



Valve Hammer Editor Help

Valve Hammer Editor copyright 2002 Valve, LLC all rights reserved.

[What's New in 3.4?](#) • [Setup Guide](#) • [User's Guide](#) [Troubleshooting](#) • [WWW Reference](#) • [Hotkey Reference](#)

What's New in 3.4?

Version 3.4

Features: The most notable change in this version is the change in name to Valve Hammer Editor. The following changes have also been made:

- Splitter view layout is now preserved from session to session.
- Dialog bars now pop to the top when the mouse cursor floats over them.
- No longer renders the 3D views when the editor is not the active application. (This will fix most people's problems with running Hammer and Half-Life at the same time causing the editor to freeze)
- Added Copy to Clipboard button to the process window.
- Sped up time to bring up the Face Properties dialog when many brushes are selected.
- Sped up 2D view scrolling.
- Added timed selection of objects by depth in the 3D view when the left mouse button is held down.

Fixed:

- Fixed a random spinlock when rendering sprites.
 - Fixed texture rendering problems with sprites, etc.
 - Fixed freeze when starting map with sprites visible.
 - Fixed a crash in the path tool.
 - Fixed problem with texture shifts being reset when importing old MAP files.
 - Fixed black brushes caused by clipping.
 - Fixed a lockup when scaling certain brushes.
 - Fixed infinite lines in 3D view in vertex mode.
 - Fixed clipping causing duplicate solids.
 - Fixed decal repositioning.
 - Fixed camera angle view bug.
 - Fixed problem with running compile tools from paths with spaces.
-

Version 3.3

Highlights:

The editor has been given a facelift, with a completely rewritten OpenGL renderer for the 3D views. This enables the addition of engine rendering code for previews of such things as sprites and glow effects. Texturing, normally the most time-consuming aspect of mapmaking, has also been streamlined. And, a number of other productivity-enhancing features have taken the most common hitches out of mapmaking.

- *OpenGL 3D Renderer*. The real-time renderer is much faster, more robust, and allows for animating visuals in the 3D view.
- *Texture Locking*. Texture alignment is now preserved through rotation, scaling, and flipping, allowing for the hassle-free construction of odd-angled geometry.
- *Powerful Texture Application Features*. Automatic texture continuity is guaranteed when lifting and applying from one face to another. Textures can be fit across one face or multiple faces, or aligned to the left, right, top, bottom, and center of faces with a single button click. Textures can be projected onto faces from an arbitrary viewpoint, useful for texturing cliff faces or other irregular geometry. In addition, texture axes are now displayed in the 3D view.
- *Sprite Preview*. Sprites are now automatically previewed in the 3D view, along with animation. Frame rate and scale key values are correctly reflected in the preview, eliminating the need to compile and run the map in the game engine when placing sprites.

Features:

Aside from the above highlights, there have been a number of smaller changes as well.

- *Go To Brush Number*. Brushes can now be found by entity / brush number for fast tracking down of compiler errors.
- *Iconic Entities*. Any point entity can be set to render as a sprite icon in the 3D view through a specification in the FGD file, making it easy to distinguish between different entities in the 3D view.
- *3D Preview of Tools*. The brush creation, clipper, cordon bounds, and

selection tools all render in the 3D view in real time, taking the guesswork out of previously painstaking operations.

- *Colored Solid Entities*. Solid entities can now be given a render color in the FGD file, for easier identification in the 2D or 3D views.
- *3D Grid*. A new 3D grid allows for easy visualization of brush sizes in the 3D view.
- *Missing Texture Replacement*. Added a Replace button to the texture bar, making it easy to replace textures when they are missing from the WAD file.
- *Texture Usage Report*. Statistics on unique textures, total texture memory, and WADs used in are reported in the Map Information dialog
- *Enhanced Camera Controls*. Real-time keyboard sampling gives smooth camera movement in the 3D view. Holding down both mouse buttons now enables dollying the camera forward and back. The mouse wheel now zooms both the 2D and 3D views.
- *Point and Click Entity Placement*. Point entities can now be placed directly in the 3D view just by pointing and clicking.
- *Entity Key Browse Button*. Added a browse button to the Entity Properties dialog for typo-free setting of sprite, sound, and model keys.
- *Clipper Measurements*. Sizes of clipped brushes can be displayed in the 2D view while clipping. This is useful for clipping brushes to a particular size, or for matching with a specific texture.

Valve Hammer Editor 3.4 User's Guide

Ready to jump into the exciting world of level editing? You've come to the right place. This User's Guide is designed to familiarize you with Valve Hammer Editor basics. It will introduce you to the essential elements that go into every level, and help you understand how all the pieces fit together.

General

- [Introduction to Editing](#)
- [3D and 2D Views](#)
- [Menus](#)
- [Dialogs](#)
- [Toolbars](#)

Working With Solids

- [Creating Solids](#)
- [Texturing Solids](#)
- [Reshaping Solids](#)

Working With Entities

- [Creating Entities](#)

Advanced

- [Using the Arch Tool](#)
- [Grouping and VisGrouping](#)
- [Creating and Using Prefabs](#)

see also: [Troubleshooting](#)

Introduction to Editing

Solids: the Foundation of 3D Design Blocks. Wedges. Cylinders. Spikes. They may not sound like much, but these are the basic building blocks of all architecture created in the Valve Hammer Editor. You can carve 'em, clip 'em, and manipulate 'em. You can combine these solids (also called brushes) to make any shape possible, real or imagined. This is known as constructive solid geometry (CSG) and this is the editing style Hammer uses.

Once you create a brush, you'll assign to it a texture, which is a pre-existing bitmap image created to make the brush resemble something in the real (or some imagined) world. Examples of textures include bricks, rock faces, and water.

Entities

You say you want more in your game world than inanimate solids? Well then, what you want are entities. Where brushes are "world objects" used to form the basic inanimate structure of your level, entities are the objects that move, have sound, or are interactive. An entity is anything that performs some type of operation or task within your level.

Entity Types

There are two types of entities: point-based and brush-based.

Point-based entities exist only at a certain exact point. Examples include lights, monsters and players. (Monsters **do** have an area, but this is defined by the game code and is not modifiable from within the map.) Some point entities are just that: points. For example, the env_beam entity, which controls Half-Life's beam effects, uses two point entities as targets; you place the two points and the beam of light runs between them.

Brush-based entities are entities that depend on a brush for their physical presence, like doors, trains, and other moving objects. A trigger is another type of brush-based entity; it requires that you indicate an area or activation field which controls the trigger's operation.

Putting it All Together

Using these simple components, you can create a virtually limitless variety of

levels. Whether its a barren room or a vast, complex world, you'll do it by using solids and textures to create your architecture, then adding lights, monsters, buttons, moving platforms and a host of other entities to bring your creation to life.

Once everything is in place, you will need to compile your level. This is the process that turns your collection of solids and entities into a playable level that you can run in Half-Life. Although the compiling process happens when you think you've finished your level, knowing something about this process ahead of time can save you many headaches.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

The 3D and 2D Views

Before you can begin creating and placing solids and entities, you need to familiarize yourself with the editor's interface. By default, your working area is divided into 4 sections. The first (top, left) window is the 3D View, and the rest are the 2D views.

The 3D View The 3D window is a dynamic 3D space where you can view your level from any angle. This is critical for checking textures and texture alignment, spotting leaks and just getting a sense of what your finished level will look like.

Using Cameras

To take advantage of the 3D view, you need to be able to place cameras. Cameras determine your vantage point(s). Hammer provides you with precise control over the camera movements in your map.



A camera in Hammer (as displayed in the 2D views) consists of two parts: the eye, and the viewing angle, which is represented by a line extending out from the eye. The length of the line that represents the viewing angle is not important, though it can help you aim the camera exactly at an object.

While it is possible to move a single camera all over the map each time you need to look at something new in the 3D window, it is more convenient to have easy access to multiple cameras placed throughout the map. Hammer allows you to easily cycle through multiple cameras by pressing PageUp and PageDown.

Camera Placement

Placing cameras in Hammer is extremely simple. First, switch to Camera mode by pressing Shift+C, then hold Shift and with the left mouse button, click-drag a line in one of the 2D views. This will create a thin red line with a large dot at one end. The dot is the camera's position, and this is where the 3D camera view will originate. The red line is the camera's viewing angle. You can adjust either end of the line to change the view. Follow the above steps to create as many cameras

in your level as you need.

Note: There are a number of options available to you when using multiple cameras. You must have the Camera tool selected to take advantage of these:

PageUp: cycle up to the next camera position

PageDn: cycle down to the last camera position

Delete: delete the current camera position

Shift: hold shift and click-drag with the left mouse button to create a new camera.

Tip: While in camera mode, you can adjust the camera position by moving the eye or viewing angle in any of the 2d windows.

Mouselook/NoClip Style Movement

Version 2.1 of Hammer introduced a new style of 3D View movement called mouselook movement. It is designed to be the same as when you are in the game and walking around with +mlook (mouselook) and the NoClip cheat both turned on. It can be enabled or disabled by pressing (lowercase) z.

Moving your mouse around will change the player's direction of focus, while W and S control forward and backward movement, and A and D control sided to side (left and right strafing) movement.

The old style keyboard shortcuts (listed below) still work. You can disable this new movement style by going into the 3D View options and disabling use mouselook navigation.

Keyboard Shortcuts

There are a number of keyboard shortcuts that you can use to quickly maneuver through the 3D view without switching to the Camera tool.

While holding the spacebar:

- holding the left mouse button allows you to rotate your angle of view in any direction, while the viewing point remains stationary.
- holding the right mouse button will allow you to move left, right, up, and down while keeping the viewing angle constant.

- holding the left *and* right buttons and moving the mouse causes the view to strafe forward, backward, right, and left. (note that this makes the following behavior redundant).

While holding the spacebar and Shift:

- the left mouse button acts the same as above.
- the right mouse button allows you to move forward and backward, as well as from side to side.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

Menus

For information about the Valve Hammer Editor's menus, click on the following links:

- [File](#)
- [Edit](#)
- [Map](#)
- [View](#)
- [Tools](#)
- [Window](#)
- [Help](#)

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

The File Menu

| | |
|----------------------------------|--------|
| <u>N</u> ew | Ctrl+N |
| <u>O</u> pen... | Ctrl+O |
| <u>C</u> lose | |
| <u>S</u> ave | Ctrl+S |
| Save <u>A</u> s... | |
| Export to <u>M</u> AP | |
| Export Again | Alt-B |
| Export to <u>D</u> XF | |
| R <u>u</u> n | F9 |
| 1 E:\Worldcraft3\...\testmap.rmf | |
| E <u>x</u> it | |

The File menu is mostly self explanatory - it deals with all of the editors file operations. The only commands that may require a bit of explanation are:

Export to .MAP This is used to export your map into a .MAP file. Hammer's normal map format is .RMF, but the compile tools will only read .MAP files. The .RMF file contains information like camera positions, VisGroups, and grouping information, whereas the .MAP file does not.

This will be done automatically when you use Hammer to compile your map, but if you use a batch file or external compile program to compile, you'll need to use this function.

(The same result can be achieved by using the Save As function and saving as a Game Map)

Export Again

Once you've used Export to .Map once, you can automatically export your map to the same location and name by selecting this function or by using it's hotkey - Alt+B.

Export to .DXF

This is used to export bits of architecture into a format readable by 3DStudio for use in constructing new model animations for use in scripted sequences. Hammer is *not* capable of importing work from 3DStudio.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

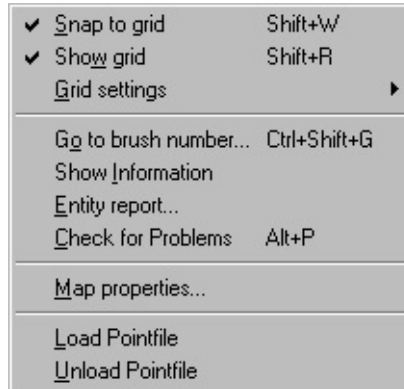
The Edit Menu

| | |
|-------------------|-----------|
| Undo | Ctrl+Z |
| Can't Redo | Ctrl+Y |
| Disable Undo/Redo | |
| Cut | Ctrl+X |
| Copy | Ctrl+C |
| Paste | Ctrl+V |
| Paste special... | |
| Delete | Del |
| Clear selection | Shift+Q |
| Select all | |
| Properties | Alt+Enter |

Everything here is pretty self explanatory. The Properties function (which is mostly handily used through it's Alt+Enter hotkey) is used to view the properties of world and entity objects.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

The Map Menu



The Map menu contains a number of functions related to the display of the map, as well as global viewing and manipulation of map contents and properties.

Grid Options *Snap to grid* and *Show grid* are fairly self explanatory. Grid Settings allow you to adjust the size of the grid (although using the [and] hotkeys let you do this much quicker.

Go to Brush Number

This brings up the [Go To Brush](#) dialog which allows you to go directly to a brush or entity by it's id number. This is handy if you're using compile tools that will report a brush/entity number along with an error (such as [Zoner's compile tools](#)). You have the option of searching through all brushes or visible brushes only - this should be set according to how the map was compiled.

Show Information

This brings up the [Map Information](#) dialog.

Entity Report

This brings up the [Entity Report](#) dialog.

Check for Problems

This brings up the [Check for Problems](#) dialog.

Map Properties

This brings up an [Entity Properties](#) dialog with the Worldspawn properties. The

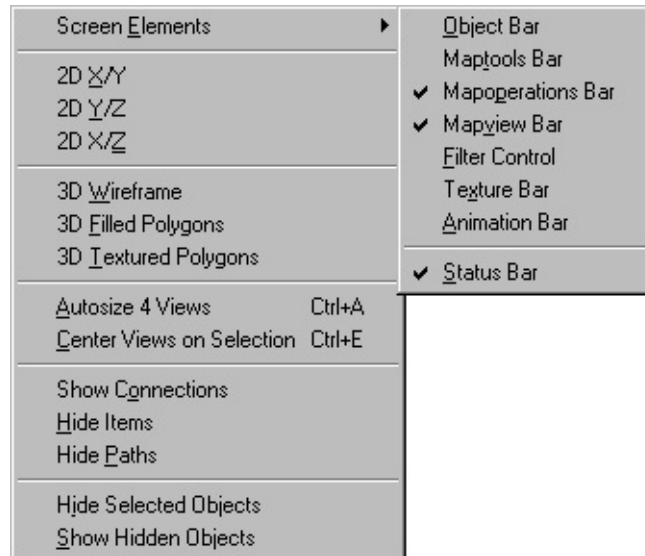
Worldspawn is an entity that defines the characteristics of the world. This is the only way to modify the Worldspawn properties.

Load and Unload Pointfile

When the compile tools report a leak in your level, you can use these functions to load the leak file directly into the editor's 2D views. For more information about finding and killing leaks, refer to the [leak fixing tutorial](#).

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

The View Menu



The purpose of the functions in this menu is to control what does and doesn't appear in the editor's 3D and 2D views. For more ways to customize the appearance of the Valve Hammer Editor, refer to the sections on the [General](#), [3D View](#), and [2D Views](#) options.

Screen Elements This sub-menu allows you to show and hide Hammer's [toolbars](#).

2D X/Y, Y/Z, X/Z

Selecting any of these will turn the currently selected view into whatever you've selected: X/Y (top), Y/Z (front), or X/Z (side) view.

3D Wireframe, Filled Polygons, Textured Polygons

Selecting any of these will turn the currently selected view into wireframe, filled polygons (flat shaded colored polygons), or textured polygons (textured as they would be in the game).

Autosize 4 Views

This function only works when you *aren't* using independent windows. The four views will be set to equal sizes based on the current available screen space.

Center Views on Selection

Choosing this function will center all of the 2D views on whatever you currently have selected.

Show Connections

Choosing this function will cause lines to be drawn in the 2D views between entities and their targets.

Hide Items and Hide Paths

These functions are used to hide items (any point entity, actually) and paths (which are created with the path tool).

Hide Selected Objects

This function is to hide the currently selected object. Doing this also places the object into a [VisGroup](#) which can then be shown or hidden using the [Filter Control](#) dialog.

Show Selected Objects

This function makes all hidden objects visible.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

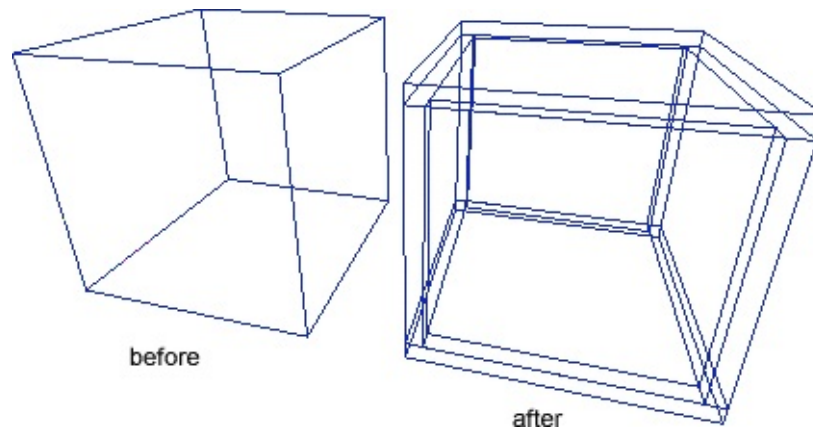
The Tools Menu

| | |
|---------------------------------|--------------|
| <u>C</u> arve | Shift+Ctrl+C |
| M <u>a</u> ke Hollow | Ctrl+H |
| <u>G</u> roup | Ctrl+G |
| <u>U</u> ngroup | Ctrl+U |
| <u>T</u> ie to entity | Ctrl+T |
| <u>M</u> ove to world | Ctrl+W |
| Texture application | Shift+A |
| R <u>e</u> place textures | |
| ✓ T <u>e</u> xure l <u>o</u> ck | Shift+L |
| Snap selected to grid | Ctrl+B |
| T <u>r</u> ansform | Ctrl+M |
| <u>A</u> lign objects | ▶ |
| <u>F</u> lip objects | ▶ |
| <u>P</u> refab Factory... | |
| C <u>r</u> eat <u>e</u> Prefab | Ctrl+R |
| <u>O</u> ptions... | |

Carve Use the Carve tool to subtract the shape of the selected object from the surrounding area. For more information, see the [Reshaping Solids](#) section.

Make Hollow

Make Hollow turns a solid object into a hollow shell of the same shape. You must have a brush selected for this to work. You will be asked to supply a value for wall thickness. Positive numbers keep the hollowed brush the same size, and negative numbers will cause it to expand.



Group/Ungroup

Group/Ungroup binds two or more objects together so they may be acted upon simultaneously. This is not the same as VisGrouping. The two functions differ in that grouping provides a way of physically binding a group of objects together, while VisGrouping enables you to organize objects into a group that may still be worked upon as separate objects. For more information, see [Grouping and VisGrouping](#).

Tie to entity/Move to world

These two commands allow you to designate an object as part of the world (a structural piece, like a wall), or as an entity (a door, for example). When you tie a brush to an entity, a dialog box pops up allowing you to choose which entity, and to modify the entity properties. This dialog will contain brush based entities only (doors, plats, triggers, etc).

Texture application

This switches the current tool to the Face Properties dialog which allows you to modify the properties of individual brush faces. See [Texturing Solids](#) for more information.

Replace textures

Replace Textures brings up a the [Replace Textures](#) dialog box which lets you search for and either mark or replace all textures of a certain type.

Texture lock

This option toggles texture lock mode on and off. Texture lock allows you to move or rotate a brush without disturbing its texture alignment.

Snap selected to grid

Not all objects will align perfectly to the grid. Some objects, such as entities, are occasionally difficult to align with the grid. This command will snap the bottom of the entity to the grid.

Transform

This brings up the [Transform](#) dialog which allows you to rotate, scale, and move an object a precise value in any of the X, Y, or Z planes.

Align objects

Align objects lets you align an object with the grid with more precision than Snap Selection to Grid.

Flip objects

Flip objects allows you to flip objects horizontally or vertically (or both, to create a mirror image of the object). Horizontal and vertical are defined relative to the currently active 2D view.

Prefab Factory...

This brings up the [Prefab Factory](#) dialog box which allows for the creation and maintenance of Hammer's prefab (.OL) files.

Create Prefab

Use this [Create Prefab](#) dialog box to turn the currently selected object into a prefab. Enter a name and description, and choose which prefab library the prefab will belong to.

Options...

This will bring up the [Options](#) dialog group which allow you to modify a number of Hammer's behaviors.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

The Window Menu

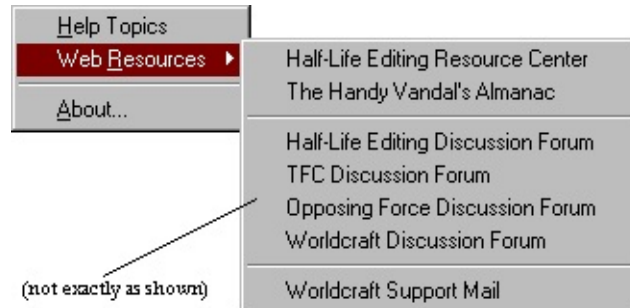


Everything here is self explanatory. The Messages function will display a window with any errors that occurred when the editor attempted to load a game data file.

The New Window function is useful for creating new windows when you're using independent window configurations.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

The Help Menu



This menu provides easy access to a number of help sources, including several websites that will provide Valve Hammer Editor and Half-Life editing information.

[Return to the Valve Hammer Editor 3.4 User's Guide](#)

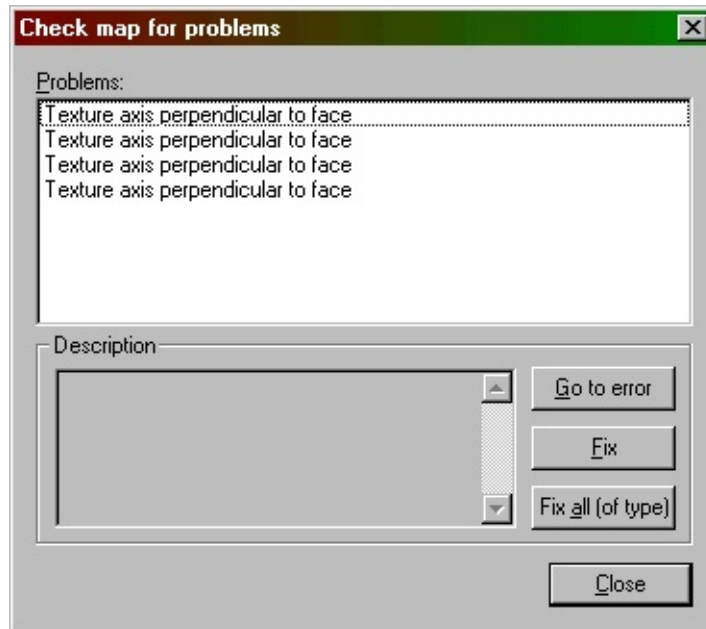
Dialogs

For information about the Valve Hammer Editor's dialogs, click on the following links:

- [Check For Problems](#)
- [Entity Report](#)
- [Face Properties](#)
- [Go To Brush](#)
- [Hollow](#)
- [Map Information](#)
- [Object Properties](#)
- [Options](#)
- [Prefab Factory](#)
- [Replace Textures](#)
- [Transform](#)

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Check For Problems



hotkey: Alt+P

Opening this dialog will cause the editor to run through your map and notify you of any errors it finds.

When you select one of the errors, the entity/solid will be highlighted (if applicable) and three options become available to you. 'Go to error' will center the 2d windows on the selected error object, 'Fix' will fix that error, and 'Fix all (of type)' will fix all errors of the selected type.

Some common errors that may occur in your maps...

Invalid solid structure

The solid has invalid structure, probably as a result of vertex manipulation. What this means is that the solid is not convex in every plane. You will need to either fix it, or if this is not possible, rebuild it.

Entity (entity_name) has unused keyvalues.

The entity contains keyvalues (variables) that are not used in its class. You can fix this error with the Fix button. Note that this error will pop up if you are taking advantage of custom compile tools which add keyvalues to entities, but

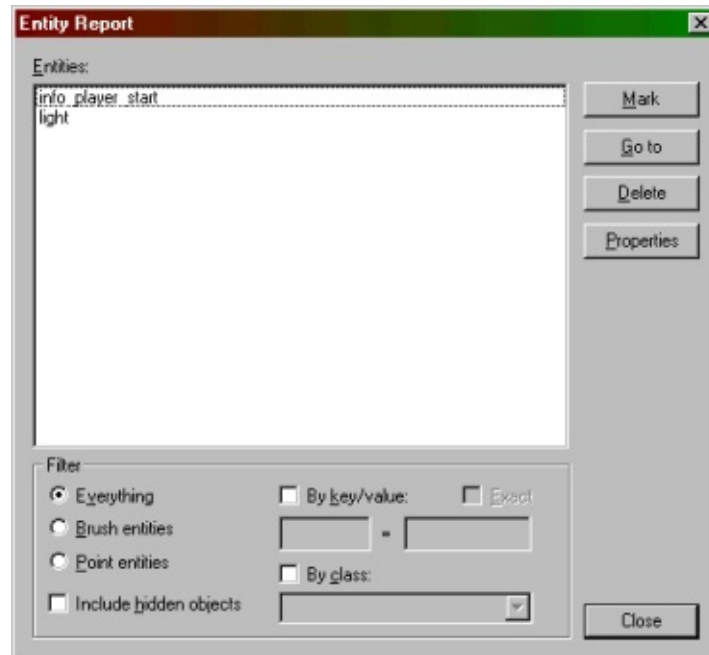
you are not using an FGD file that has these special keyvalues listed.

There is no player start.

There is no player start in the map. To add one, select the Entity tool, drag the crosshairs to the position you want the player to start at, and press Enter.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Entity Report



The Entity Report can be used to manage your entities and do quick searches for specific entities, providing an easy alternative to searching through your map manually.

Entities This box lists the entities in your level based on what is specified in the Filter section. You can select a single entity, or multiple entities by shift-clicking or ctrl-clicking. With entities selected, you can use the four buttons to the right: Mark, Go to, Delete, and Properties.

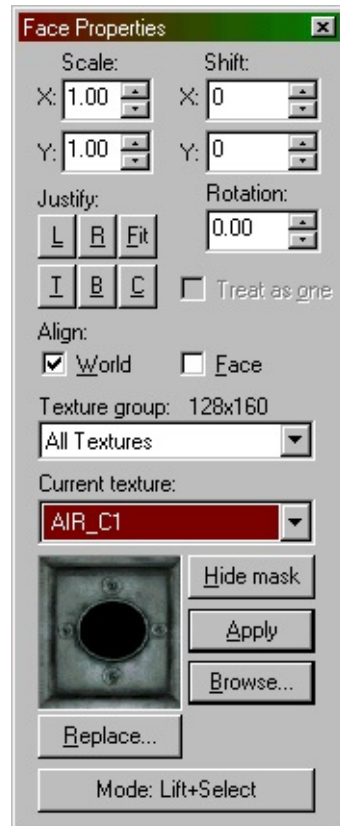
Filter

Here you can specify filters so the only certain entities are displayed. First, you can choose whether all types of entities will be displayed, or only brush or point based entities. You can also decide whether currently hidden entities should be displayed.

When searching for a specific entity or group of entities, it will be convenient to use the search by key/value or search by class. While searching by key/values, entities with partial matches will be shown unless the Exact checkbox is checked.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Face Properties



Hotkey: Shift+A

The Face Properties dialog allows you to manipulate the texture properties of selected brush faces. To select a brush face, click on it in the 3D view with the left mouse button. To select multiple faces, hold down the Ctrl key while you do this. You can also select multiple brushes *before* bringing up the Face Properties tool.

"Right-Click" Texture Application

A special feature of the Face Properties tool is that of being able to apply the current texture to a brush face by right-clicking on that face.

In order to make the textures match up from one face to the next, first align the texture on one of the faces, lift the texture from that face (left-click by default), then Alt-right-click on the other faces one by one. The texture should flow continuously across all the faces.

Scale (X/Y)

You can modify the scale of a texture to shrink or enlarge it. Numbers less than 1 shrink, and numbers greater than 1 enlarge. You can use negative values to mirror the texture in one or both planes.

Shift (X/Y)

This simply allows you to shift the texture along the horizontal or vertical texture axes.

Rotation

Rotation allows you to rotate the texture on a brush face.

Justify

There are six buttons you can press here: L (left), R (right), Fit (fit to face), T (top), B (bottom), and C (center on face). These allow you to quickly and easily align a texture on a brush.

Treat as one

If you have multiple faces selected, you can check this option to treat all the faces as one single face.

Align: World and Face

These checkboxes allow you to set the texture alignment style for the selected faces. World alignment causes textures to be aligned according to world coordinates (if you're familiar with versions of the editor before 3.3, this was the standard behavior). Face alignment uses the coordinates of the face as a basis for alignment.

Texture Group

You can select which group of textures you want to use - either all of them, or individual texture WADs.

Current Texture

The name of the current texture is displayed here, along with a thumbnail picture of the texture.

Hide Mask

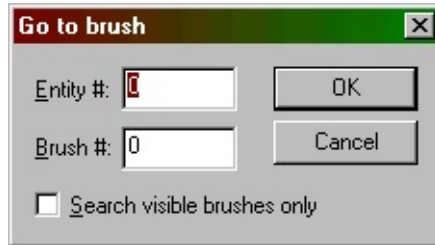
If this button is toggled on, the selected brush face(s) will not appear as selected (with a reddish hue). This may make it easier to perform some texture manipulations.

Mode

This lets you select what will happen when you click on a brush face with the left mouse button. The available options are Lift+Select, Lift, Select, Apply (texture only), Apply (texture + values), and Align to View.

- ***Lift+Select*** - Clicking on a brush face in the 3D view will select that face and make it's texture the current texture.
- ***Lift*** - Clicking on a brush face will make it's texture the current texture without selecting the face.
- ***Select*** - Clicking on a brush face will select that face without changing the current texture.
- ***Apply (texture only)*** - Clicking on a brush face will apply the current texture to that face without changing any of it's other properties.
- ***Apply (texture + values)*** - Clicking on a brush face will apply the current texture *and* texture properties.
- ***Align to View*** - This is like projecting a texture from the camera face, as if you were a slide projector and the texture is a slide. It's useful for painting a texture onto a cliff or other irregular things.

Dialogs: Go To Brush



Hotkey: Ctrl+Shift+G

When you've got an invalid brush in your map, some compile tools will give you the id number of the brush that is bad. For example, Zoner's Half-Life compile tools will give you something like this error:

Entity 0, Brush 4, Side 8: has a coplanar plane at (304, -384, 0), texture GENERIC99

To find the invalid brush in your map, you'd just bring up the Go To Brush dialog (by pressing Ctrl+Shift+G) and enter 0 as the entity number and 4 as the brush number, then press OK. The invalid brush will become selected and centered in the 2D views.

Note: if you've compiled or exported your map with the "visible objects only" setting enabled, you should enable the "Search visible brushes only" setting here as well.

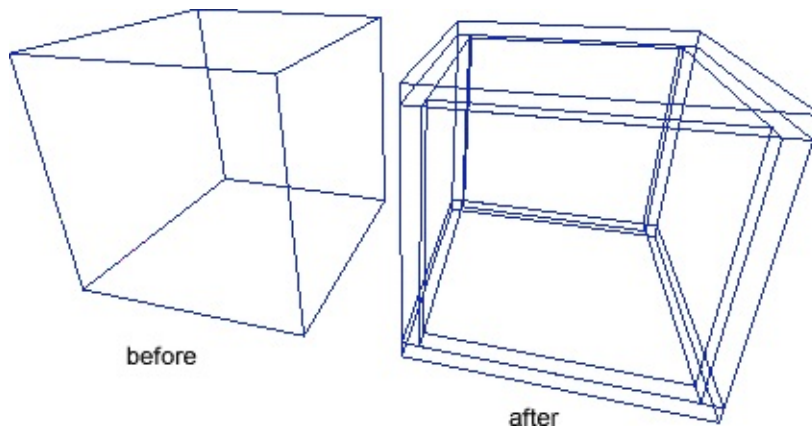
[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Hollow



Hotkey: Ctrl+H

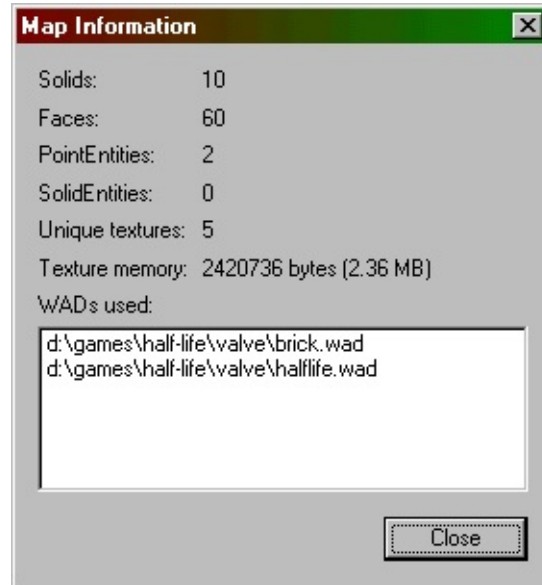
The hollow function allows you to take a block and hollow it.



To hollow outward (ie: to make the block the inside area of the hollow cube), use a negative number for the wall thickness.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

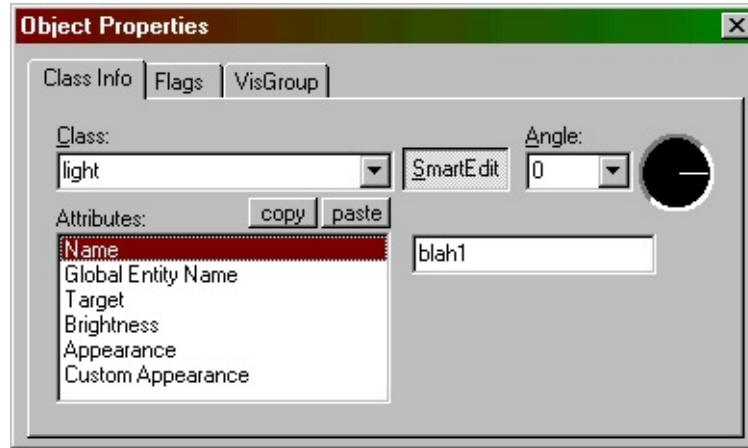
Dialogs: Map Information



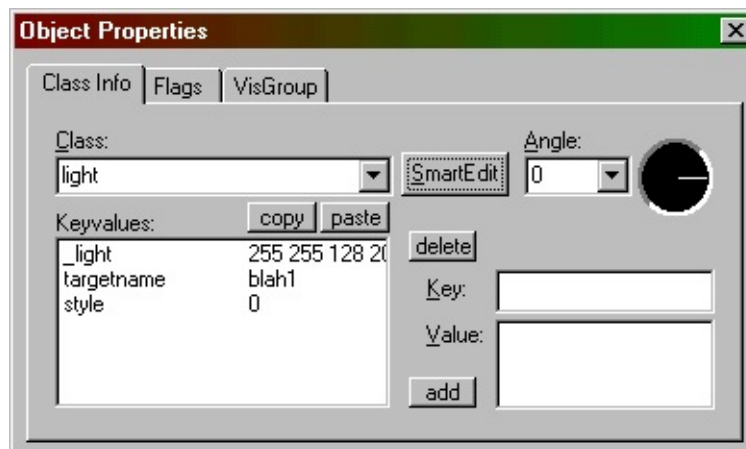
This dialog gives some statistical information about the current map.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Object Properties



regular



SmartEdit off

Hotkey: Alt+Enter

The Object Properties dialog box is the primary method of modifying entity properties. It also allows you to adjust VisGroup properties for one or many brushes.

Class Info

You can modify entity properties in the Class Info dialog. Class Info has two modes, a regular editing mode and SmartEdit (which is toggled on and off by the SmartEdit button). When you use the SmartEdit mode, all of the entity's key (entity variable) descriptions are shown to you. If you are already familiar with

the entity properties, you may find it more convenient to turn this mode off and manually insert the keyvalues. This section is only available for entities.

- ***Angle Control*** - The angle control is comprised of two parts - the Angle text box and the angle compass. These two allow you to accomplish the same thing - setting an angle between 0 and 359. The Angle text box also allows you to choose "up" and "down" which correspond to angles -1 and -2.
- ***Browsing*** - Some entity properties - those that require a path/filename of a sprite, sound, or model - will have a small button beside their property when you click on them. Clicking on this button will allow you to browse through the appropriate game directory for the sprite, sound, or model. This behavior is controlled by the game data file. For more information on that, see [FGD Format Changes](#).

Flags

You will find listed here all the available flags for an entity. This section is also only available when editing an entity's properties. A flag allows you to toggle on or off specific features of an entity.

VisGroup

This section allows you to assign the selected object to a VisGroup or remove it from one. You can also bring up the Object Groups dialog from here. For more information, see [Grouping and VisGrouping](#).

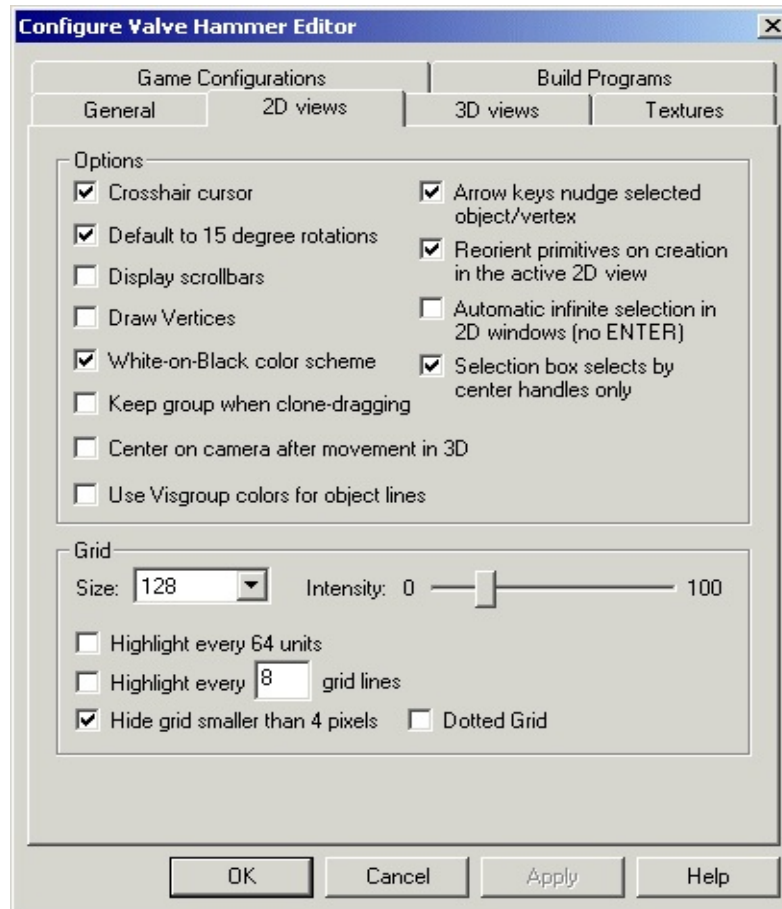
Options

For information about the Valve Hammer Editor's Option dialogs, click on the following links:

- [2D Views](#)
- [3D View](#)
- [Build Programs](#)
- [Game Configurations](#)
- [General](#)
- [Textures](#)

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Options: 2D Views



The 2D Views options allow you to configure a number of the editor's behaviors. Settings here will depend largely on personal preference.

Options

- ***Crosshair cursor*** - when this is enabled, the cursor will turn into a crosshair when it is over a brush.
- ***Default to 15 degree rotations*** - when rotating a brush, the rotations will be in 15 degree increments rather than 1 degree.
- ***Display scrollbars*** - if this is enabled, each 2D view will have a set of scrollbars to allow you to move the view around. If this is not enabled, you can still move around the 2D views by holding down the Space bar and

clicking and dragging inside a 2D view with the mouse.

- **Draw vertices** - when enabled, brush vertices will be drawn in the 2D views.
- **White-on-black color scheme** - Brushes will appear as white lines drawn on a black background. When this is not enabled, the reverse is in effect.
- **Keep group when clone-dragging** - Select this feature if you want newly cloned objects to retain grouping properties of the original object when clone dragging (selecting an object, holding down shift, and dragging the object).
- **Center on camera after movement in 3D** - Use this option if while using the mouse to move around the 3D view while in camera mode, you want the 2D views to automatically center on the new camera position.
- **Use VisGroup colors for object lines** - When this is enabled, the object lines for any non-entity brush will be the color specified in its [VisGroup](#) when viewed in the 2D views. Objects not associated with a visgroup will still appear white.
- **Arrow keys nudge selected object/vertex** - When this is enabled, you can move a selected object or vertex by using the arrow keys. The movement is performed relative to the current active 2D view.
- **Reorient primitives on creation in the active 2D view** - When this is enabled, primitives will be oriented with their "top" shown in whichever 2D view they were created in. This is most easily shown when creating a cylinder. With this option disabled, the cylinder will be oriented so that the round part shows up in the xy 2D view, regardless of which view you created it in. This is mostly just a time saving feature.
- **Automatic infinite selection in 2D windows (no Enter)** - When this is enabled, dragging a selection box over an object or objects will immediately cause them to be selected (normally you would need to press Enter). This option can be turned on and off from the MapOperations toolbar as well.
- **Selection box selects by center handles only** - When using a selection box to select a number of objects, it defaults to selecting any object the box touches. With this option enabled, it will only select those objects whose "handles" (the small "x" at the center of each object) are within the selection box.

Grid

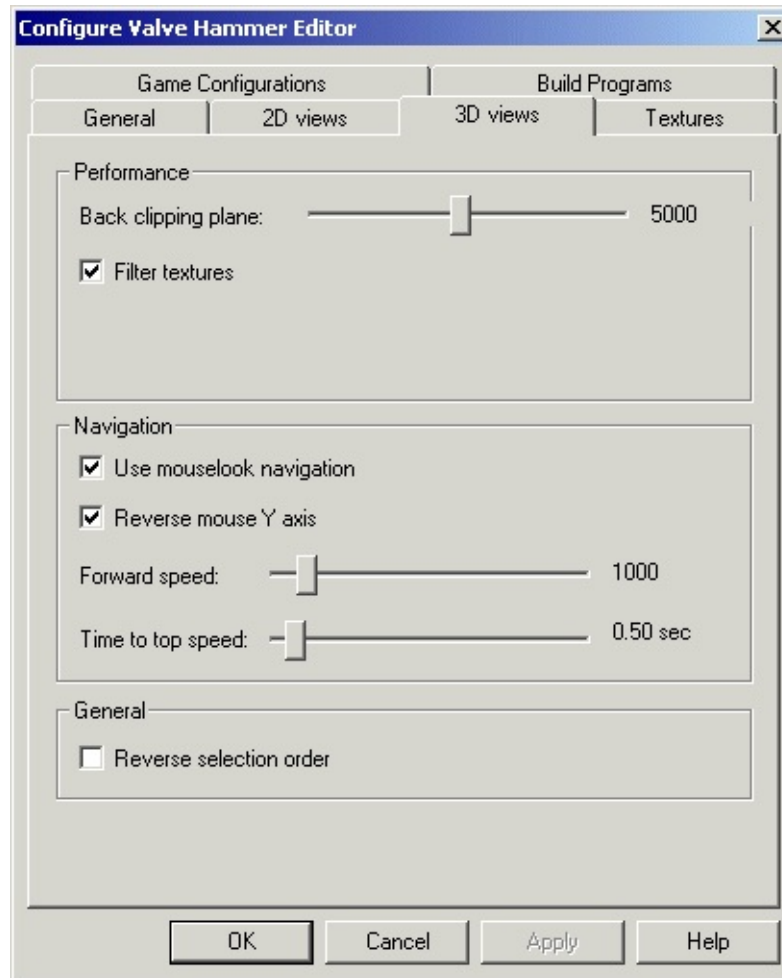
- **Size** - This determines the initial grid size when you start a new map. The

options are 8, 16, 32, 64, 128, and 256 units. You can change the grid size while editing by pressing "[" and "]".

- **Intensity** - This is a slide bar from 0 to 100. It sets the intensity (brightness) of the grid lines.
- **Highlight every 64 units** - This highlights the grid every 64 units.
- **Highlight every 8 grid lines** - This will highlight one line every 8 grid lines. The actual number can be set to whatever you like.
- **Hide grid smaller than 4 pixels** - If the grid size is smaller than 4 pixels, it will generally appear as a gray background instead of a grid, unless zoomed in extremely close. When enabled, this option turns off the grid when it gets that small.
- **Dotted grid** - Enabling this option causes the grid to be displayed as a series of dots, rather than lines. This may help if you find that the grid lines blend into your map lines.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Options: 3D View



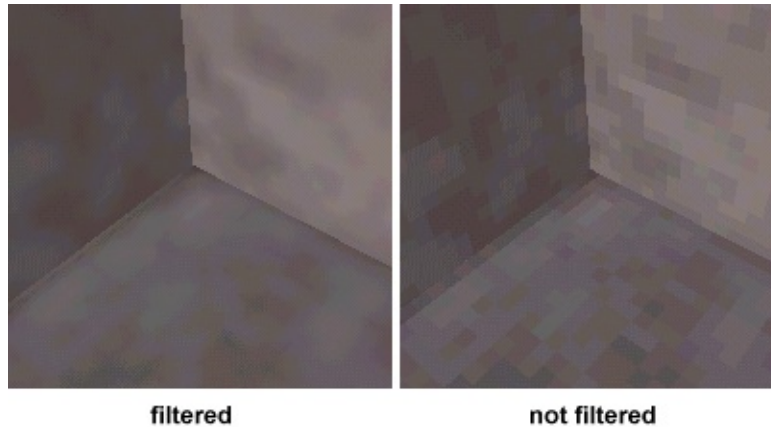
The 3D Views options allow you to set the behavior of the editor's 3D views.

***Note:** if you are familiar with previous versions of the editor, you may note the absence of the Hardware Acceleration checkbox. The Valve Hammer Editor will now always attempt to run in OpenGL mode, and if that isn't available through hardware, it will attempt to emulate it in software.*

Performance

- **Back clipping plane** - This sets the distance in units at which the editor will stop drawing in it's 3D view. If you are finding the 3D performance to be too slow, set this to a lower value.

- ***Filter textures*** - when this feature is enabled, bilinear filtering is done on textures resulting in a smoother non-pixelated look when viewed closeup. Disabling this may result in a slight speed increase, or a dramatic speed increase on older video cards.



Note: if you disable texture filtering, this is known to cause texture swimming and loss of texture perspective correction. This generally occurs with older video cards.

Navigation

- ***Use mouselook navigation*** - When this is enabled, "mouselook navigation" will be available. The z hotkey toggles this mode on and off. While on, moving the mouse will change the viewing angle in the 3D view, and the WASD keys can be used for forward, left, backward, and right movement.
- ***Reverse mouse Y axis*** - When you are using the mouse to navigate through the 3D view, if this is enabled pulling the mouse backwards will cause the view to look upward, and moving the mouse forward will cause the view to look downward.
- ***Forward speed*** - You can set the maximum forward movement speed here.
- ***Time to top speed*** - This is the amount of time it takes to go from 0 to the above maximum movement speed (basically this sets the acceleration).

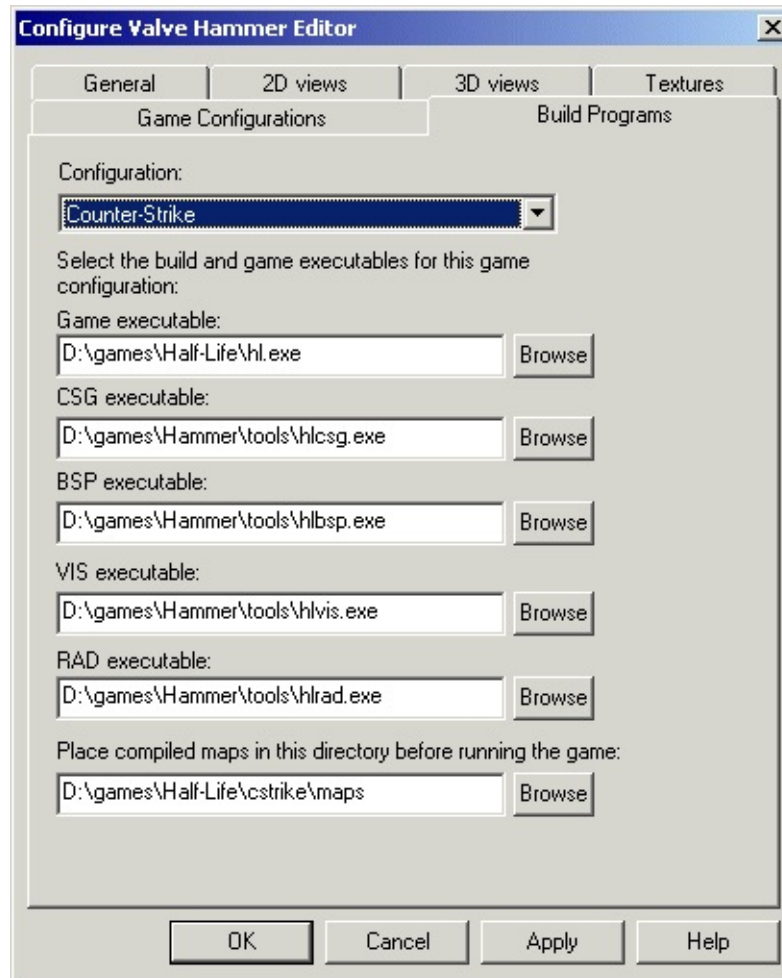
General

- ***Reverse selection order*** - User's of some video cards may find that when selecting objects in the 3D view, the furthest object will get selected rather than the one you clicked on. If you experience this problem, enable this to

reverse the selection order.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Options: Build Programs



The Build Programs options allow you to setup configuration specific compile programs.

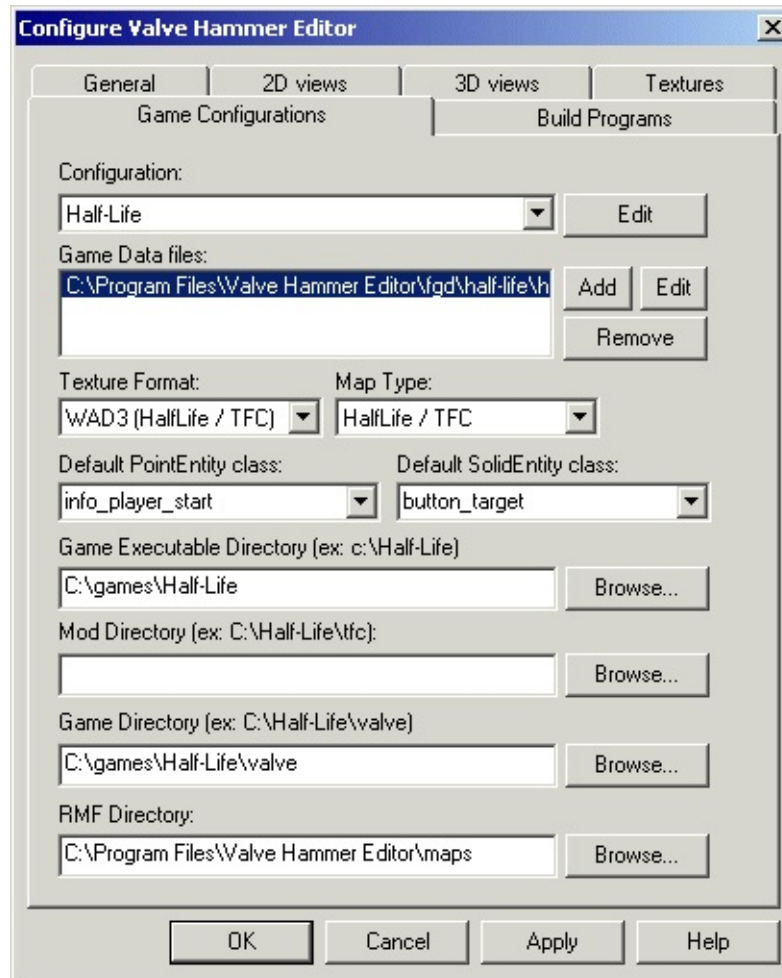
- **Game executable** - This should be the Half-Life executable file (hl.exe). Simply browse to your Half-Life folder, select hl.exe, and press the Open button. The value entered here links to the \$game_exe variable in the advanced compile mode.
- **CSG executable** - The CSG program should be specified here. The normal Valve version is qcsg.exe. In the example above, Zoner's compile tools are used, so it is hlcsq.exe. The value entered here links to the \$csg_exe

variable in the advanced compile mode.

- ***BSP executable*** - As above, for the BSP program. The value entered here links to the \$bsp_exe variable in the advanced compile mode.
- ***VIS executable*** - As above, for the VIS program. The value entered here links to the \$vis_exe variable in the advanced compile mode.
- ***RAD executable*** - As above, for the RAD (lighting) program. The value entered here links to the \$light_exe variable in the advanced compile mode.
- ***Place compiled maps in this directory before running the game*** - The value entered here links to the \$bspdir variable in the advanced compile mode. In the normal compile mode, this indicates where the compiled map will be placed.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Options: Game Configurations



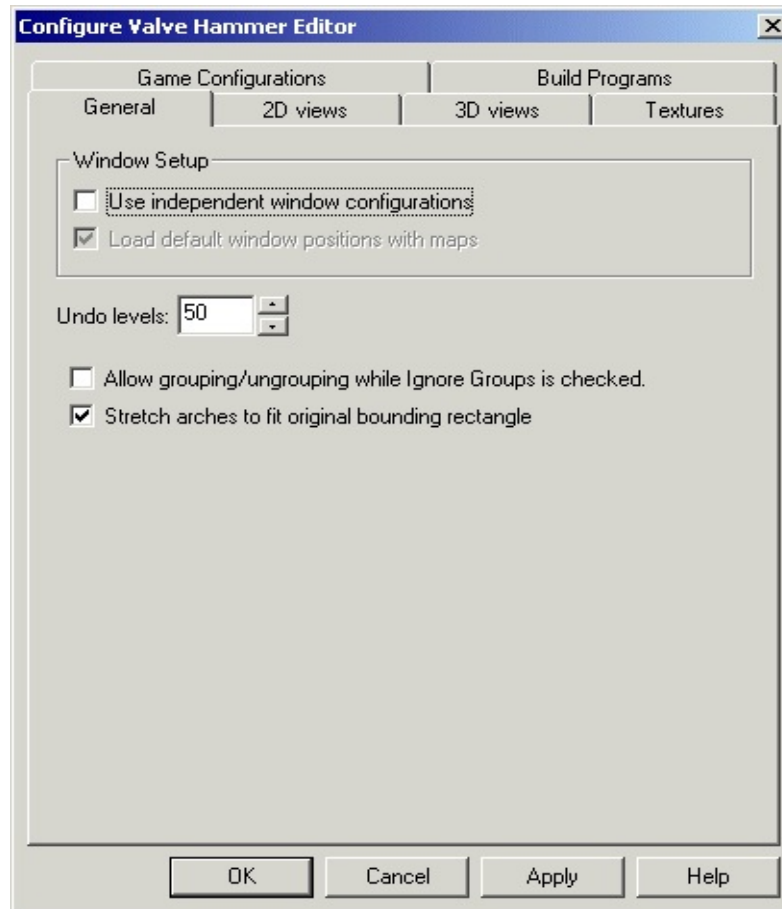
The Game Configuration options allow you to setup the editor to edit Half-Life and its mods. A separate configuration is needed for each game or mod.

- **Configuration** - if this is a new installation of the Valve Hammer Editor, this drop-down menu will be empty, otherwise your different game configurations will be listed here. If you've installed over a previous version of the editor, be sure to go through each configuration and make sure all of the settings are still correct.
- **Edit button** - clicking on this brings up a small dialog that allows you to create, rename, and delete game configurations. If this is a fresh install,

you'll need to at least create one game configuration before you can use it.

- **Game data files** - every game configuration needs at least one game data file. The game data file contains all of the entity information for a game or mod. Except in special cases, you should only be using one game data file. The buttons beside this text box are used to add, edit, or remove game data files.
- **Texture format** - You only have one choice here, the WAD3 texture format, which Half-Life and its mods use.
- **Map type** - Again, you only have the one choice here - the Half-Life map format.
- **Default Point- and SolidEntity class** - The settings here depend largely on personal taste - they define the default point and solid entities. If unchanged, they will be set to the first point and solid entity found in the game data file.
- **Game executable directory** - This should be set to the folder that the Half-Life executable is in (ie: c:\Half-Life). The value entered here links to the \$exedir variable in the advanced compile mode.
- **Mod directory** - This should be set to the mod directory. Going with the above example for the executable directory, if you were setting this up for TFC, the value here would be c:\Half-Life\tfc. The value entered here links to the \$moddir variable in the advanced compile mode.
- **Game directory** - This should be set to the default game directory - ie: c:\Half-Life\valve. The value entered here links to the \$gamedir variable in the advanced compile mode.
- **RMF directory** - Here you should *specify* the default map directory for this game configuration. When you save a map that uses this game config, it will be saved here.

Options: General



The General options let you configure miscellaneous Valve Hammer Editor Options.

Window Setup

- ***Use independent window configurations*** - The editor's Independent Window Configuration is simple to use and extremely handy. It lets you customize your windows setup virtually any way you'd like. To add new windows, select New Window from the Window menu. Windows can be resized any way you like
- ***Load default window positions with maps*** - if this is checked, the last used window setup will be loaded when you load a map. If this is not checked,

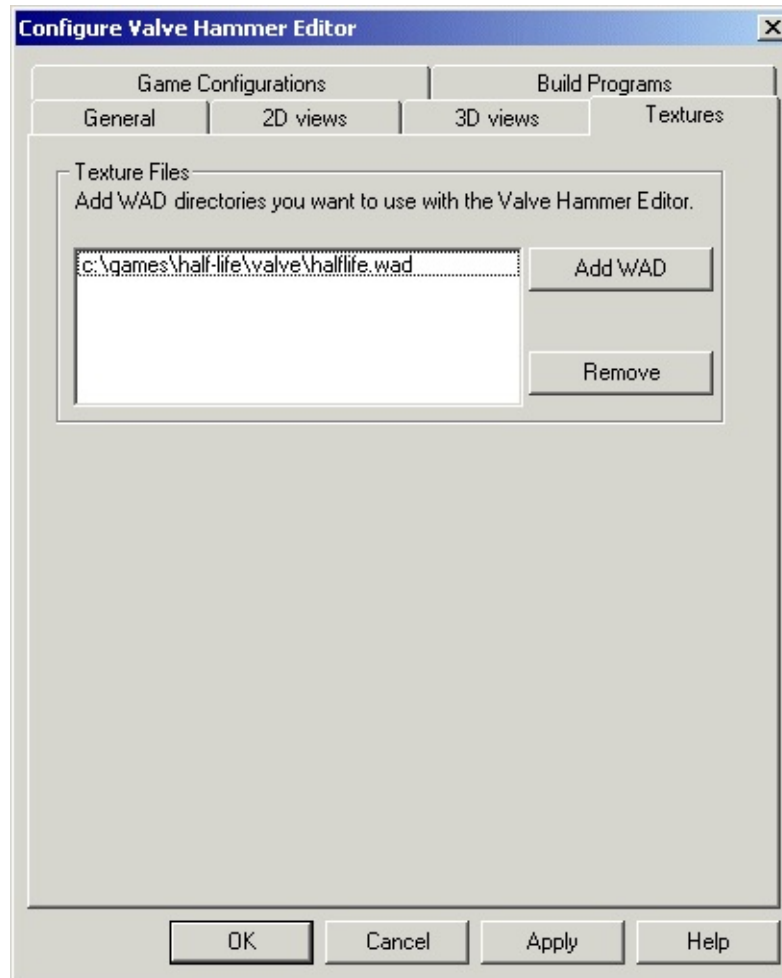
you can still load the window setup manually using the Load Window State button on the [Map View](#) toolbar.

Miscellaneous

- ***Undo levels*** - the undo levels default to the last 50 actions. This can take up quite a bit of memory. Setting this to a lower value may improve performance slightly.
- ***Allow grouping/ungrouping while Ignore Groups is checked*** - when Ignore Groups is toggled on, this setting will allow you to group and ungroup objects.
- ***Stretch arches to fit original bounding rectangle*** - when this is set, arches will fill the entire bounding box that you've defined, otherwise they will treat the bounding rectangle as a "full circle" area and only take up a portion of that area.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Options: Textures



Add WAD This will bring up a dialog box from which you can select a texture WAD file.

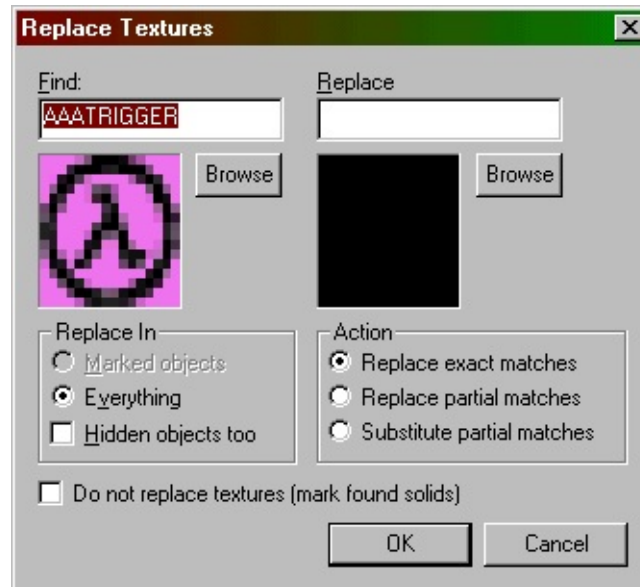
***Note:** When adding texture WADs, do not add `cached.wad`, `gfx.wad`, `pldecals.wad`, or `spraypaint.wad` - these are not meant to be used for map making.*

Remove

Remove will cause the selected texture WAD to be removed from the list. This may be useful if you are running low on memory.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Replace Textures



Find/Replace These text boxes and texture pictures show the texture to be found in the map, and the new, replacement texture. You can browse through the entire texture library to choose the "find" texture, the "replace" texture, or both.

Replace In

The Replace In section lets you specify what should and shouldn't get replaced. You can replace everything, or only marked objects, and you can choose to include or exclude hidden objects.

Action

This lets you choose which search-and-replace method you would like to use...

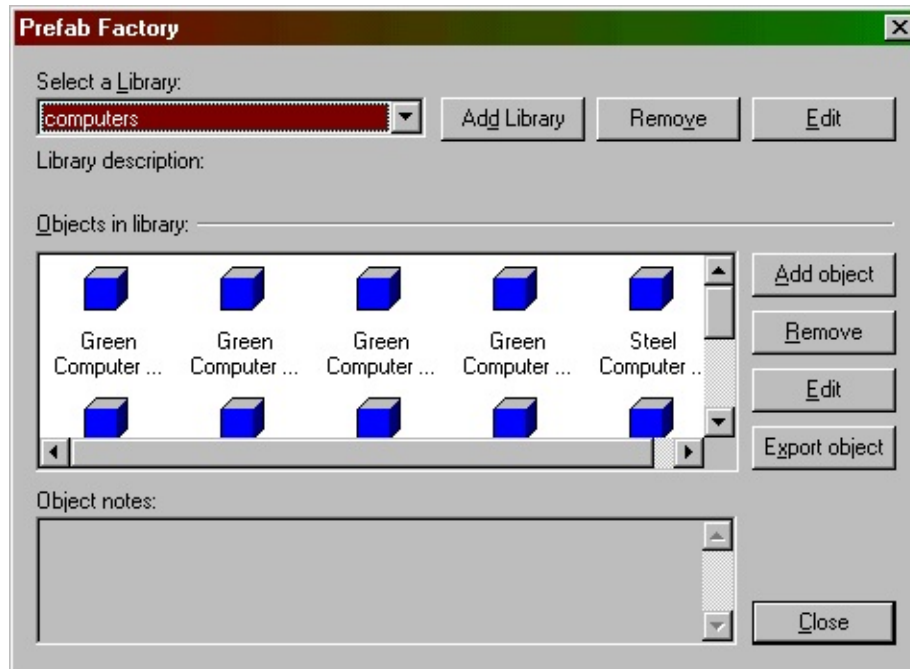
- Replace exact matches
- Replace partial matches
- Substitute partial matches - when this is enabled, you can search for a specific part of an entity name, and substitute that part with something else. For example, if you have a bunch of slime textures, ("slimedoor", "slimefloor", etc) and you want to change them to "rustdoor", "rustfloor", etc, you can enable this, search for "slime" and replace with "rust".

Do not replace textures (mark solids found)

Use this feature to simply mark (select) faces containing the given texture, instead of replacing them with another texture. This is extremely useful if you would like to modify the texture attributes of every texture of a certain type, but don't want to select each one manually (while you are in the Face Properties dialog).

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Prefab Factory



Much like the [Entity Report](#) dialog lets you manage entities, the Prefab Factory allows you to easily manage your collection of [prefab libraries](#), including adding and deleting objects and editing existing prefab objects.

Select a Library This allows you to select the library you want to work on at the moment. The three buttons to the right work directly on the selected library. Add Library creates a new empty library. Remove deletes the library from your harddrive. Edit allows you to edit the name and description of the library. The description of the currently selected library is displayed under the library selection box.

Note: Prefab libraries are stored in your Valve Hammer Editor "prefabs" directory. You can manually add and delete them by adding or removing prefab files (.OL files) in this directory.

Objects in library

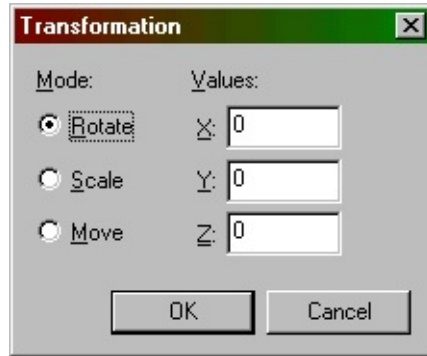
This displays the objects available in the currently selected library and allows you to work on them using the four buttons to the right. Add object brings up a browse box and lets you pick from .MAP and .RMF files to add to the library as

prefab objects. Remove deletes the selected object from the library. Edit allows you to edit the name, description and the map data of the currently selected object. Export object will let you export the currently selected object as a .MAP file.

When you select an object, its description, if any, will be displayed in the Object Notes section.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Dialogs: Transform



Hotkey: Ctrl+M

The Transform dialog gives you precise control over a selected object.

You have the option to rotate, scale, or move the selected object in any of the x, y, or z planes. Values entered are in units, except in the case of rotation, where degrees are used.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

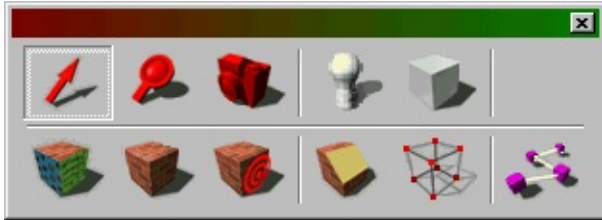
Toolbars

For information about the Valve Hammer Editor's Toolbars, click on the following links:

- [Map Tools](#)
- [Map Operations](#)
- [Map View](#)
- [Textures](#)
- [Filter Control](#)
- [New Objects](#)
- [Status Bar](#)

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Toolbars: Map Tools

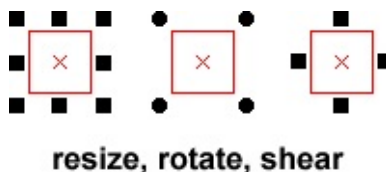


The Map Tools toolbar lets you switch between the different editing modes. This toolbar can safely be closed as each button also links to a hotkey.



Selection - (Shift+S) The selection tool allows you to accomplish a number of things. To simply select an object, click once. Continue clicking on the selected area to cycle through the three basic transformation modes (resize, rotate, and shear).

- Resize mode allows you to grab any corner or edge of the selected object and resize it by pulling it to the desired size.
- In rotate mode, you can grab any corner of a selected object and pull it around to rotate the object into any position you wish. If you have the rotations set to 15 degrees by default (this is set in the 2D Views options), this can be overridden by holding shift while you rotate the object.
- Shear mode lets drag the edges of an object parallel to its surface. For example, if you grab the top edge of a selected object while in shear mode, you would be able to drag that edge right or left.



Tip: You can clone a selected object by holding down the shift button and click-dragging the object. This can be done regardless of what selection mode the object is in (resize, rotate, or shear). See [Creating Solids](#) for more information.

Note: While in selection mode, you can select multiple objects by dragging a selection box around them and pressing Enter. If Auto-Selection is toggled on, pressing Enter is not necessary.



Magnify - (Shift+G) This tool allows you to increase the magnification factor of the 2D views. Zoom back out again by clicking the right mouse button. You can also zoom in and out by pressing the number keys while the mouse cursor is in a 2D view, or by using the + and - keys.



Camera - (Shift+C) This tool allows you to place and modify cameras within your level. While in camera mode, hold shift and left-click-drag a line to create the camera. A line with a dot (the view point) will be drawn in the 2D views, where you can adjust it until it is pointing in the direction you'd like. To place more than one camera in a level, just repeat the above steps.

Cameras are good for helping you visualize the creation of your level. They are especially useful when using the Vertex Manipulation tool as they allow you to see exactly what you are doing to a brush. You will find multiple cameras especially useful if you are working on several areas of a level at once. Placing a new camera in an area negates the need to search for that area when you want to work on it again.

When you're in camera mode, you can cycle through all available cameras by pressing the PageUp/PageDn keys. To delete the current camera, press Delete. More information can be found in [The 3D and 2D Views](#).



Entity - (Shift+E) This tool allows you to place point-based entities in a map. (Point-based entities are those entities which exist only at a point, and do not rely on a brush for their effect) Once selected, the [New Objects](#) dialog will contain a list of entities from which to choose. Click on one of the 2D views and press Enter to place the entity. If Snap To Grid is enabled, the new entities will be snapped to the grid.

Note: In order to place a brush-based entity (a func_door for example) you must first create the brush with the Block Tool (see below), then select it, and press Ctrl+T. You will be given a list of entities to choose from which will be different than those shown with the Entity Tool.



Block - (Shift+B) This is the basic creation tool available to you. It allows you to create any of the different types of primitive shapes. For more information, see [Creating Solids](#).



Texture Application - (Shift+A) This toggles the [Face Properties](#) dialog on and off. This dialog allows you to edit the texture properties of individual brush faces. For more information, see [Texturing Solids](#).



Apply Current Texture - (Shift+T) When pressed, the current texture is applied to every face of the selected brushes. For more information, see [Texturing Solids](#).



Apply Decal - (Shift+D) A decal is a texture that can be placed on top of another texture. Common examples include scorch marks or bullet holes. Use the Apply Decal button to place a decal in the 3D window. Decal textures can be viewed and selected in the Textures window. Half-Life's decal textures are designated by the use of a { as the first character in the texture name.



Clip - (Shift+X) Clipping allows you to slice the currently selected brush. You have the option of only splitting the brush, or actually slicing a piece right off. For more info, see [Reshaping Solids](#).



Vertex Manipulation - (Shift+V) Vertex manipulation gives you complete control over the shape of a brush. You can reshape the brush by manipulating individual vertices and brush edges. For more information, see [Reshaping Solids](#).



Path - (Shift+P) This tool greatly simplifies the creation of paths for trains and monsters. It allows you to create dynamic paths that you can add and delete nodes to.

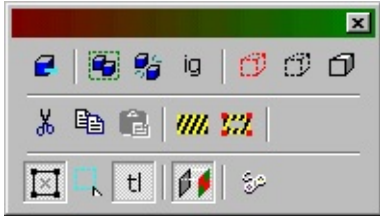
With the Path Tool selected...

1. Hold Shift and the left mouse button, and drag a line in a 2D view. This creates the first two path points.
2. A dialog box will pop up where you can set the name of the path, the path direction (one way, circular, or ping-pong), and the entity type (typically path_corner or path_track).
3. To add new points to the path, click on one of the existing path points, hold Ctrl and drag with the left mouse button. Do this as needed until you have your path complete.

note: The start of the path is denoted by the point with the green circle around it.


[Return to the Valve Hammer Editor 3.x User's Guide](#)


Toolbars: Map Operations




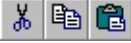
The Map Operations toolbar allows you to perform a number of operations on objects in your map as well as setting default behavior for some mapping properties.

 **Carve** - When the Carve button is pressed, the selected brush will subtract its shape from the non-selected brushes around it.

 **Group/Ungroup** - Group/Ungroup binds two or more objects together so they may be acted upon simultaneously. This is not the same as VisGrouping. The two functions differ in that grouping provides a way of physically binding a group of objects together, while VisGrouping enables you to organize objects into a group that may still be worked upon as separate objects. For more information, see [Grouping and VisGrouping](#).

 **Ignore Groups** - Ignore Groups allows you to modify individual brushes that are part of a group or entity by temporarily disabling "groups."

 **Hide Selected/Unselected, Show All** - Along with the [Filter Control](#) dialog, these buttons are used to control what does and doesn't appear in the 2D and 3D space of the editor. Hide Selected and Hide Unselected both create new VisGroups with the newly hidden objects, and Show All will make visible any hidden VisGroup. For more information on VisGroups, see [Grouping and VisGrouping](#).

 **Cut, Copy, Paste** - These buttons follow the standard windows interface for cut/copy/paste.

 **Toggle Cordon State, Edit Cordon Bounds** - These buttons allow you to

"cordon off" an area of your map. When the cordon state is toggled on, a thick red line will appear around that area. When you compile, only the inside of that red area will be compiled, and a box will be placed around the area to seal leaks.

This is extremely useful when you are working on a large level and want to look at changes made in only a small area. Rather than wait a considerable amount of time for the entire map to compile, you can selectively compile only a small area by using the Cordon functions.

Note: Be sure to place a temporary player start within the bounds of the cordoned area before you test compile that section.



Toggle Select-by-Handles - When this option is toggled on, you will only be able to select objects by their center handles in the 2D view. To select objects by their center handles in 3D view, you must also be in wireframe mode. This is useful when there are several brushes overlapping in the 2D view, making the standard selection method (clicking on brush lines) awkward and inaccurate.



Toggle Auto-selection - When Toggle Auto-selection is toggled on, you can select multiple objects by simply clicking your mouse button and dragging a selection box around the objects. When this option is toggled off, you can also select multiple objects by dragging a selection box around the objects and then pressing Enter.



Texture Lock - This button toggles texture lock mode on and off. Textures will then stay correct when you move or rotate a brush.

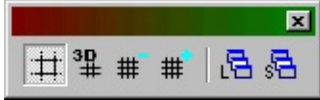


Align to World/Face - This button toggles the texture alignment style between world alignment and face alignment. World alignment will align textures according to the world grid coordinates. Face alignment will align textures according to the brush face.







Run Map - This button brings up the current compile dialog. For more information on compiling you map, see [Compiling and Running Your Level](#).



Toolbars: Map View



This toolbar controls some of the view settings for the Valve Hammer Editor.

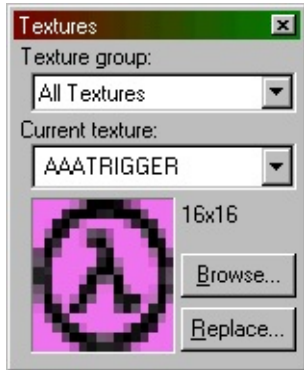
  **Toggle Grid, Toggle 3D Grid** - These toggle the grid on and off in the 2D and 3D views.

  **Smaller Grid, Larger Grid** - These decrease and increase the grid size. The [and] hotkeys can also be used for this purpose.

  **Load, Save Window State** - These buttons are used to save and restore window configurations when the independent window configuration is being used. Independent window usage can be enabled or disabled in the [General options](#).

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Toolbars: Textures toolbar



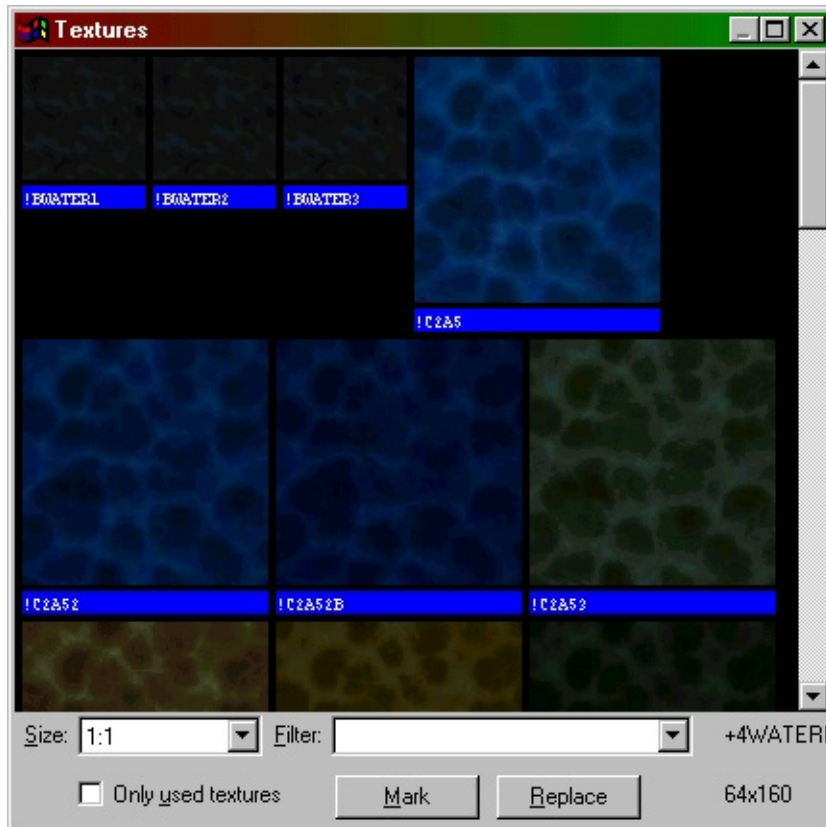
The Textures toolbar is used to select the current texture.

Texture group - The texture group pull-down menu lets you choose either All Textures, or individual texture WADs. This affects which textures will be displayed when you Browse.

Current Texture - This lists the name of the current texture and below that, a small picture of the texture. The pull-down menu lists the most recent texture choices.

Browse - This brings up the Texture Browser, pictured below.

Replace - This will bring up the [Replace Textures](#) dialog.



Texture Browser properties The texture browser has a number of properties that make texture handling easier.

- **Size** - This lets you set the display size of textures - 1:1 (actual size), 32x32, 64x64, or 128x128.
- **Filter** - You can enter substrings in this text box. Only textures whose names contain the substring will be listed.
- **Only used textures** - if you enable this, only previously used (in the current map) textures will be displayed. You can combine this with the filter substrings.
- **Mark** - click on this to mark faces that have the selected texture on them.
- **Replace** - this brings up the [Replace Textures](#) dialog.

Toolbars: Filter Control



All VisGroups will be listed in this control toolbar.

VisGrouping is a powerful tool for limiting visible objects in the map. This allows for much easier editing on large levels but does require that you use the .RMF file type for saving your maps since the .MAP format doesn't include the VisGroup information.

VisGroups are created using the [Hide Selected and Hide Unselected](#) buttons on the [Map Operations](#) toolbar. When a VisGroup is created, it is inserted into the Filter Control dialog with a name of "n objects", where n is the number of objects in the VisGroup. The name can be changed by clicking on the group, then clicking on it again (but not double-clicking!).

There are a number of buttons that can be used in the Filter Control dialog:

- **Apply** - Will cause unselected VisGroups to become invisible in the 2D and 3D views. Select a group by clicking just to the left of its name. A small hand will appear.
- **Edit** - This allows you to edit the name and color of a VisGroup, as well as add and remove new groups.
- **Mark** - This causes selected VisGroups to be highlighted in the 2D and 3D views.
- **Purge** - This will remove any groups from the VisGroup window that no longer contain any objects.

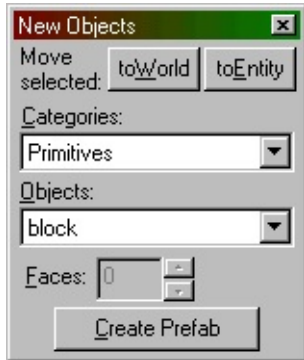
Note: You can delete a VisGroup from the list by dragging it out of the filter control dialog window.

Another feature of the Filter Control window is the ability to merge two VisGroups. This is done by dragging one group on top of a second. Hammer will ask if you want to combine the two groups, and if you do, they will be merged and the first group will take on the VisGroup properties (name, color) of the second group.

For more information on VisGroups, see [Grouping and VisGrouping](#).

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Toolbars: New Objects



The New Objects dialog controls all the facets of creation within Hammer. Entities, prefabs, brushes - the creation of all this can be controlled from here.

- **toWorld** (Ctrl-T) - toWorld will remove any entity settings of the selected solid or group of solids.
- **toEntity** (Ctrl-W) - This will attach the currently selected solid or group of solids to an entity.
- **Categories** - When the Entity Tool is active, no categories are available. With Block Tool selected, there will be at least one category available: the "Primitives", which contain the editor's five basic solid shapes: block, cylinder, spike, wedge, and the arch tool. If any Prefab libraries have been defined, then the library names will also be listed here.
- **Object** - This list contains all objects that belong to the chosen category. For entities, the list contains all entity classes available. For solids, if the 'Primitives' category has been selected, then this list contains the five types of primitives. If a Prefab library has been selected, then this list contains all objects in the library.
- **Faces** - This text box becomes active if the Block Tool is active, and either 'cylinder' or 'spike' is the current solid type. The value (which can be changed by either the keyboard or clicking the LEFT MOUSE BUTTON on the little arrows) represents the number of side faces the cylinder/spike will be created with.
- **Create Prefab** (Ctrl-R) - This will let you turn a selected object into a prefab object which will be easily reusable. When you press this button, a

Create Prefab dialog will pop up. Here you can enter the name and description of the prefab, as well as choose which library it will be placed in.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Toolbars: Status Bar

The status bar contains a number of important pieces of information. Each bit of information has it's own little area on the bar. The areas are as follows:

- ***selection information*** - The first bit of information on the status bar is about what is currently selected. If you've got a non-entity brush selected, you will see something like "solid with 6 faces". If you've got an entity selected, you will see something like "func_breakable" or "name1 - func_breakable". (The second example "name1 - func_breakable" - will be shown if you've selected an entity that has a name).
- ***coordinates*** - Follow the selection information, you'll see something like "@32, 144". These are the coordinates of the mouse cursor in whichever 2D view it's currently in.
- ***selection size information*** - When you select a brush or entity, it's size will be listed here. You'll see the width, length, and height values listed as something like "32w 128l 64h".
- ***zoom size*** - If a 2D view is currently in focus, it's zoom value will be shown next.
- ***grid information*** - The last box displays something like "Snap: On Grid: 16". This means that Snap To Grid is enabled (Shift+W to toggle it) and the current grid is in 16 unit increments (Use [and] to decrease and increase the size).

Creating Solids

Creating Solids from Scratch Creating solids is quite simple. The process always follows the same four basic steps:

1. Select the Block tool.
2. In the [New Objects](#) dialog, select Primitives in the Categories box. In the Objects selection box you will have 5 options. Select any one of these objects:
 - block
 - wedge
 - cylinder
 - spike
 - arch

Note: The arch primitive is a special case. It is not actually a primitive object, but an Arch tool which allows you to quickly and easily make multi-piece arches. For more information on the Arch tool, see [Using the Arch Tool](#).

3. Drag a box in a 2D view, making sure it is the correct size and in the correct position in each of the views.
4. Press Enter to create the solid.

Creating Solids by Cloning

Cloning provides a simple way to quickly duplicate an object or group of objects. This includes both brushes and entities, and groups can contain a mixture of both.

1. Go into Selection mode by pressing Shift+S.
2. Select the object (brush, entity, or group of brushes and/or entities) that you would like to clone.
3. Hold shift, and click and drag the selected object to its new position with

the left mouse button.

4. While shift is still being held, release the mouse button. The object has now been cloned.

Prefabs

Once you have a prefab library created (see [Creating and Using Prefabs](#)), it is a simple task to insert a prefab into your level.

There are two ways to put a prefab into your level.

The simple method is, from the [New Objects](#) dialog, to select the prefab library from the Categories selection box, and the specific prefab from the Object selection box, and then press the Insert original prefab button. This will insert the selected prefab in the center of the 2D views. You can do this with either the Selection or Block tool selected. Simply move it around into place. (Make sure Texture Locking is turned on!)

The second way to insert a prefab is with the Block tool. Press Shift+B to go into block mode, then create a box the same approximate size you want the prefab to be. Select the prefab library and prefab object as described above. Now press enter to create the prefab. This will create prefabs at any size you specify, rather than inserting them at original size then requiring you to resize them manually.

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Texturing Solids

Textures are the designs applied to the surfaces of solids. These textures show what material a given solid is made of (brick, steel, sand, etc.). So, while solids give your level form, textures indicate function, create atmosphere and bring your level to life. Skilled application of textures is one of the most important aspects of creating a good level. The Valve Hammer Editor's tools give you easy access to all of the texture properties.

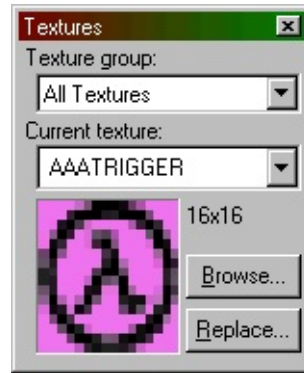
Half-Life's textures are stored in WAD files. A WAD file is a collection (a wad) of textures that the compile tools use to put textures into the compiled map. Texture WADs can be added and removed from the [Textures options](#).

Note: the WAD file dates all the way back to Doom. Doom uses the WAD format to store complete level information, including map, textures, monsters, etc.

When Quake came along, the WAD format changed into the WAD2 format. WAD2 is less of an essential game file than it is a design file. It is used to store 8-bit single-palette textures. During the compile process, the compile tools take textures from a WAD2 and place them directly into the compiled map files.

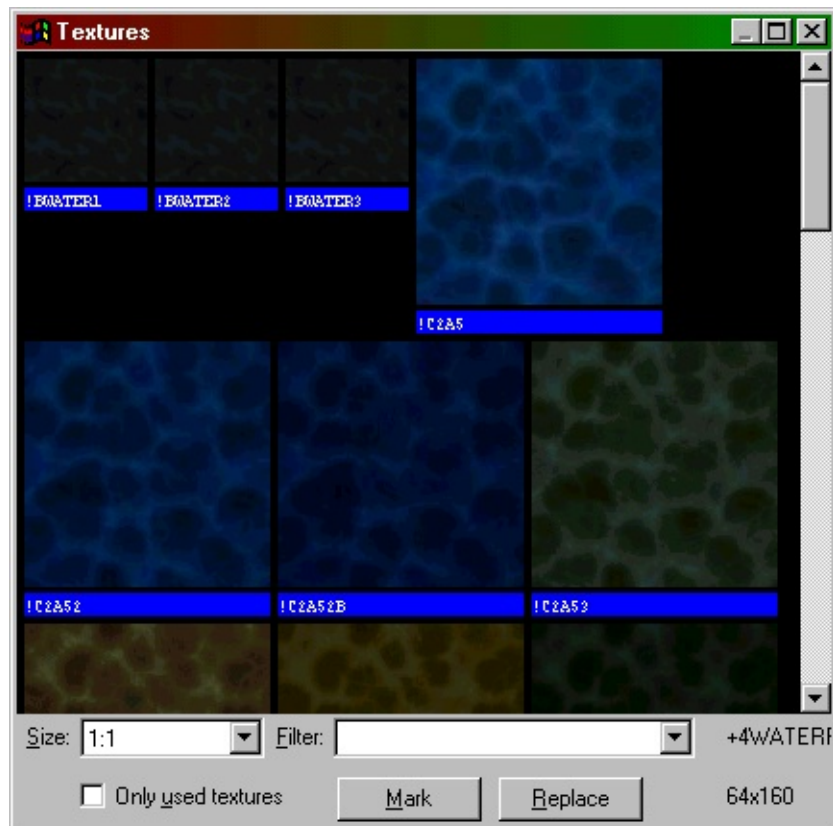
Half-Life uses the WAD3 format. It is essentially the same as the WAD2 format, except that each texture stored includes its own palette. This allows for a much richer color depth than previous texture systems.

Your first encounter with textures will most likely be through the [Textures](#) toolbar.



The [Textures](#) toolbar shows the current texture, and lets you select a new texture. The Current texture drop-down list shows the last eight textures used, then continues with a list of the remaining textures. Use the Texture group scrollbox to view a different subset of textures. You will have a choice between viewing all textures, or viewing individual texture WAD3 files. (A WAD3 file is a file that contains a group of Half-Life textures). You can add or remove WAD3 files in the [Textures Options](#) dialog.

Texture Browser The Browse... button opens the texture browser, where you can select from all available textures (as dictated by the Texture Group setting).



The Size scroll box allows you to set the displayed size of the textures. You can choose from 32x32, 64x64, 128x128, or 1:1 (full size).

The Filter dialog box allows you to type in a search string. Any texture that includes that string as part of its name will be displayed. You can specify more than one filter string. For example, if you specify the search string "sign 7," any texture with "sign" and "7" in its name will be displayed. Previous search strings can be accessed by clicking on the down arrow.

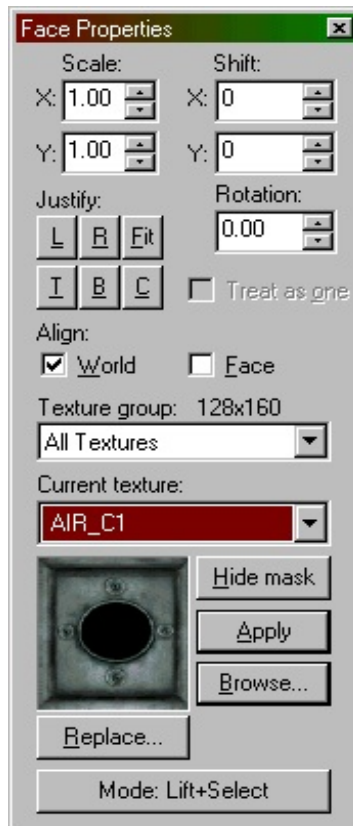
The Only used textures option tells the browser to display only those textures that have been used in your level. This helps you limit your use of different textures, and avoid using different textures for the same purpose. For games such as Half-Life, textures are stored in the compiled maps. Reducing the number of textures will decrease the compiled map size. Fewer textures to load also means memory savings while playing the map in the game.

The Mark button allows you to mark (select) all texture faces that contain the selected texture. You can then adjust those faces as required (textures, surface attributes, etc). This is best used in conjunction with the Face Properties dialog. If the normal Apply Textures map tool is used, any brush with one face selected will have the new texture applied to every face of the brush.

The Replace button (also accessible through the Tools menu) will bring up the [Replace Textures](#) dialog.

Texture Application Mode

You will eventually need to modify a texture in some way. The Face Properties dialog (Texture Application mode) allows you to manipulate the texture properties of selected brush faces. To select a brush face, click on it in the 3D view with the left mouse button. To select multiple faces, hold down the Ctrl key while you do this. You can also select multiple brushes *before* bringing up the Face Properties tool.



Scale (X/Y)

You can modify the scale of a texture to shrink or enlarge it. Numbers less than 1 shrink, and numbers greater than 1 enlarge. You can use negative values to mirror the texture in one or both planes.

Shift (X/Y)

This simply allows you to shift the texture around in the X and/or Y planes.

Rotation

Rotation allows you to rotate the texture on a brush face.

Justify

There are six buttons you can press here: L (left), R (right), Fit (fit to face), T (top), B (bottom), and C (center on face). These allow you to quickly and easily align a texture on a brush.

Treat as one

If you have multiple faces selected, you can check this option to treat all the faces as one single face.

Align: World and Face

These checkboxes allow you to set the texture alignment style for the selected faces. World alignment causes textures to be aligned according to world coordinates (if you're familiar with versions of the editor before 3.3, this was the standard behavior). Face alignment uses the coordinates of the face as a basis for alignment.

Texture Group

You can select which group of textures you want to use - either all of them, or individual texture WADs.

Current Texture

The name of the current texture is displayed here, along with a thumbnail picture of the texture.

Hide Mask

If this button is toggled on, the selected brush face(s) will not appear as selected (with a reddish hue). This may make it easier to perform some texture manipulations.

Mode

This lets you select what feature you want the left mouse button to be. The available options are Lift+Select, Lift, Select, Apply (texture only), Apply (texture + values).

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Reshaping Solids

Once you've created a simple, textured solid, you can use Hammer's tools for refining your work. These tools include vertex manipulation, face splitting, clip planes and carving. Each of these tools lets you customize your solids by modifying the basic shapes you started with. You will find that you can achieve the same or similar effect using different tools; some are just faster and easier than others when it comes to making the precise change you are trying to make.

- [Vertex Manipulation](#)
- [Face Splitting](#)
- [Vertex Scaling](#)
- [Clip Planes](#)
- [Carving](#)

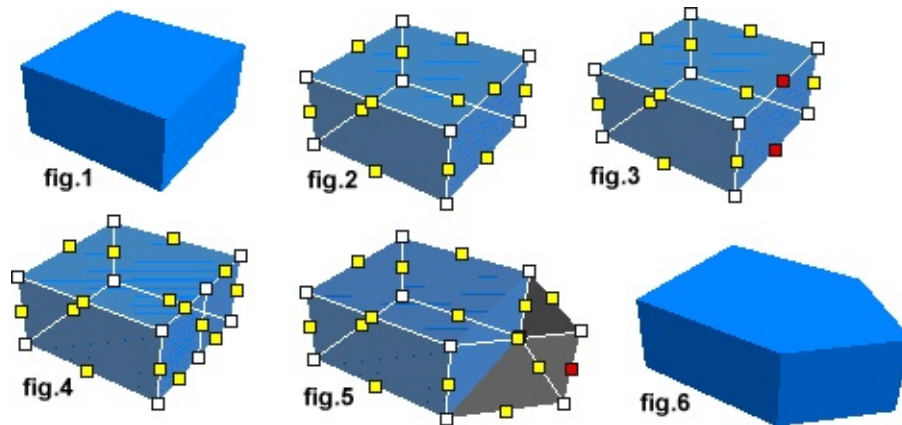
Vertex Manipulation

Vertex manipulation allows you to move individual vertices and edges of a solid, easily creating irregular shapes. This lets you make shapes that are impossible with plane clipping, and extremely difficult to do with carving. Vertex manipulation is also useful for modifying specific pieces of a group of objects, where resizing would interfere with other objects in the group.

Note: While Vertex Manipulation makes the creation of new complex shapes easy, it also make the creation of invalid shapes easy. Remember that concave shapes are not valid within the editor.

Vertices and Edges

A vertex (or plural, vertices) is a corner of a solid. Moving one vertex will not effect any other vertex, but it will change the position of any edges associated with it. An edge is a point between two vertices which, when moved, will change the position of its two associated vertices, as well as their associated edges (no other vertices are effected besides the first two).



1. Select the object(s) you would like to manipulate. (fig. 1)
2. Select the vertex manipulation (VM) tool from the MapTools toolbar. The selected solid will turn into a wireframe-type solid with partial shading (fig. 2). Vertices will be marked as white dots, and edges as yellow dots.

Note: You can keep pressing the VM button, or press Shift-V, to cycle through the 3 VM modes. They are: vertices and edges (default), vertices only, and edges only.
3. Click on the vertices/edges you want to move. They will be highlighted (fig. 3). Note that you can move both vertices and edges.
4. You can either drag the selected vertices or edges with the mouse, or move them in the selected 2D view using the arrow keys.

Tip: To undo anything done in VM mode, you will have to first exit VM mode.

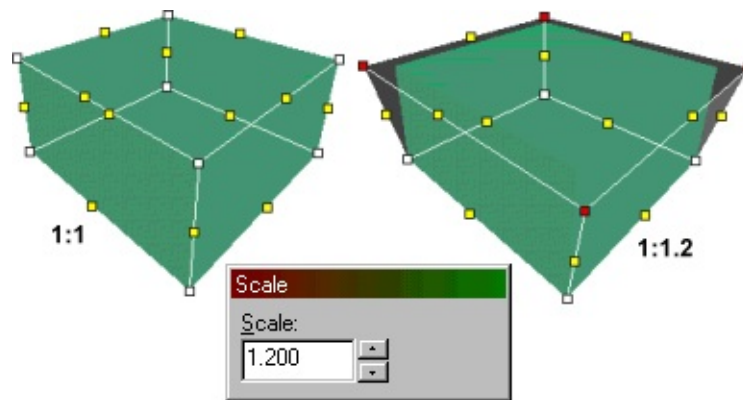
Face Splitting

Face splitting allows you to add additional faces to a brush. This is a useful tool when you want to add complexity to an object in your level. Simply select two opposing edges (fig. 3) and press Ctrl+F. This will turn the two edges into vertices, and place a new edge between them (fig.4).

Vertex Scaling

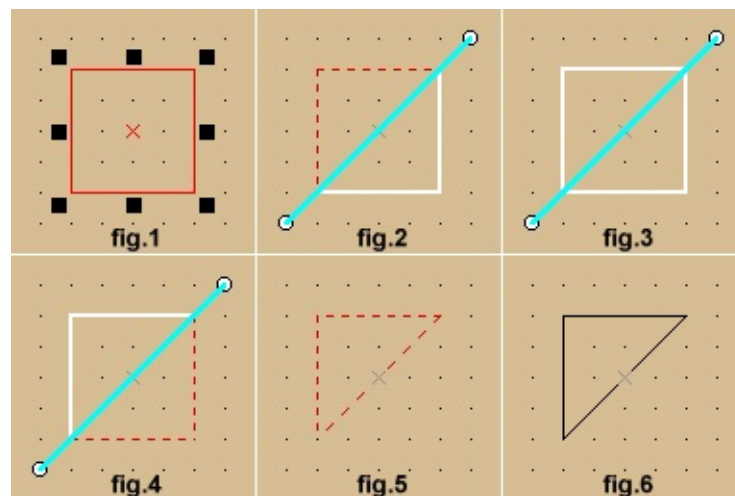
Vertex scaling allows you to select a number of vertices and change their scale in relation to each other. The process is quite simple.

1. Select an object.
2. Go into Vertex Manipulation mode.
3. Select a number of vertices. In the example picture below, all of the vertices on the top of a block have been selected.
4. Press Alt-E to bring up the Vertex Scaling box. You can press the up or down arrows to move in 0.1 increments, or enter a scale value directly in the text area. The scale changes will automatically be reflected on the selected object.
5. When you are done with the scaling, press Escape to close the Scale box.



Clip Planes

Clip planes let you make a precise cut in a solid, dividing it into two pieces. You then have the option of keeping one or both of the resulting two solids. You may find using Clip Planes quicker and more efficient than carving or vertex manipulation.

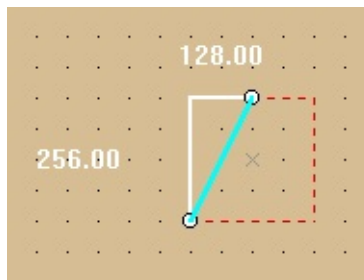


1. Select the object to be clipped. (fig. 1)
2. Select the Clip Plane button on the MapTools toolbar.

Note: You can cycle through the clip modes by clicking on the clip plane button, or by pressing Shift-X (fig.2, 3, 4). You can move both points of the clip line by holding CTRL and dragging one point of it.

3. Drag a line across the selected object. (fig. 2) This is the clip line. The part of the object to be kept will be highlighted in thick white lines.
4. When you are satisfied with the resulting object, press Enter to perform the clip. (fig. 5, 6)

Version 3.3 of the Valve Hammer Editor added a feature to display the size of the solid area of the clipped brush. This is useful if you're trying to clip a brush to an exact size.



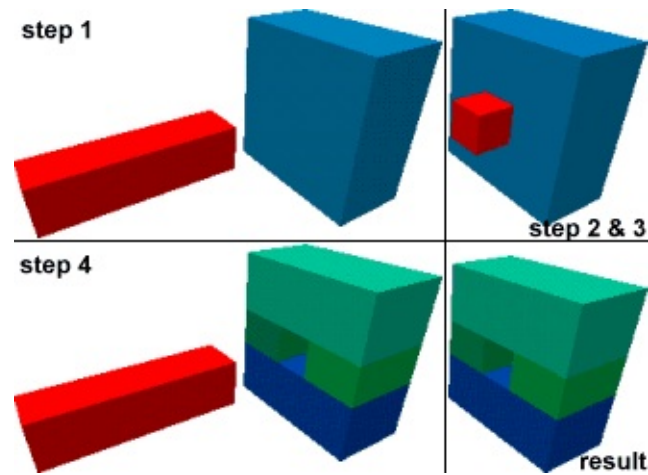
To toggle this on and off, have a 2D view in focus and press the o hotkey.

Carving

In the editor, any solid can be used to carve a volume out of other solids. For example, you can place a solid within the room's wall and tell Hammer to subtract the solid from the wall -- effectively punching a hole right through the wall! This feature is not limited to using cubic shapes as a carving tool: any solid -- cylinders, cubes and wedges -- can be used to carve other solids.

While carving, especially with a cylinder, it is best to carve the smallest areas possible. For example, if you attempt to create a hole in a large wall using the cylinder, the editor will automatically break the wall face into several pieces, something you don't want. Instead, a more efficient way to carve would be to first carve a square hole (the same size as the circle) and leave the square in place. Then you can take the cylinder and carve a hole in the small square. This

will eliminate the splintering effect for the wall.



1. In this step, you have the object you want to carve into (in blue), and the object you will be carving with (in red). The red object should be selected.
2. Position the carving object so that it is where you want the hole to be.
3. Press Ctrl+Shift+C, or right click on the selected object and select Carve. You can also press the Carve button on the MapOperations toolbar.
4. Either delete the object you used to make the carve, or use it to fill the hole. (For example, use the piece carved out of a doorway as the door.)

Note: In the picture above, you can see that the cube (which was previously one object) is now broken into 4 separate objects. This is because the game engine cannot handle concave objects, and the editor has broken it into convex pieces.

Creating Entities

Entities give life to your level. They are the monsters, doors, switches and lights that turn your static architecture into an interactive environment. Careful use of entities is critical to creating an interesting, unique and fun level for others to play.

Unlike solids, entities are not created within the editor. Most users will simply choose from a pre-existing set of entities provided with Half-Life. Once you have selected the appropriate entity, the editor gives you control over its placement and other properties.

There are two types of entities: point-based and brush-based.

Point-based entities exist only at a certain exact point. Examples include lights, monsters, and player start points. (Monsters *do* have an area, but it is defined by the game code and is not modifiable from within the map.) Some point type entities are just that: points. For example, the env_beam entity, which controls Half-Life's beam effects, uses two info_target entities as targets and the beam of light runs between these two points.

Brush-based entities are entities that depend on either a brush for their physical presence (like doors, trains, and other moving entities) or an "area" (like triggers, which require an activation field).

Point-based entity placement Placement of point entities is simple:

1. Press Shift+E to go into entity mode..
2. From the [New Objects](#) toolbar, select the entity you want.

Note: the entity list in the New Objects dialog is for point-based entities only. For information on brush-based entities, please refer to the following section.

3. Click in a 2D window, then position the entity cursor with the mouse.
4. Pressing Enter will create the entity.

Tip: You will most likely need to edit the entity properties after you place a point-based entity. Select the entity and press Alt-Enter to view the entities properties dialog box. This dialog pops up automatically for brush-based entities.

Brush-based entity placement

Brush-based entities are a bit more complicated than point-based entities, but should not pose any difficulties once you know their basics.

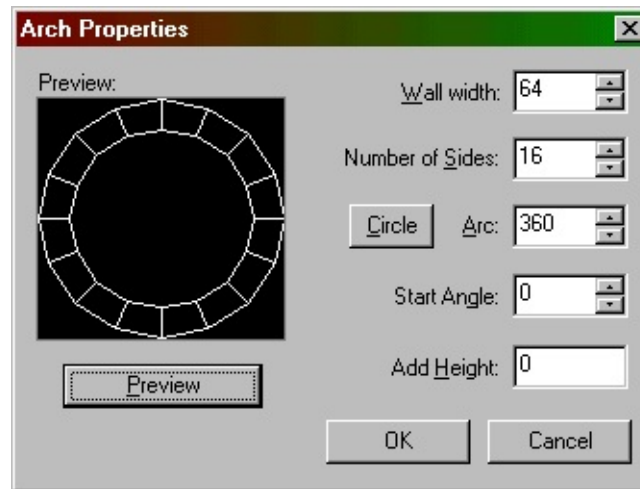
1. [Create the object](#) where you would like it. You can use as many brushes as you'd like when building a brush-based entity.
2. Select the entire object.
3. Press the Ctrl+T hotkey, which will turn the selected objects into a brush based entity and...
4. ...bring up the entity properties dialog. Select the appropriate entity from the Class list box. Once the entity type is selected, you can modify the entity properties as needed.

Tip: Bring up the entity properties for a brush-based entity the same way you do for a point-based entity: select it and press Alt-Enter.

5. You will notice that brush-based entities appear as a different color in the 2D windows, typically purple.

Using the Arch Tool

The Arch tool allows you to make complicated curved arches or circular staircases simply and quickly.



There are a number of options you can modify here:

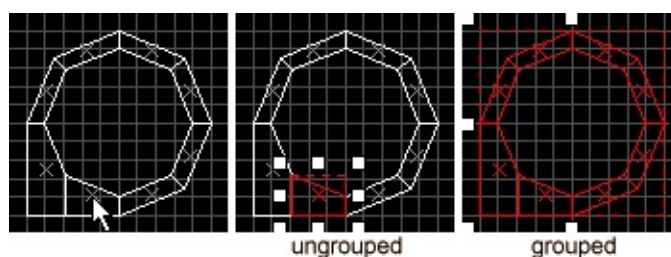
- **Wall width** - this is the width of the individual pieces of the arch. You can extend the arch walls outward by using a negative value here.
- **Number of sides** - enter a value between 3 and 100.
- **Arc** - this is the number of degrees the arch will span. You can enter a value here that is between 8 and 360.
- **Start Angle** - this lets you adjust the angle of the arch pieces.
- **Add Height** - Each successive piece of the arch will be raised this many units. This allows you to make, for example, a curved stairway. In that case, you'd set this value to the height you want each stair to rise.

Grouping and VisGrouping

Even relatively modest levels can contain hundreds--even thousands--of objects. Hammer offers two ways of keeping track of items in your level: normal grouping and visibility grouping (VisGroups).

Note: VisGroup refers to "visibility group" and has nothing at all to do with the Vis compile tool. VisGroups only affect what you see in the editor and have no in-game effect.

Grouping Normal grouping is used to literally "glue" things together.

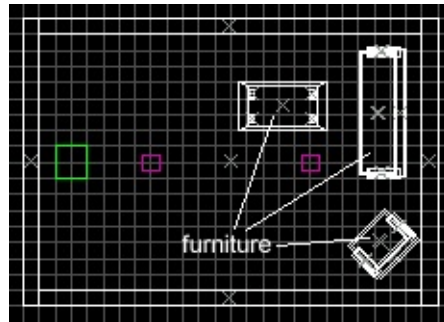


The object shown above is made of eight brushes. If you were to click on a single brush when the object is ungrouped, only that brush would be selected; you'd have to ctrl-click each piece to select the whole thing (or drag a selection box over the whole thing). When the object is grouped, however, you can click on any individual brush to select everything in entire group.

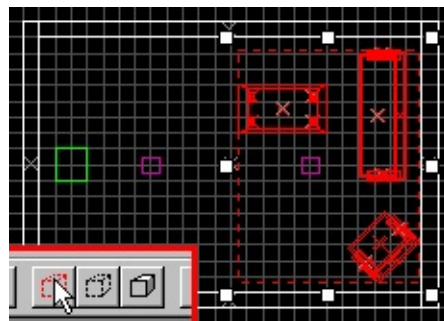
- brushes do not need to touch in order to be grouped together
- groups may contain any combination of brushes and entities
- you can work on an individual piece of a group of objects by enabling the ignore groups option
- grouped objects can be combined with VisGroups (see below)

VisGroups

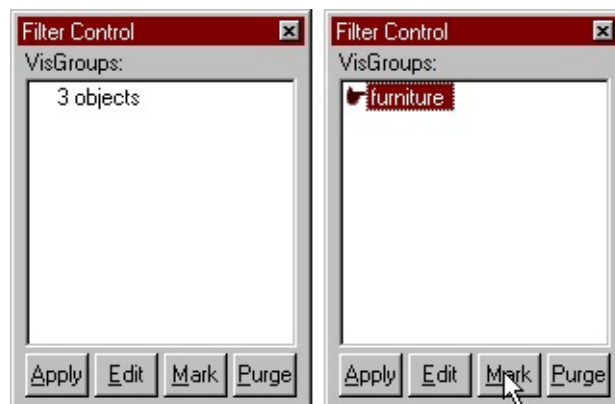
VisGroups stands for visibility groups. VisGroups make managing objects easy by letting you control what you see on your screen. With VisGroups, you can arbitrarily assign objects to a specific group; you can then turn the visibility of that group on and off. Let's say you have the following section in your map:



Now, you'd like to be able to make the furniture a group, but you still want to move each piece around independently from the others. This is precisely what VisGroups are good for.



First, select the furniture pieces, then click on the hide selected objects button. This will do two things: hide the objects in the 2D and 3D views, and create a new VisGroup with the selected objects.



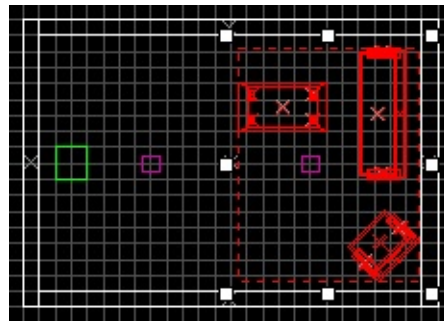
The new VisGroup will appear in the [Filter Control](#) toolbar named after the number of objects it contains. To change the name of the VisGroup to something more suitable, select the VisGroup, then click on it again. You will be able to change the name to whatever you like. You can also click to the left of the name and a hand will appear next to the VisGroup. This is how you set which

VisGroups are visible and which are not. You must press the Apply button for the changes to take effect.

Some things you can do with the mouse in the [Filter Control](#) toolbar:

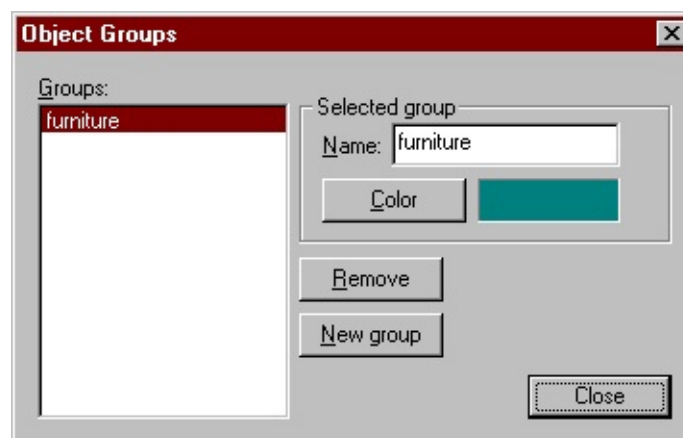
- drag one group onto another group (this will cause the first group to be merged with the second group, and the resulting group will keep the name of the second)
- drag a group out of the Filter Control dialog (this will allow you to delete VisGroupings)

If you make the furniture group visible, then select it and press the Mark button, the group will be highlighted in the 2D and 3D views.



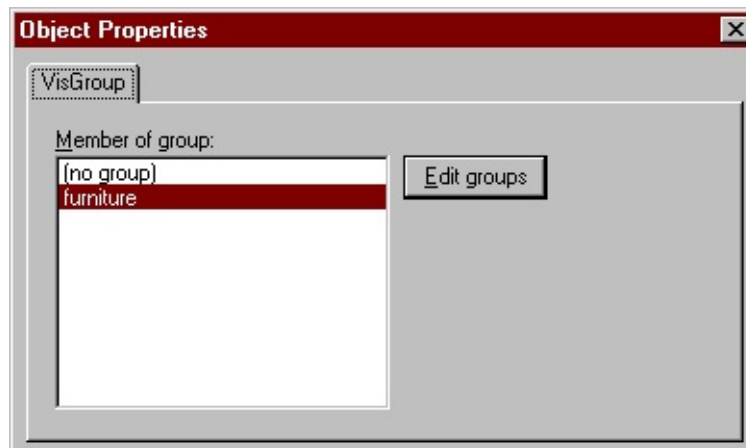
The other two options available to you are Purge and Edit. Pressing the Purge button removes any VisGroups from the Filter Control dialog that do not actually contain any objects (which is possible if you have deleted objects or made one VisGroup part of another).

The Edit button brings up the Object Groups dialog.



This dialog allows you to change the name and color of existing VisGroups, or delete them. Deleting the VisGroup does not actually delete the group's objects, it just eliminates the VisGroup, making them individual unlinked objects. You also have the option of creating new (empty) VisGroups with the New Group button.

The last way you can use VisGroups is through an object's [properties](#), in the VisGroup tab.



The above dialog will also be a part of an entity's [Properties](#) dialog. To assign the object to a VisGroup, just select the appropriate group. To remove the VisGrouping from this object, click on (no group).

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Creating and Using Prefabs

Prefab libraries allow you to easily store pieces from your levels that can be used over and over. Examples of useful prefabs include hallways, light fixtures, unique objects (vehicles, etc), and furniture.

First, you need to create the library.

In the [Prefab Factory](#) dialog, click on the Add Library button. The Edit Library dialog will appear. Enter the name of the library and a brief description. Press Enter to create the library.

Now, you have an empty library and a map with several objects you'd like to add as prefabs.

To add an object to the prefab library, find the object in your map and select all of it and press Ctrl+R.



The Create Prefab dialog will appear allowing you to define the name and description of the prefab, as well as which library it will be placed into. Repeat this process for as many objects as you'd like.

Placing Prefabs Placing the objects you make is just as simple.

In the [New Objects](#) toolbar, select the library from the Categories list, select the

object from the Objects list, and press the Insert original prefab button. The object will be inserted in your map, centered in the 2D views. For more information on placing prefabs, see [Creating Solids](#).

[Return to the Valve Hammer Editor 3.x User's Guide](#)

Valve Hammer Editor 3.4 Setup Guide

Valve Hammer Editor 3.4 has had many changes made to it since version 2.1, and some of these changes mean that you'll need to set things up a little differently compared to what you might be used to.

Before You Start Before you continue on to use Hammer, there are a few things you should keep in mind?:

- Hammer 3.4 makes use of OpenGL for its 3D rendering. As such, driver compatibility issues may exist, so be sure you are running the latest version of your OpenGL video drivers. Below are some links to the major driver pages.
 - [NVIDIA driver page](#)
 - [3DFX driver page](#)
 - [ATI driver page](#)
 - [Matrox driver page](#)
- Hammer 3.4 only allows for the creation of Half-Life (and Half-Life mod) levels.
- In order for Hammer 3.4's 3D sprite preview to work, you must unpack the Sprites folder from your PAK file. For more information, see the [Unpacking Your PAK](#) tutorial. It is also recommended that you unpack your sounds and models folders from the PAK file.

Setup

The first thing you should do after installing Hammer is configure it's settings. The options settings can be reached by selecting Options from the Tools menu. There are six different tabs in the Option's dialog:

- [Game Configurations](#)

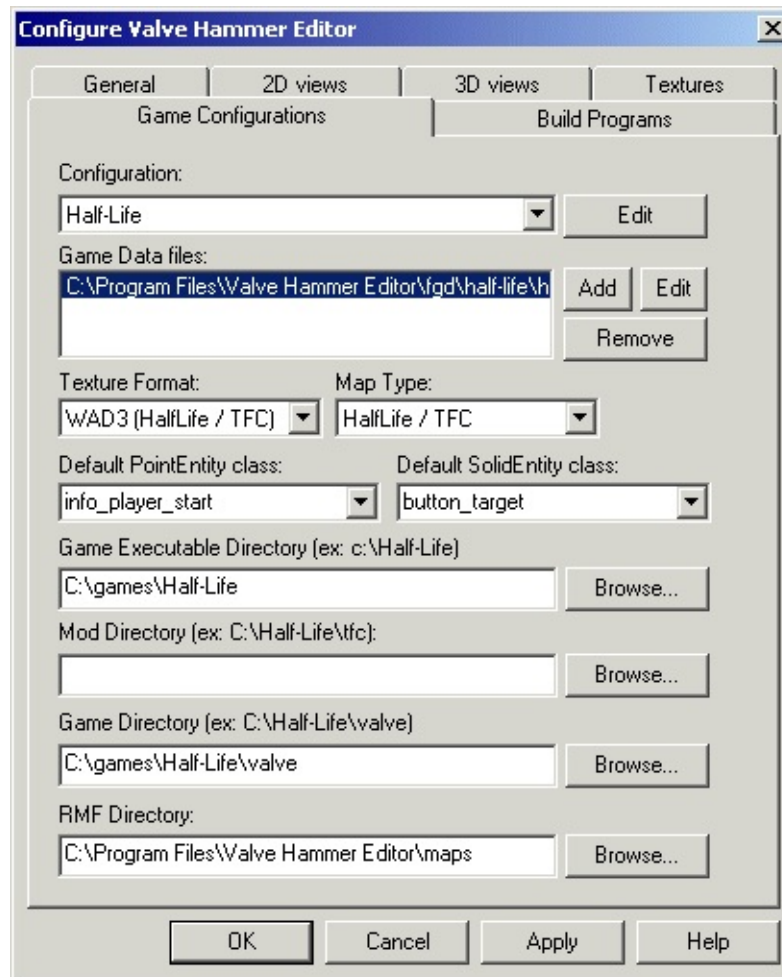
- [Build Programs](#)
- [General](#)
- [2D Views](#)
- [3D Views](#)
- [Textures](#)

If you are going to use Hammer to compile your maps, you also need to setup the compile modes.

- [Compile Setup](#)

Game Configurations

The Game Configuration options allow you to setup the editor to edit Half-Life and it's mods. A separate configuration is needed for each game or mod.



Configuration

If this is a new installation of Hammer, there won't be any game configurations yet. Click Edit, then Add, and enter Half-Life (or whatever Half-Life mod you're setting up the editor for).

For new users, note that you can add more than one game configuration. This is handy if you plan on working on Half-Life levels *and* Half-Life mod levels.

Game Data Files

The game data file contains all of the entity information that will be used to create your maps. Hammer's game data files have the file extension .FGD. It must be present in order for you to create levels.

1. Click on the Add button.
2. An Open dialog will appear. If it doesn't show the contents of your Hammer folder, browse to it.
3. In the FGD folder, browse to and select the game .FGD that you require. Hammer ships with game data files for Half-Life, Team Fortress 1.5, Counter-Strike, and Day of Defeat.
4. You will be taken back to the Options dialog. You will see that C:\Valve Hammer Editor\halflife.fgd (or wherever you've stored the FGD) has been added to the Game Data File list.

Note: There is a new version of the Half-Life game data file [available here](#). You can unzip it into your Hammer directory and use the same steps as above to add it as your Half-Life game data file. Be sure to remove the previous version of the game data file first.

Texture Format and Map Type

The Hammer editor is only capable of creating Half-Life maps. As such, these two options are locked on WAD3 (Half-Life) for the texture format, and Half-Life for the map format.

Default PointEntity and SolidEntity class

Select a default point and solid entity here. This controls which entity will appear selected when you go to place an entity. It is only a time-saving device. I recommend using info_player_start as the default point entity and func_door as the default solid entity.

Game Executable Directory

The value here should be the directory where your Half-Life executable is stored. The value should *not* include hl.exe. For example, C:\Half-Life is correct, but C:\Half-Life\hl.exe is not.

Mod Directory

If you are setting up the editor for a mod, the game directory of the mod should go here. For example, if Half-Life is in C:\Half-Life, and you're setting up Hammer for Team Fortress Classic, the Mod Directory value would be C:\Half-Life\tfc. For a normal Half-Life game configuration, the Mod Directory should be set to C:\Half-Life\valve.

Game Directory

This value should be the base game directory for Half-Life. Keeping with the above examples, this would be C:\Half-Life\valve.

Note: The above two settings are required for the search paths for displaying sprites in the 3D view. They allow you to use custom development directories rather than locking you into using the standard Half-Life directories. For the most part, it is likely that you *will* be using the standard directories.

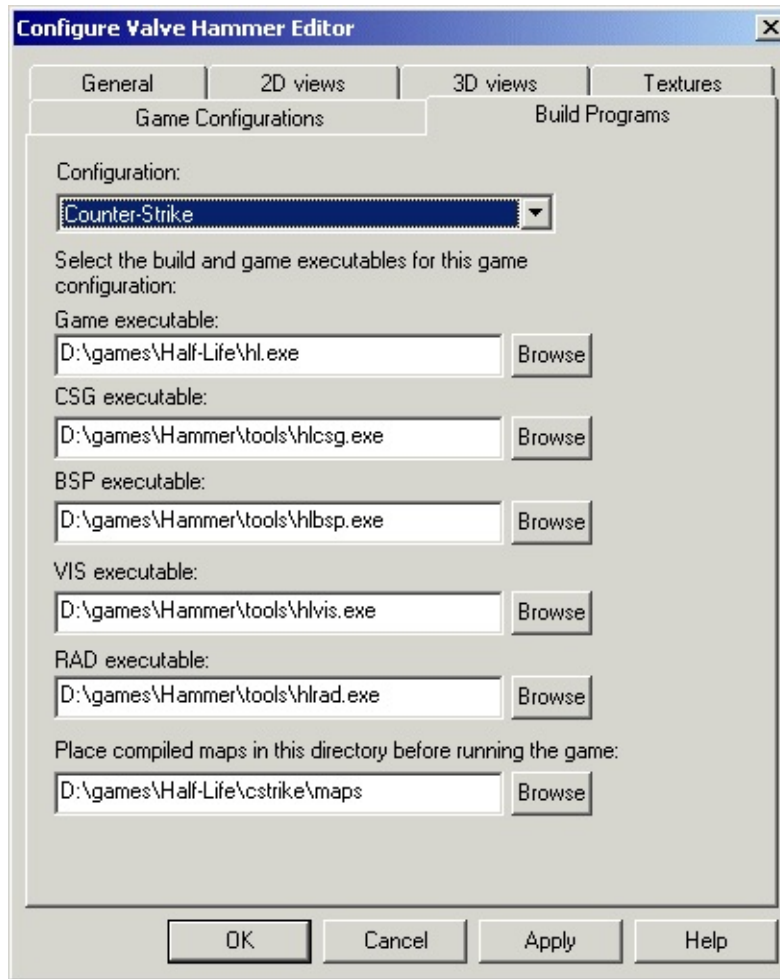
RMF Directory

The value here is the directory where the editor will store your maps. I recommend using the maps folder that gets created in your Hammer directory.

Note: As a point of trivia, RMF stands for Rich Map Format. It is the Valve Hammer Editor's proprietary map format. Before a map can be compiled, it must be converted from an RMF file to a .MAP file. The editor handles this conversion automatically, but you can also do it manually by selecting Export to MAP from the Files menu.

Build Programs

The settings in the Build Programs dialog affect how the editor handles things when you compile your map. If you don't plan on using the editor to compile your maps, then this section is irrelevant and does not need to be filled out.



Configuration

Choose which game configuration you are setting up the Build Programs for here. If you've only created a Half-Life game configuration, that's all you'll have access to here.

Note: If no Configuration is shown, click on the Ok button to return to the editor screen, then select Options from the Tools menu again and click on the Build Programs tab. The Configuration you've previously created should now be listed.

Game executable

Specify the game executable here. If you know the path to the Half-Life executable, type in the path and filename here, otherwise, click on the Browse button and browse to the Half-Life directory. Select hl.exe and click the Open button.

CSG, BSP, VIS, and RAD executable

This is where you specify the compile tools to be used. The standard tools are, in order, qcsg.exe, qbsp2.exe, vis.exe, and qrad.exe. If you are using Zoner's compile tools, they will be hlcsf.exe, hlbsp.exe, hlvis.exe, and hlrad.exe. By default the tools are stored in the *Valve Hammer Editor\tools* folder.

Note: There is another file, lights.rad, in your *Valve Hammer Editor\tools* directory. It contains information that controls the texture lighting when you compile a Half-Life map. If you move the compile tools to a different directory, you must also move the lights.rad file.

Place compiled maps in this directory before running the game

Compiled maps must be placed in a specific directory for Half-Life to recognize their existence. This directory is the Valve\Maps\ directory, located in your Half-Life folder. For example, my Half-Life game directory is D:\games\half-life, so my compiled-maps directory is D:\games\half-life\valve\maps. If no Maps directory exists in the Valve folder, you must create one.

Note: If you are setting up the editor for mod editing, replace "valve" in the above example with the mod directory. For example, d:\games\half-life\tfc\maps.

General Options

You can leave these options at their default settings for now. Once you are familiar with Hammer, it is recommended you play around with these settings until you've got things set like you prefer. More information can be found in the [General](#) dialog page.

2D Views

You can leave these options at their default settings for now. Once you are familiar with Hammer, it is recommended you play around with these settings until you've got things set like you prefer. More information can be found in the [2D Views](#) dialog page.

3D Views

You can leave these options at their default settings for now. Once you are familiar with Hammer it is recommended you play around with these settings

until you've got things set like you prefer. More information can be found in the [3D Views](#) dialog page.

Textures

Without any textures installed, you cannot create a map. The main Half-Life texture file is halflife.wad, and it is located in your Half-Life directory, in the Valve folder. To add it to your Texture list, follow these steps.

1. Click on the Textures options tab.
2. In the Textures options, click on the Add WAD button.
3. An Open dialog will appear. Use it to browse to your Half-Life directory.
4. Double-click on the Valve folder to view its contents.
5. Find halflife.wad and select it.
6. Click on the Open button to add it to your Textures list.

Note: You can also add liquids.wad, xeno.wad, and decals.wad using the above steps. There are a number of other .WAD files that may be available in the Valve folder, including cached.wad, gfx.wad, pldecals.wad, and spraypaint.wad - do not use these. These files are generated by Half-Life and will not be the same on all user's systems.

Compile Setup

If you are going to use the editor to compile your maps, there are two different compile modes for you to use: the normal compile mode and the advanced compile mode.

Normal Compile Mode

The normal compile mode allows for one button compiling with no further setup other than what has been mentioned above.

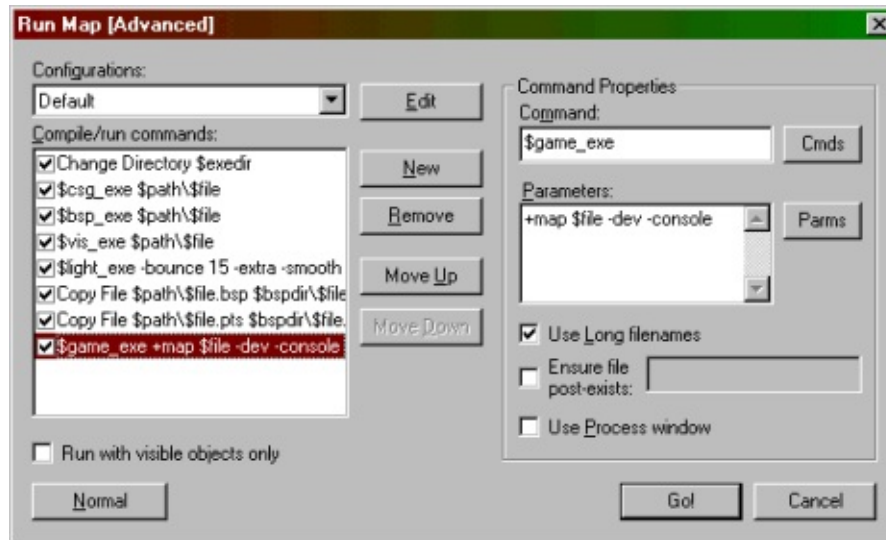


- ***Additional game parameters*** - This is where you specify the additional parameters for the game executable. If you are compiling a map for a mod, be sure to include the `-game <moddir>` parameter as shown in the above picture. `-dev` and `-console`, also shown, are good parameters to use when testing maps.
- ***Run CSG, BSP, VIS, and RAD*** - These settings allow you to define which tools will be run along with a couple of their most basic parameters.
- ***Don't run the game*** - If this option is checked, the game will not be started after the map has finished compiling.
- ***Save visible objects only*** - If this option is checked, only objects that are not hidden will be compiled into the map.
- ***Expert (button)*** - Press this to go into Advanced Compile Mode.

Note: User's of Worldcraft 2.x may remember that the normal compile mode did not work correctly. This problem is fixed now, and the normal compile mode can be used without error, but the advanced compile mode is still recommended due to it's increased versatility.

Advanced Compile Mode

The advanced compile mode gives you much more freedom over how the compile process is performed.



To make a standard compile setup, follow these steps:

1. Press the New button 8 times. Eight check-boxes will be created.
2. Click on the 1st checkbox line. Click on the Cmds button and select Change Directory. Click on the Parms button and select Game Executable Directory.

Note: It is important that you select the commands from the Cmds menu rather than typing them in the Command text box. Although they may look the same, typing out the commands will cause problems in the compile.

3. Click on the 2nd checkbox line. Click on the Cmds button and select CSG program. In the Parameters text box, type \$path\$file. It is very important that "\$path\$file" be typed correctly.
4. Click on the 3rd checkbox line. Click on the Cmds button and select BSP program. In the Parameters text box, type \$path\$file.
5. Click on the 4th checkbox line. Click on the Cmds button and select VIS program. In the Parameters text box, type \$path\$file.
6. Click on the 5th checkbox line. Click on the Cmds button and select LIGHT program. In the Parameters text box, type \$path\$file.
7. Click on the 6th checkbox line. Click on Copy File in the Cmds menu. In the Parameters text box, type \$path\$file.bsp \$bspdir\$file.bsp.
8. Click on the 7th checkbox line. Click on Copy File in the Cmds menu. In the Parameters text box, type \$path\$file.pts \$bspdir\$file.pts. (The .pts file

is created when there is a leak in your level)

9. Click on the 8th checkbox line. Click on Game Program in the Cmds menu. In the parameters text box, type +map \$file. Other parameters you might also want to use are:

- **-dev** - this puts the game in developer mode, providing extra debugging information for you when you're testing your map.
- **-console** - this enables the in-game console which is accessible by pressing the ~ key.
- **+set deathmatch 1** - this starts the game in deathmatch mode. Some games, like Counterstrike, require this.
- **-game <moddir>** - this should be used when you're making a map for a mod. For example, if you're making a TFC map, you'd need to use the -game tfc parameter.

Conclusion

Hammer is now setup for mapping. If you are new to mapping, it is best that you proceed to the [User's Guide](#) section to familiarize yourself with the basic functions of Valve Hammer Editor 3.4.

WWW Reference

There are a number of places you can go to on the internet for help with VHE 3.4 and Half-Life editing.

Design Support If you are having technical problems setting up or running Valve Hammer Editor, email tools@valvesoftware.com for information and assistance. When emailing Design Support, be sure to include the [information necessary for a quick response](#).

Editing Support

Below is a short list of websites dedicated to providing information on Half-Life editing and related topics.

[The Handy Vandal's Almanac](#) - This is an amazing resource site - it provides an index of pretty much every Half-Life editing resource available on the net.

[Valve Editing Resource Center](#) - The primary source of information for the Valve Hammer Editor and Half-Life editing.

[Editing Discussion Forum](#) - With over thirty thousand discussion posts and an excellent search function, you should be able to find information here on almost any editing problem.

Half-Life Map Design Toolkit

There are a number of utilities that every mapper should find useful. Note that these utilities are not themselves supported by Valve. In most cases, you can refer to a specific utility's website for support.

[Zoner's Half-Life Tools](#) - Zoner's compile tools have a number of improvements and advanced features not found in the standard compile tools.

[GenSurf](#) - This is a fine program for making realistic looking terrain. This works extremely well when using Zoner's compile tools.

[Wally](#) - Wally is a fantastic texture editing tool. It allows you to create textures and Half-Life WAD files from scratch, or editing existing ones. Wally also acts as a PAK file browser/editor.

[Half-Life Model Viewer](#) - This is an excellent OpenGL accelerated model viewer for Half-Life. It lets you view individual animations and their names. HLMV also acts as a PAK file browser.

[SprView](#) - SprView is a small sprite viewer that lets you view Half-Life sprites and adjust their framerate and play-back properties exactly as you'd adjust the sprite entity.

Hotkey Reference

Tools

- Shift+S - Pointer Tool
- Shift+G - Magnify Tool
- Shift+C - Camera Tool
- Shift+E - Entity Tool
- Shift+B - Block Tool
- Shift+A - Texture App Tool
- Shift+T - Apply Current Texture
- Shift+D - Decal Tool
- Shift+X - Clipping Tool
- Shift+V - Vertex Manipulation Tool
- Shift+P - Path Tool

Clipboard/Selection

- Ctrl+C - copy objects to clipboard (also ctrl+insert)
- Ctrl+V - paste objects from clipboard (also shift+insert)
- Ctrl+X - cut objects to clipboard (also shift+del)
- Ctrl+E - center on selection (vertex/object) in all views
- Shift+Q - clear selections
- PgUp - previous selection in "hit" list
- PgDn - next selection in "hit" list

Grid

- [- grid lower
-] - grid higher
- Shift+R - toggle grid on/off
- Shift+W - toggle snap to grid
- p - (with mouse cursor in 3D view) toggle 3D grid on and off

Grouping

- Ctrl+G - group objects
- Ctrl+U - ungroup objects

Carving/Hollowing

- Ctrl+Shift+C - carve
- Ctrl+H - hollow

Creating Objects

- Enter - create object drawn with block/entity tool
- Alt+Shift+C - insert original prefab
- Ctrl+R - create prefab with selected objects
- Ctrl+T - create entity with selected objects; to add objects to an existing entity, select it and the objects and hit Ctrl+T (you will be prompted)

Other Object Operations

- Ctrl+B - snap selected objects to grid (based on bounding box)
- Alt+Enter - selected object's properties

File Stuff

- Ctrl+N - new file
- Ctrl+O - open file
- Ctrl+S - save file

Undo/Redo

- Ctrl+Z - undo
- Ctrl+Y - redo

2D Views

- Ctrl+I - flip vertical
- Ctrl+L - flip horizontal
- Tab - switch view types (top/side/front)
- +/- - zoom in/out (hold Ctrl to synchronize all 2D views)
- 1 to 9 - preset zoom levels
- Space - hold space and left mouse button to drag view
- Alt - disable snap to grid while dragging with the mouse

3D View

- Space - While holding the spacebar:
 - holding the left mouse button allows you to rotate your angle of view in any direction, while the viewing point remains stationary.
 - holding the right mouse button will allow you to move left, right, up, and down while keeping the viewing angle constant.
- Space+Shift - While holding the spacebar and Shift:
 - the left mouse button acts the same as above.
 - the right mouse button allows you to move forward and backward, as well as from side to side.
- z - Pressing the (lowercase) z puts the editor 3D view into its new "noclip" mode (this mode can be disabled in the 3D options dialog)
 - the mouse acts like +mlook is on (moving it around will change the player's direction of focus).
 - W - forward
 - S - backward
 - A - strafe left
 - D - strafe right
- Shift+Right-mouse-click - will select an entire brush and bring up the 3D View context menu, as above.

- p - (with mouse cursor in 3D view) toggle 3D grid on and off
- o - (with mouse cursor in 3D view) display frames per second and yaw/pitch of camera.

Morph Tool

- Ctrl+F - split faces (vertex manipulation) - must have two edges or vertices selected
- Alt+E - vertex scaling

Selection Tool

- Ctrl - hold Ctrl and click to select multiple objects
- Shift - hold Shift when rotating to constrain rotation to 15 degrees; hold Shift when moving an object to create a copy (clone) of that object
- (drag) - hold left mouse button and drag to select with a box; press Enter to select objects hitting the box, or press Shift+Enter to select objects only entirely within the box.

Camera Tool

- Shift - create a new camera by holding Shift and dragging with the left mouse button.
- PgUp - cycle to the previous camera position
- PgDn - cycle to the next camera position
- Delete - delete the current camera position

Clipper Tool

- Enter - perform clip
- Ctrl - hold Ctrl and drag with the left mouse button to move both handles of the clipper
- Shift - hold Shift and drag with the left mouse button to create a new clip plane without performing the previous clip.
- o - (with mouse cursor in 2D view) toggle clip sizes on and off

Miscellaneous

- Alt+B - export again
- Ctrl+A - auto-size 4 views to center
- Shift - (while in Texture Application mode) select entire object
- Shift+L - texture lock
- Shift+Z - maximize/restore view
- Ctrl+M - transform dialog
- Alt+P - check map for errors
- F9 - run map
- Escape - abort current tool/mouse operation (drag/drop)

Troubleshooting

Below are some of the most common questions and answers from Design Support mail.

- [What information should I send to Design Support?](#)
 - [How do I copy the contents of the compile Process Window?](#)
 - [How do I setup the editor?](#)
 - [I'm getting the error "Couldn't Validate Half-Life".](#)
 - [My multiplayer map isn't appearing in the multiplayer map list.](#)
 - [One of my toolbars is missing! How do I get it back?](#)
 - [How do I get back to having 4 views - the 3D, top, front, and side views?](#)
 - [My level has a leak. How do I fix it?](#)
 - [My sprites have suddenly turned into textures!](#)
-

↑ *What information should I send to Design Support?*

Before you email [Design Support](#), make sure you've setup the editor as outlined in the [Valve Hammer Editor 3.4 Setup Guide](#), otherwise you will be referred there before any help can be provided.

When emailing [Design Support](#), be sure to include the following pieces of information:

- What operating system you're using.
- What video card you're using (and in the case of multiple video cards, *all* of the video cards you're using).
- Which video drivers you're using.
- Anytime compiling or running a map is involved in the problem, provide the complete contents of the compile Process Window in your email (see note below).

This information isn't always required, but it can really speed up the problem resolution when all information is available.

[\(top\)](#)

↑ ***How do I copy the contents of the compile Process Window?***

When debugging map problems, one of the most important bits of information is the compile process output. [= 4\)](#)

[BSPSPopupOnMouseOver\(event\)" CLASS="BSSCPopup">This is the compile Process Window](#) (the actual contents when you see it will be similar, but may not be exactly alike).

Right-click inside the window. A small menu will pop up. Choose *Select All*. All of the text in the Process Window will become selected. Now, right-click on the selected text and this time choose *Copy*. You can now paste it wherever you like (ie: into an email, if you're emailing Design Support).

[\(top\)](#)

↑ ***How do I setup the editor?***

For information on this, please refer to the [Valve Hammer Editor 3.4 Setup Guide](#).

[\(top\)](#)

↑ ***I'm getting the error "Couldn't Validate Half-Life".***

This error occurs when you try to run the Half-Life executable without first changing to the Half-Life folder. There are a number of reasons this can happen.

- The editor and/or the advanced compile mode is setup incorrectly. Make sure you've set things up as outlined in the [Valve Hammer Editor 3.4 Setup Guide](#).
- The editor's advanced compile mode can appear to be setup correctly and certain commands will appear to fail for no reason. This most commonly happens with the Change Directory and Copy File commands. When this type of error occurs, you will see the following type of output in the compile Process Window:

```
** Executing...  
** Command: Change Directory
```

**** Parameters:** D:\games\Half-Life

*** Could not execute the command:**
"Change Directory" D:\games\Half-Life\valve
*** Windows gave the error message:**
"The operation completed successfully."

Even though it looks correct, and the parameter is a valid directory, it produces that error. Confusing huh? :) To fix a command you've already created, just select it, then select the appropriate command from the Cmds menu. Nothing will appear to change, but it *will* be fixed.

[\(top\)](#)

↑ ***My multiplayer map isn't appearing in the multiplayer map list.***

For a level to appear in the multiplayer map list, it must have an info_player_deathmatch in it.

The info_player_deathmatch entity creates a player spawn point for multiplayer games. When a player spawns in a level (either because he is just entering the level, or he has just died and is respawning) his location will be picked randomly from all of the available info_player_deathmatch spawn points.

Its a good idea to include a large number of these spawn points to prevent predictable spawning.

[\(top\)](#)

↑ ***One of my toolbars is missing! How do I get it back?***

This requires a bit of registry tweaking. Load up Regedit by selecting *Run* from the Start menu and typing **regedit**, then pressing the Enter key.

Once in Regedit, expand the following path:

HKEY_CURRENT_USER\Software\Valve\Valve Hammer Editor

In Regedit's Valve Hammer Editor folder, select and delete the key called Barstate-Summary. This will reset all of the toolbars to their default

positions.

[\(top\)](#)

↑ ***How do I get back to having 4 views - the 3D, top, front, and side views?***

Select *Options* from the Tools menu. Click on the *General* tab. Under Window Setup where it says *Use independent window configurations*, make sure there is *not* a check next to that option, then close and restart the editor.

[\(top\)](#)

↑ ***My level has a leak. How do I fix it?***

For information on this, please refer to the tutorial entitled [How to I fix leaks in my level?](#)

[\(top\)](#)

↑ ***My sprites have suddenly turned into textures!***

There is an issue where, if you open a new 3D window (or close the existing one then open a new one), sprites are displayed as seemingly random textures in the 3D view. To fix this, save your map, then close and reload it.

FGD Format Changes

A number of changes have been made to the FGD format to allow for enhanced entity properties and to facilitate the display of sprites in the 3D view.

Iconic Entities Iconic Entities are those entities that have a sprite associated with them. Icon sprites should be stored in Valve Hammer Editor\sprites and are referenced with the **iconsprite()** helper. The format is like this:

```
@PointClass iconsprite("sprites/lightbulb.spr") base(Light) = light : "Light Entity"
[
    &ldots;
]
```

Icon sprites should be of alphatest type.

Sprite-based Entities

Sprite-based entities are those entities that have some connection to a sprite. For Half-Life, this includes the env_glow, env_sprite, and cyclor_sprite. These entities use the **sprite()** helper. Below is the modified version of the env_sprite.

```
@PointClass sprite() base(Targetname, RenderFields) size(-4 -4 -4, 4 4 4) = env_sprite : "Sprite"
[
    framerate(string) : "Framerate" : "10.0"
    model(sprite) : "Sprite Name" : "sprites/ glow01.spr"
    scale(string) : "Scale" : ""
    spawnflags(flags) =
    [
        1: "Start on" : 0
        2: "Play Once" : 0
    ]
]
```

The first difference is the addition of the **sprite()** helper in the definition line. The other is that the model key is now a **sprite** type, rather than a generic string. The sprite listed in this key will be displayed in the 3D view.

The sprite() helper is looking specifically for the model key. It will also look at the value of the framerate, scale, and rendermode keys, so the sprite will be displayed as it is meant to (ie: animated sprites animate at the specified framerate).

File Browsing

Any entity property that accepts the path/filename of a sprite, sound, or model should now be of the sprite, sound, or studio type, rather than just a simple string.

```
model(sprite) : "Sprite Name" : "sprites/ glow01.spr"  
model(studio) : "Model Name" : "models/can.mdl"  
message(sound) : "WAV Name"
```

The result of using these key types rather than the generic string type is the entity properties will have a button beside them allowing you to browse through the appropriate game directories for the files. This does not provide any preview capabilities.

For this to work correctly, you must have the sprite, model, and sound folders unpacked from the PAK files to the appropriate directories. Wally (the texture editor) provides a simple UI for exploring and exporting files from the PAK file. Wally is available at <http://www.telefragged.com/wally/>.

Decals

The decal entity definition now requires the addition of the **decal()** helper to make it work properly in the editor.

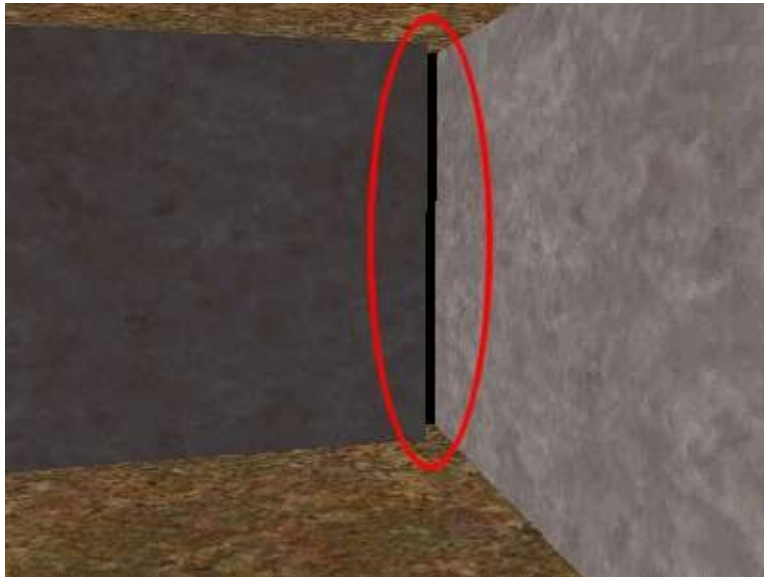
```
@PointClass decal() base(Targetname, Appearflags) = infodecal : "Decal"  
[  
    texture(decal)  
]
```

Without the **decal()** helper, decals will appear as small square blocks in the editor's 3D view.

How do I fix leaks in my level?

Everyone experiences them. Everyone hates them. They're the frickin' level leak. Below are some causes and effects, and a number of ways you can deal with leaks.

The Cause First, you should know what causes them. The most common cause of a leak is a gap in your level that "leaks" into the void. Look at the picture below.



What you see is an obvious gap. Through the gap is an area outside the level. When this gap exists, the compile tools don't know what is inside and what is outside the level. Because of this, the BSP process will report its LEAK LEAK LEAK error, and VIS will not run, and RAD will only perform direct lighting calculations (that is, no light bounces).

Sometimes these gaps aren't quite so obvious. They can be as little as one unit wide and still cause a leak.

Another big cause of leaks is having a point entity outside the level. All point entities *must* be inside valid level space.

One more cause is when you seal off your level with a brush entity, like a

func_door. Brush entities are ignored where visibility by the engine is concerned, and so they also count as a leak if they are contacting both the inside and the outside (the void) of the level. They create the same condition as if there were a gap in their place.

The Effects

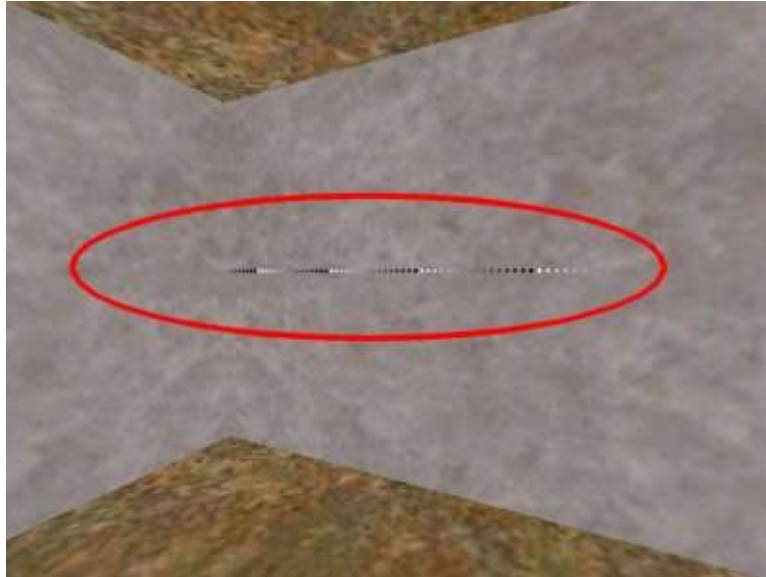
A leak in a level has a number of bad effects. First, the BSP process will report the leak, and it will not produce a portal file (*mapname.prt*). The portal file is used by the VIS process to perform its visibility calculations. Since there is no portal file, VIS will not run at all. When VIS doesn't run, it doesn't produce the files necessary for the RAD process to calculate its light bounces. Due to this, the RAD process will only perform direct lighting - no light bounces. (If you are using Zoner's Half-Life compile tools - [you should be!](#) - the RAD process won't run at all)

The most serious effect by far is that the VIS process will not run when a leak is present. When the level is run in Half-Life, the rendering engine will not have any visibility info available with which to limit itself, so it will attempt to draw the entire level. This will most certainly lead to high polygon counts and may lead to things like doors, items, and the sky blacking out or not being drawn if the maximum rendered polygon limit (which is 800 polygons in software rendering mode) is reached.

The Fixes:

The Pointfile

There are a number of ways to deal with leaks. The first way you should check is by loading the pointfile. Make sure that *mapname.pts* is being copied into the *valve\maps* folder along with the compile map (you'll have to move it manually otherwise). The *.pts* file contains a set of coordinates that form a line of dots that should lead you to the leak. To load the pointfile in the game, bring down the console and type pointfile.



You will see a message similar to 1050 points read, and you should see something similar to the above picture. (You may have to run about your level to find it).

The default engine limit is 2048 points, which may not be enough to trace a line to the leak in your map. If this is the case, start Half-Life at the command prompt with "hl -particles 10000" (or some higher number) to allow the engine to display more particles, and thus trace a longer line to help you find your leak.

When you've loaded the pointfile, type "noclip" to allow you to walk through walls and "r_fullbright 1" to make the map bright so you can see everything. (If you're using Zoner's compile tools, the map will already be fullbright since the RAD process didn't run.) If there are monsters placed in the level, typing 'god' and 'notarget' at the console will make things easier as well. The 'notarget' command causes monsters not to become hostile to your presence, and 'god' makes you immune to damage of any sort. If you noclip outside the level, it might also be useful to type 'gl_clear 1' in the console. This will turn the void into a solid color, rather than the mess that it will be if you're using the glide or opengl renderer.

Once you find the line, follow it. It bounces all over the place, goes in screwy directions, both inside and outside the level. At some point, it will pass through a hole in the walls of your level. There's your leak.

The Big Block Method

If your level is especially large and you're having difficulty tracking down your leak, you might benefit from using the "Big Block Method" of leak tracking. What this is essentially, is you covering a large portion of your level with a solid block (make sure it goes completely through from top to bottom). A recommended size is a quarter of your level.

Place the large block over a portion of your level, then attempt to compile. Move the block to a new section and recompile until the leak error disappears. When this happens, you know the leak is somewhere within the space of the block. Now you can repeat the procedure using a smaller block, only within the space that was covered last with the big block.

This method would only work, of course, if you had one leak, or a number of leaks that would all be covered at once by the block.

Neal White adds this about the Big Block method:

I discovered the Big Block Method on my own, but I use it for an entirely different purpose. It is **very** helpful in tracking down those damn leaf-saw-into-leaf portal bugs. It's the only good way I've found to track them down. It also can be used to drastically reduce your level compile times when you just need to test the lighting in a room or two.

Great idea!

The Bright Light Method

Some people favor this method over the big block, but depending on the level, it can be very time consuming. It involves placing a large box around the entire level, then placing a number of very bright purple (or other easily visible color) lights inside the box, but outside what would be your level. Now, compile the level, and since the level is inside a big box, it should compile fine, and the RAD process should also run fine. Once its in the game, run around your level looking for the light leaks.

The main drawback to this is that the compile for the map may take an exceptionally long time, and you'd probably be better off using the pointfile method, but this method may work better if you're faced with multiple leaks throughout your level.

***Note:** Some people would point out - if placing a box around your level gets rid of the leak, why not*

stop there? Well, it gets rid of the leak, but it doesn't get rid of the main bad effect of the leak - VIS still cannot properly optimize the level because it doesn't know what is inside or outside the level. You will still see the high polygon counts and poor framerate. This is not a solution.

An Ounce of Prevention

It was pointed out to me that I originally missed talking about the most important way to fix leaks, and this is by preventing them in the first place. Take your time when building, make sure things are snapped properly to the grid. This is the note I received -

this is where the phrase "an ounce of prevention is worth a pound of cure" comes in.. ie: take your time .. make sure all wall joins are solid and thick .. I always make 64 unit walls all around the outside of my map and then build detail inside of that .. this allows me to see if I happen to have misaligned brushes ...

The 64 unit thick walls aren't completely necessary, but they will make it more obvious if you have a misaligned brush. The whole point is, be careful, be attentive, take your time. I'll end this before it sounds like a father/son lecture.

Conclusion

Leaks are bad, and can be really frustrating to find. Hopefully this information has shed a little bit of light on the subject for those that need it. Also remember, sometimes a good eye is all that's needed to find the leak. Look around the level in the editor - look for gaps or misaligned brushes, point entities in the void, etc. Kill those leaks!

This article originally appeared at the [Half-Life Editing Resource Center](#)

How do I access the Half-Life console?

Accessing the Half-Life Console

Contrary to past Quake-engine games, Half-Life does not, by default, allow you to access the console from within the game. To activate the console, use the following method.

1. Find the shortcut icon that loads up Half-Life. Right-click on it and select Properties from the pop-up menu.
2. The properties dialog for the shortcut will come up. The first available text box, marked Target, is what you'll want to edit.

The default value here will be the path and executable of Half-Life. Something like `c:\games\half-life\hl.exe` for example. Change this to `c:\games\half-life\hl.exe -console` (the path will be different for you, so just add the '-console' to the end of what's already there). Note that there is a space between `"...hl.exe"` and `"-console"`.

You can add any other parameters you'd like to use here as well. For instance, I have the following parameter list: `-dev -console -numericping` which causes the game to start in developers mode, with the console activated. and with pings displayed as numbers rather than dots (for net games).

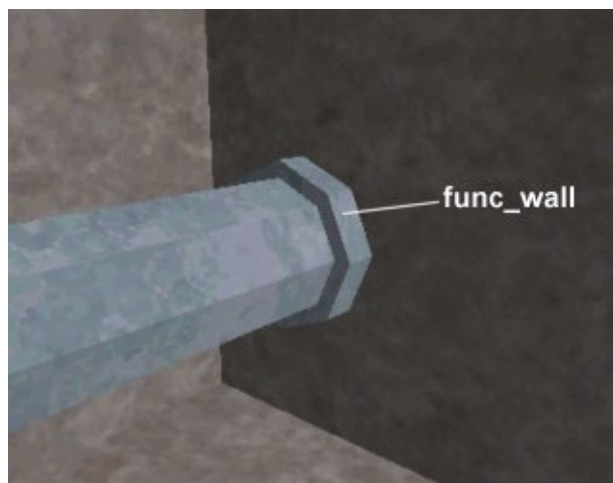
This article originally appeared at the [Half-Life Editing Resource Center](#)

How can I reduce polygon counts?

Polygon Reduction Methods: A quick and dirty guide. This article originally started out as an email to someone, but it kept going on and on and on. So, here it is. I'm not claiming this is a complete guide to low polygon counts, but its some quick advice that's good to know.

There are a number of things to keep in mind concerning r_speeds / polygon count.

- entity brushes will not break up other solid brushes, world brushes will That is, if you have a 10 sided cylinder that makes contact with a flat brush, that flat brush will be broken into many pieces (polygons). This can affect both how the brush looks and how many polygons are required to be drawn. There are two main ways to work around this problem
 - where one object meets another, leave a 1 (or more) unit gap. This is best done when the player can't see the area where the gap is (ie: at the top of a pillar, or under some shallow water.
 - where one object meets another, leave a small gap and join the two objects with a func_wall. For example, if you've got a set of pipes that meet up with a wall, put a pipe flange on the wall and connect the pipe to that. Make the flange a func_wall.



Below are some screenshots of the above concepts in action. I used columns as examples but the concept is applicable to many different things.

[= 4\) BSPSPopupOnMouseOver\(event\)" CLASS="BSSCPopup">](#)



This picture shows a normal column with no polygon reduction efforts made. Click to see a larger picture.



This picture shows a normal column with the above mentioned polygon reduction methods. Click to see a larger picture.

The above two pictures were taken in software mode with `r_drawflat` turned on. This will draw the world in flat/solid polygons (similar to the editor's flat-shaded

view). This is an excellent way of checking out your level for areas that may be causing problems with high polygon counts.

***Note:** To turn on `r_drawflat`, type **`r_drawflat 1`** in the Half-Life console. You need to have run Half-Life with the **`-dev`** and **`-console`** parameters for this to work (but then, you should use those parameters anyway when you're testing maps).*



This picture shows a normal in-game shot of both columns.

There isn't a heck of a lot of difference between the two columns, except that the column on the right has a much lower polygon count.

Another thing to keep in mind is texture size. The compile tools split brushes along their texture sizes. For instance, if you apply a 128 x 128 texture to a large wall, that wall would be split into 128x128 polygons. If you change the scale of the texture to X*2 and Y*2, the wall will be split into 256x256 polygons. The scale of a texture can have a big effect on the number of polygons. Whenever possible, when it won't be visually negative, increase the scale of a texture (this isn't that often, mind you, but for outdoor areas, it can be handy).

***Note:** You can access the scale properties of a texture in Hammer by pressing `shift+A` to bring up the Texture Application Tool, then clicking on a brush face to select it. Modify its scale and press the Apply button to change the brush face.*

You'll notice in the included pictures that the walls and ceiling/floor are single polygons. What I've done is stretched them to X*1000 Y*1000 scale. In the in-game shot, you'll see that this produces a flat shaded texture. The only time you

would do this in a real map would be one that is completely surrounded by blackness, when the stretched texture wouldn't be visible. Of course, in that case, you could just use the black sky.

Last, remember that the game engine isn't as smart as you. Or maybe its a whole lot smarter. It can see through doors (and any other brush entity), and can see somewhat around corners. Make sure you have world brushes sufficiently blocking sight into other areas, otherwise the engine will be drawing those areas too.

This article originally appeared at the [Half-Life Editing Resource Center](#)

Unpacking Your Pack

Half-Life stores the majority of it's content in PAK files. The PAK file is an archive that contains files in a specific directory structure that the game can read from.

For editing purposes, it is most preferable when some or all of a PAK's contents is "unpacked" - that is, that the contents reside in their normal directory structure rather than inside the PAK file.

There are a number of benefits to having the PAK file.

- easy access to sounds, sprites, and model files.
- for mod makers, easy to add and remove contents
- makes it possible to "browse" for content in the editor's entity properties, where applicable
- make it possible for the editor to display sprites in the 3D view (Hammer does not read out of the PAK file)

The first thing you need is a program capable of browsing and editing the PAK file. Wally is a good tool for this, since it's also a fantastic texture editor and is freely available. You can get Wally from it's website, here -

<http://www.telefragged.com/wally/>

Unpacking

Unpacking PAK contents is pretty simple. The following example shows how to unpack the Sprite folder.

1. Load Wally and click on Open in the File menu. Browse to the Half-Life folder. Double-click on the valve folder. Click on pak0.pak and press the Open button.
2. A window will open with the contents of the PAK file displayed. Right-click on the Sprites folder and select Export. In the Destination text box, browse to and select the valve folder inside your Half-Life folder.

3. Press the Ok button to extract the files.
4. That's it.

It's important to maintain the correct directory structure of files. If you're unpacking a PAK that is in a specific game directory, the unpacked files should be exported *only* to that game directory.



```
20.0 ms 63 / 62 / 72 poly 0 surf  
19.9 ms 63 / 62 / 72 poly 0 surf  
24.8 ms 63 / 62 / 72 poly 0 surf  
20.5 ms 63 / 62 / 72 poly 0 surf
```



This is what the columns look like in the game. (I've stretched the scale of the background walls and floor/ceiling as it was unimportant and provides a better viewing area for the columns).

The column on the left is the one that is composed completely of world brushes.