X
ライン: 文字: エラー: コード: URL:	このページのスクリプトでエラーが発生しました。 1 1 文字が正しくありません。 0 file:///C:/PROGRA <sup>~</sup> 1/LANSAV12/X WIN95/X LANSA/x_dem/ex ecute/VF_SY001_Nodes_RAMP_DEFAULT_SESSION.js	
	このページのスクリプトを実行し続けますか? (はい(Y) いいえ(N)	

この問題の原因

RAMPをデザイン・モードで実行する場合とエンド・ユーザー・モード で実行する場合の大きな違いは、javascriptの実行方法です。

デザイン・モードでは、RAMPに影響する変更が行われた場合、フレー ムワークを保存するたびにjavascriptは再ロードされます。javascriptが再 ロードされると、オブジェクトのプロパティは常にリセットされます。 また、有効になっているRAMPが変更された場合、フレームワークが保 存されるたびに、一連のjavascriptファイル(<system

prefix>Nodes\_nnnnnnnnnnnnnnnn.js)が生成されます。このファイルは 各セッションに1つずつ生成され、nnnnnnnnnnnnnktセッションIDを 表します。

このファイルがエンド・ユーザー・モードで使用されるファイルです。 このファイルは、フレームワークが保存された時にjavascriptとして書き 出された画面とスクリプトを表します。

エンド・ユーザー・モードでは、このファイルはセッションごとに1回 ロードされます。ファイル内の各javascriptファンクションは、セッショ ン起動時に1回だけ呼び出されます。この方法を使うことで、エンド・ ユーザー・モードのRAMPの起動時間はデザイン・モードと比較して大 幅に短縮されます。

デザイン・モードでRAMPを実行した場合にエラーが発生せず、しかし エンド・ユーザー・モードでは上記のようなエラーが発生する場合、一 番に疑われる問題はユーザー定義スクリプト(ナビゲーション・スクリ プト、呼び出しなど)に構文エラーがあることです。

対策

javascriptのどの行にエラーがあるかを探すには、ファイルを基本.HTM ファイルにロードするだけです。

たとえば、次のような内容のtest.htmというファイルを作成します。

<html>

<head>

<title>Untitled Page</title>

<script language="javascript" type="text/javascript" src="<your nodes.js file here>"></script>

</head>

<body>

Hello World

</body>

</html>

nodes.jsファイル名を<script>タグのsrc= 属性に指定して、Test.htmを javascriptと同じフォルダーに配置します。

Internet Explorerを使用して、ツール/インターネットオプションの詳細設 定タブをチェックし、「スクリプトエラーごとに通知を表示する」に チェックマークが付いていることを確認します。これでTest.htmを実行し てスクリプト・エラーが発生すると、エラーが発生した場所の行番号を 表示できます。ヒント:エラーは、次のようなjavascriptファンクション の内部に存在する可能性が最も高いと思われます。

これと実際のスクリプト名を関連付けることは少し困難です。この実際のスクリプトを探すには、ファンクション名のnnnnnnnnnnnnnnの部分と同じファイル内を検索します。次に、そのスクリプトがオブジェクトとして定義され、さらにユーザー名

(oS.uScriptUserName="NAVIGATE\_SCRIPT\_13"など)が含まれている javascript行を検索します。

RAMPツールを使用してスクリプトを編集し、エラーを修正して保存します。

# よくある質問

RAMP-TSまたはRAMP-NLとのフレームワークの関連付けについて 画面の分類で操作を間違えました。分類を変更するにはどうしたらいい ですか?

同じ名前を持つ2つの画面を区別する方法

Webブラウザー・ウィンドウはRAMPスクリプトからどのように使用で きますか?

現在の5250画面の下部に表示されているメッセージを、どのように RAMPスクリプトに取り込むことができますか?

RA(レコードの自動前進)フィールドはどのように処理するのですか? なぜ各5250画面でF12=キャンセルボタン、F3=終了ボタンおよびファン クション・キーを使用不可にする必要があるのですか?

刷新するアプリケーションで、すべての5250画面を識別して、スクリプ トを記述する必要がありますか?

RAMPツールを取得して固定セッションを割り当てる方法 独自のファンクションをスクリプトに追加できますか?

複数の区画でスクリプトを動作させるにはどのようにすればよいですか? 使用しているテーマをRAMP-TSスクリプトで指定する方法

aXes5250セッションで中断メッセージを使用しない方法

使用しているテーマをRAMP-TSスクリプトで指定する方法

RAMP-TSスクリプトで、アプリケーションで使用しているVL/VLFテーマを確認したい場合があります。

JavaScript変数GLOBAL\_VL\_Themeは、RAMP-TSスクリプトにアクセス することができます。これは、BLUE、SILVER、OLIVE、または GRAPHITEを含んでいる必要があります。

GLOBAL\_VL\_Themeのコンテンツは、変更するべきではありません。

RAMP-TSまたはRAMP-NLとのフレームワークの関連付けに ついて

XMLファイルで定義されるフレームワークは、特にRAMP-TSまたは RAMP-NLにバインドされます。つまり、そのタイプのRAMPセッション のみで使用できます。

フレームワークでRAMPを初めて使用するときは、RAMP-NLにバインド するか、RAMP-TSにバインドするかを決定します。バインド・プロセス は次のように行われます。

- まず、[フレームワークの詳細]タブ([フレームワーク] -> [プロパ ティ])でノードXMLファイルとして定義されている既存のRAMPノー ド・ファイルの内容が検証されます。ノード・ファイルにすでに RAMP-TSまたはRAMP-NLのDestinationとスクリプトが含まれる場 合、フレームワークは該当するRAMP-TSまたはRAMP-NLに自動的に バインドされます。
- ノードXMLファイルが空であるか、存在しない場合は、フレーム ワークに関連付けられているサーバーが検証されます。1つ以上の RAMP-NLサーバーが定義され、かつRAMP-TSサーバーが定義されて いない場合、フレームワークはRAMP-NLに自動的にバインドされま す。
- 他のすべての状況では、フレームワークはRAMP-TSに自動的にバインドされます。

RAMPツールを起動すると、ウィンドウのタイトルに、フレームワーク がRAMP-TSにバインドされているか、RAMP-NLにバインドされている か表示されます。

フレームワークをRAMP-TSまたはRAMP-NLにバインドするときの注意 事項は、次のとおりです。

- フレームワークに関連付けられているサーバーが適切であり、混同 されていないことを確認します。通常は、RAMP-TSまたはRAMP-NL サーバーが定義され、両方が定義されることはありません。必要に応 じて、異なるフレームワーク用に別々のサーバーXMLファイルを作 成して、1つのフレームワーク内でRAMP-TSサーバーとRAMP-NL サーバーが混用されないようにします。
- [名前を付けて保存]を使用して新しいフレームワークを作成すると
   きは、ノードXMLファイルも新しいファイル名に変更します。新し

いフレームワークを保存したら、新しいノード・ファイルと空のノード・ファイルを使用して新しいフレームワークで操作を開始できるように、ノードXMLファイルを削除します。

間違ってフレームワークをRAMP-NLまたはRAMP-TSにバインドした場 合は、メモ帳でフレームワークXMLファイルを開き、プロパティ定義、 <PROPERTY NAME="U5250HANDLER" VALUE="TS"/>または <PROPERTY NAME="U5250HANDLER" VALUE="NL"/>を検索します。 XMLファイルからこのプロパティを削除し、変更を保存します。設計者 としてフレームワークを起動し、[フレームワークの詳細]タブ([フレーム ワーク] -> [プロパティ])に進みます。ノードXML Fileの名前が正しいこ と、および使用している他のフレームワークとは異なることを確認しま す。RAMP-TSサーバーまたはRAMP-NLサーバーが定義されていること を確認します。両方のタイプが定義されていてはいけません。設計者と してフレームワークを起動し、RAMPツールをもう一度起動します。 RAMPツールの起動時に、あらかじめ定義されているバインド・ロジッ クがもう一度実行されます。

バインド・プロセスで不適切な結果が再度得られた場合は、フレーム ワークのサーバーが正しく定義されていなかったり、RAMP-TSまたは RAMP-NLに強制的にバインドする定義がノードXMLファイルに含まれ ていることが原因です。 画面の分類で操作を間違えました。分類を変更するにはどうし たらいいですか?

5250の「画面とスクリプトのリスト」で画面定義を削除します。その画面は、Tracking Info エリアに未定義として表示されます。

画面に関連付けられているスクリプトも削除されます。このため、画面 でのナビゲーションを再追跡する必要があることに注意してください。

#### 同じ名前を持つ2つの画面を区別する方法

同じ名前が指定されている画面を区別するには、バリアント名を使用します。「RAMP-TS画面スクリプトによるJavaScriptオブジェクトの定義」を参照してください。

バリアント名は、vName、vTypeなどと同様の画面のプロパティです。ス クリプトでバリアント名を参照するには、以下を使用します。

this.vLatestVariant

通常、異なるビューを持つ画面でバリアント名を使用します。代表的な 例には、Work with Active Jobsなどの画面があります。これに

WRKACTJOBと名前を付けることができます。F11を押すと、同じ画面が表示されますが、今回は別の一連の列が表示されます。

前の画面で表示されなかった、この画面の列見出しのいずれかを画面定 義の一部として使用し、バリアント名を使用して保存できます。同様 に、F11をもう一度押すと、別の異なる一連の列が表示されます。ここ でも、定義のいずれかの列見出しを使用して、別のバリアント名で画面 を保存できます。

詳細については、「RAMP-TS015:画面バリエーションの概要と処理」 を参照してください。

# Webブラウザー・ウィンドウはRAMPスクリプトからどのよう に使用できますか?

次の例は簡単なWebブラウザー・フォームです。これは、3つの入力 フィールドを引数として受け付け、それを表示し、変更を許可して、変 更された値を呼び出し元のRAMPスクリプトに返します。

```
<HTML>
<HEAD>
</HEAD>
<BODY onload="BODY Load();" onunload="BODY UnLoad();" >
<script>
function BODY_Load() /* Map arguments passed in to web form fields */
 FieldA.value = window.dialogArguments[0];
 FieldB.value = window.dialogArguments[1];
 FieldC.value = window.dialogArguments[2];
function BODY UnLoad() /* Map web form fields into return values */
 var arrayRets = new Array();
 arrayRets[0] = FieldA.value;
 arrayRets[1] = FieldB.value;
 arrayRets[2] = FieldC.value;
 window.returnValue = arrayRets;
function OK Click() /* Handle OK button by closing the web form */
 window.close();
ł
</script>
<P>Input details and click OK"<br/>
<input id="FieldA" type="text"><br/>
<input id="FieldB" type="text"><br/>
<input id="FieldC" type="text"><br/>
<input id="Button1" type="button" value=" OK " onclick="OK_Click();">
</BODY>
</HTML>
```

#### 次のような画面が表示されます。



次の例は、Webブラウザー・フォームの表示に使用するRAMPボタン・ スクリプトです。ユーザーがF5を押すとフォームが表示されます。この とき、SURNAME、GIVENAME、ADDRESS1の各フィールドが5250 フォームから取得され、マッピングされます。

```
switch (objScriptInstance.FunctionKeyUsed)
 case KeyEnter:
   SENDKEY(KeyEnter);
   break:
 case KeyF5:
   {
    var arrayArgs = new Array();
    arrayArgs[0] = GETVALUE("SURNAME");
    arrayArgs[1] = GETVALUE("GIVENAME");
    arrayArgs[2] = GETVALUE("ADDRESS1");
    arrayRets =
window.showModalDialog("Example.htm",arrayArgs,"dialogHeight:155px
    SETVALUE("SURNAME",arrayRets[0]);
    SETVALUE("GIVENAME",arrayRets[1]);
    SETVALUE("ADDRESS1",arrayRets[2]);
    delete(arrayArgs);
    delete(arrayRets);
   }
   break;
 default:
   SENDKEY(objScriptInstance.FunctionKeyUsed);
   break;
```

# 以上はユーザーが実行できるいくつかの処理の簡単な例です(このすべて について、明示的にも黙示的にも一切保証するものではありません)。

}

# 現在の5250画面の下部に表示されているメッセージを、どのようにRAMPスクリプトに取り込むことができますか? 次のようなスクリプトを使用します。

var strMessage = GETVALUE("ActiveForm.Message"); /\* Get the message into JavaScript variable strMessage \*/

{

if (strMessage != "") ALERT\_MESSAGE(strMessage); /\* If a message
was retrieved, display it in a message box \*/
}

RA(レコードの自動前進)フィールドはどのように処理するので すか?

一部の5250アプリケーションでは、RA入力属性(レコードの自動前進)を 持つフィールドを使用できます。このフィールドを表示するプログラム では、ユーザーが最後の数値や文字を入力した時に、Enterキーが自動で 押されます。

RAMPコレオグラファは、キーストロークに基づくこのような状況のスクリプトを自動生成できません。その代わりに次のような行が生成されます。

/\* Set up data fields on form xxx \*/

#### SETVALUE("utxtMenuOption","");

/\* Send the key required to navigate to xxx \*/

生成されたスクリプトを編集して、値とEnterキーの押し下げ処理を次のように指定する必要があります。

/\* Set up data fields on form xxx \*/

SETVALUE("utxtMenuOption","2");

/\* Send the key required to navigate to xxx \*/

SENDKEY(KeyEnter);

なぜ各5250画面でF12=キャンセルボタン、F3=終了ボタンおよ びファンクション・キーを使用不可にする必要があるのです か?

Windowsアプリケーションを移動する方法を考えてみましょう。

刷新するアプリケーションで、すべての5250画面を識別して、 スクリプトを記述する必要がありますか?

いいえ。

通常、5250アプリケーションの一部のエリアはほとんど使用されない か、ほんの少数のユーザーしか使用しません。

アプリケーションのエリアに適用する刷新の度合いは、そのエリアがエ ンド・ユーザーに表示される頻度、および完全に刷新した場合に得られ るメリットの量に基づく必要があります。

サブシステムへのアクセスを提供する1つの画面の刷新

この例では、uCodeTableMaintという名前の5250メニューまたはWork with 画面が、System Code Table Maintenance (例えば、アプリケーションの定 義と制御に使用される州、企業、通貨、利率などの従来のコードとパラ メータ・テーブル)を処理する47個の異なる5250画面へのアクセスを管理 します。

uCodeTableMaintは、次のように「アプリケーション・サブシステム」と して表すことができます。



このアプリケーションのエリアは完全に刷新する必要がないので、最も 速い刷新方法は、他の画面へのアクセスを提供する1つのRAMP画面を作 成することです。 これを行うには、以下を実行します。

- Code Tablesというビジネスオブジェクトを作成し、アプリケーションに関連付けます。
- そのビジネスオブジェクトに、例えばMaintainという1つのRAMP画 面(またはタブ)を定義します。これがオブジェクト・レベルのコマン ドであることを確認し、コマンドをクリックするたびに自動的に実行 されるようにデフォルトのコマンドにしてください。
- 5250 Work with画面uCodeTableMaintを識別し、フレームワークに対して Destination画面として定義し、Maintain画面に関連付けます。

ユーザーがフレームワーク・アプリケーションのCode Tablesをクリック すると、すぐにuCodeTableMaint 5250画面にナビゲートされます。





uCodeTableMaint画面が表示されたら、ユーザーは、通常の方法で他の47 個の関連画面に移動できます。



これは、uCodeTableMaint管理サブシステム全体の最小限の刷新です。 5250画面uCodeTableMaintだけをフレームワークに定義して、スクリプト を記述する必要があります。他の47個の画面を識別またはスクリプトを 記述する必要はありません。

### RAMPツールを取得して固定セッションを割り当てる方法

「Destination画面の詳細」で、Destination画面に固定セッション (SESSION\_Aなど)を割り当てる場合、どのようにすればよいでしょう か?

コマンド・ハンドラーを選択する必要があります。このとき、チェック ボックスにチェックマークを付けるだけでなく、コマンド・ハンドラー の行が青に変わるようにします。次にセッションをコマンド・ハンド ラーに関連付けます。

これは一般的でないかもしれませんが、複数のコマンド・ハンドラーが1 つの目的フォームに関連付けられている場合があります。したがって、 セッションを変更するコマンド・ハンドラーを実際に示す必要がありま す。 複数の区画でスクリプトを動作させるにはどのようにすればよいですか?

スクリプト内でハード・コーディングされた区画の参照を以下のコード で置き換えます。

objFramework.Partition

コードはスクリプト・ポップアップ・メニューで入力できます(現在のフレームワークを選択し、区画を選択します)。

# 独自のファンクションをスクリプトに追加できますか? はい。ファンクションを画面スクリプトに追加できます。つまり、これ は他のすべての画面から隠されていて、画面オブジェクト自体のみにア クセスできます。

形式は次のとおりです。

MyFunction : function (parms) { logic } ,

#### これは、以下により呼び出されます。

this.MyFunction(parms);

<SYSINFO>ブロックの前にこのファンクションを配置します。末尾のコ ンマに注意してください。

また、ファンクションをUF\_VFSY40\_RTS,JSに追加すると、ファンク ションはすべての画面オブジェクトにアクセスできます。この参照を ファンクションに渡すことで、ファンクションは一般にユーザーの想定 どおりに機能できます。「ユーザー定義のスクリプト・ファンクショ ン」を参照してください。

## aXes5250セッションで中断メッセージを使用しない方法

aXesはIBMiの中断メッセージをうまく処理できません。これはWebブラ ウザのようなプル型技術ではサーバーからのプッシュ型の要望にうまく 対応できないからです。

特にRAMP-TSとaXes-Mobileアプリケーションの場合、aXes 5250セッションで中断メッセージを使用しないようにしてください。

ユーザーのIBM i ログオン・プログラムでIBM i のCLコマンドを使用すれば、通常は中断メッセージを使用しないように設定できます。

# CHGJOB BRKMSG(\*HOLD) STSMSG(\*NONE) MONMSG MSGID(CPF0000 MCH0000) CHGMSGQ MSGQ(\*WRKSTN) DLVRY(\*HOLD) MONMSG MSGID(CPF0000 MCH0000)

CHGMSGQ MSGQ(\*USRPRF) DLVRY(\*HOLD) MONMSG MSGID(CPF0000 MCH0000)

付録

# RAMP-TS画面の識別はどこでどのように保持されますか?

デフォルトでは、RAMP-TSは画面を /ts/screensフォルダに保管します。 各画面定義は、N.scnというファイルに保管されます(例:1.scn、2.scn、 3.scn)。これらのファイルには、画面の特性を定義するJSONフラグメン トが含まれています。

これらのファイルは開発中に必要なだけです。実行時、これらすべての ファイルはscreens.jsnという名前の1つの統合されたファイルで公開され ます。screen.jsnは、実行環境に配布される唯一の画面ファイルです。

このフォルダに関して知っておく必要がある重要な項目

- 定期的にこのフォルダ(およびサブフォルダ)を更新します。N.scn ファイルの内容にはかなりの時間が投入されています。
- /ts/Screensフォルダまたはそのサブフォルダ内のファイルをコ ピー、移動、リネーム、または削除してはいけません。
- これらのファイルの内容を変更するには、RAMP-TSエディターを 使用します。

画面定義の分割

画面定義は必ずしも/ts/screens内にある必要はありません。画面定義 を/s/screens/Project1または/ts/screens/Project2などのサブフォルダ内に分割 することができます。

フォルダ名(Project1)を入力するだけです。以下のように、このフォルダが/ts/screensのサブフォルダであることが明確になります。

-RAMP-TS (ターミナル サーハニー)				
目に有 リーハーとし U1未1子 TP アドリッ				
エノドレス	61.88.115.211			
	8080			
実行モート・ロートンパス	/ts/skins/	🦳 HTTPS を使用		
RAMPツールモート・ロート・ハウス	/ts/dev/	📄 HTTPS を使用		
プライベート定義フォルダ	Project1	── 共有オブジェ外を含む		
RAMP-TS ツールのインストールと構成をテスト				

フレームワークのRAMP-TSサーバーを定義するときは、このサブフォル ダ名をプライベート作業セットとして指定します。画面定義は、分割で きない完全に独立したプロジェクト・ベースでこのように分割できま す。

画面はコピー、移動、リネーム、または削除してはいけないため、画面 を作業ユニットや開発者ベースで分割するべきではありません。