BUILD–DOCUMENTATION

.□□□□□□...3□.□□□□□□.□□□.□□□.□□□.□□□□.□□□.□□□□□□□.□□□.□□□□.□□□□□□.□□□.□□□.□□□□□.□□□□□□□.□□□□□□□□□□~.□□□.□□□□□.□□□□□□□.□□□□□□□□□□.□□□.□□□□.□□□□□□□□□□□□□□□.□□□.□□□□□□□□..□□.□□.~.□□.□□□.□□□.□□□□.□□□□.□□□□□□□□□□.□□□□□.□□□.□□□□□.□□□□□□□□□□.□□.□□□□□.□□□~.□□□□□□.□□.□□□.□□□.□□.□□□□□□□□□.□□□□□.□□□.□□□□□.□□□□□□..□□□.□□□□□.□□□□□□□□.□□□□□□□□□.□□□.□□□□□.□□□□□□□□□□□□□□□~

Learning the Build Editor

Copyright & Legal Stuff

Table of Contents

1) Introduction

You are reading the Beginner's documentation for Build, the Level Editor 3D Realms used to make their Duke Nukem 3D and Shadow Warrior levels. It is not a complete document and it is not supposed to replace a complete Level Design Handbook. It is an introduction to both Duke Nukem 3D and Shadow Warrior Level Design as well as probably any game using the Build engine and the Build editor.

This documentation was written by Steffen „*Duke Addict*" Itterheim. I'm not working for Apogee/3D Realms, but I'm working with them. I help to maintain their Compuserve Forum as a SysOp together with Joe Siegler. Besides that I'm part of their small beta tester team, one of those 13 privileged persons who get to play Apogee/3D Realms games before anyone else. I would like to thank Joe Siegler and Keith Schuler for their help and support while I was writing this documentation.

2) What is Build?

Build - a word that means „to construct" or „to develop". Build is the 3D engine 3D Realms used to create immersive 3D games and it has been licensed by other companies as well, so there are a lot more Build games on the market than just 3D Realms' Duke Nukem 3D and Shadow Warrior. Build is also the editor program that was used to create all the levels for these Build games. If I talk about Build in this document, I mean the Build editor, unless otherwise mentioned.

Build is a great tool, to say the least. It is also a complex tool, so be aware that you need to learn quite a lot. Yes, you might have heard people say it's the easiest to use and most powerful editor for 3D games. If this is your first look at Build, you may wonder why - I'll try to explain it to you. Before Build there was no editor that would display the 2D map in full 3D and even let you edit the level while you're walking through it in first person perspective. You will immediately see the changes you have made to walls or sprites thus reducing the time spent to load the level in the game, just to look if your changes came out right. However, Build can not animate objects, that is you can't use doors or platforms in Build. In the Build editor everything is key driven, you do not have a menu. You need to memorize a lot of key commands and if you're reluctant to do so, Build is not for you. If you're willing to memorize the keys you will learn how much time they can save you compared to any <click> <click> <click> <drop> <choice> <ok> style menu system. I hope to make it as easy for you as possible to successfully master the first steps of Level Design using the Build editor.

3) Understanding 3D Level Design

The Build engine takes a 2D layout and renders it in realtime 3D. This means that you're bound to some restrictions that you should know of before you begin. Since you have to design the level structure as a 2D ground plan you can't create a solid 3D object that floats in mid-air. You can not even build bridges.

Oh, do I hear some protest? Yeah, you're right! When you play games like Duke Nukem 3D or Shadow Warrior, you probably noticed „real bridges" you can walk under and above. But these are either made of sprites, meaning flat objects you can place anywhere in 3D space or by using a floor/ceiling „mirror" trick in Shadow Warrior that allows one to see up to a second floor (or down to the first floor). All monsters are sprites for example, and unless you turn on the 3D objects (voxels) in the game all items like keys, weapons and health packs are sprites, too. But we get to this later, for now just know that you can't create true 3D objects unless you use the sprite trick, or the new design tricks to make one believe that there are true 3D locations in Shadow Warrior.

If you take a look at one of the original levels you will notice that the map is split into many differently shaped and sized clusters. These clusters are called sectors and define a certain area of a map. A sector is a polygon of any shape or size. A polygon is an enclosed area that has at least three sides (walls or walls) and three points (vertex, pl. vertices). The most simple sector is a triangle, consisting of three vertices (points) and three walls, enclosing a triangular area of space. These three walls define an area in space which you can give certain attributes. This is called a sector. A sector defines values for the height and angle of the floor and ceiling, the brightness in this sector, the floor and ceiling texture and many more things. If you want to have two different floor/ceiling textures in your room, or different brightness levels, you need to create two or more sectors and change the attributes of each sector accordingly to get the desired result. The same goes for different floor or ceiling heights. Don't worry though, Build makes it easy for you to split and join sectors as well as creating new ones, applying textures to floors, ceilings and walls, adjusting brightness levels and creating slopes.

4) Your First Level

One needs to get experience and should learn how the engine behind the game works in order to create good levels. Designing levels is not easy and takes a lot of time, designing great levels takes probably ten times as long. Your first level will not be much more than just a bunch of simple rooms. You will need to use your creativity and the textures provided with the games to add chairs, desks and various other items to make the room look as realistic as possible. And a good level consists of many such realistic rooms which form together the theme of the map, for example a movie set, a volcano or a japanese garden with cute little horny labbits. As a level designer you are responsible for every aspect in your level, gameplay and flow as well as architecture and immersion. You also have to create all the special effects like crashing cars, exploding walls and you will have to place weapons and monsters.

5) Setting up Build

Use the BSETUP.EXE program to change keys for strafing and other keys. I recommend NOT to change all keys because many descriptions of Build game effects rely on the fact that most people are using the default keys. Plus it can be troublesome if you assign keys for movement to a key that performs a function, like stretching textures. So be very careful when changing Build's default keys with the BSETUP program. What you should do in BSETUP is to change the video mode of the 3D display to 640x480, unless you like the default 320x200 mode. A higher resolution will make it easier for you to spot the desired texture in the texture list - in very high resolutions like 800x600 and above it will have an adverse effect though, because the textures will be too small to figure out how they will look painted on a wall. 640x480 will also make editing easier on very fast computers, like Pentiums with more than 133 MHz, because texture panning and certain other functions will be really fast in 320x200 mode. The video mode set in BSETUP will only affect the 3D view mode, the 2D view mode resolution can not be changed. Ignore the sound and network setup options in BSETUP, they're not used. If the changes did not took effect after restarting Build make sure that SETUP.DAT is not set to be read-only (write protected).

6) Starting Build

When you load Build the first time you will see a screen like in figure 1. Notice the two main areas, the upper one is the drawing board with a grey grid where you design your basic level layout and at the bottom is the status bar which displays all kinds of information. This is the 2D mode in Build, the top view of your level where you place sprites, create sectors and modify the layout of your level. Editing textures, light levels and other effects is done by using the 3D mode. I will get to that later.

The white arrow is your viewpoint's position in Build and somewhere else on the map there's a brown arrow. The brown arrow is the player's start position in the game, the point the player starts at when you load the level in the respective Build game, so be careful where you place it. To change the brown arrow's position press the **Scroll Lock** key which is located above the **Pos1** key. The brown arrow will then be placed at the exact same position of the white arrow, facing the same direction. To not have to walk through the whole level to get to the place where you want to place the white arrow right

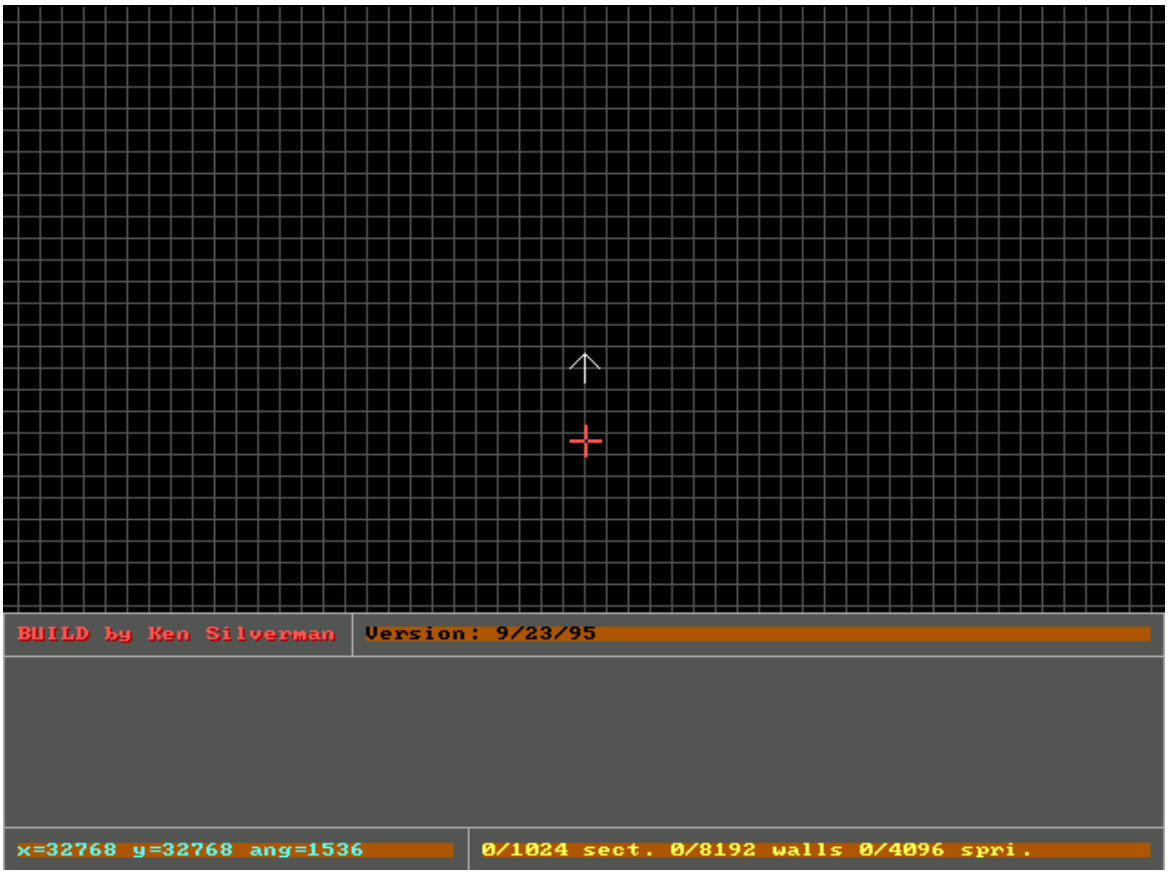


Figure 1: Build Opening Screen

click in the drawning board at the location where you want to have the view focus. This procedure will also scroll the map in general (the 2D view always centers around the white arrow), always placing the white arrow to the new location. To move around the map faster hold down the right mouse button while you move the mouse in the desired direction. The red crosshair is the actual mouse cursor in 2D mode.

7) Drawing Sectors

A sector is the simplest and most fundamental structure in a level. Walls are what make up a sector. So you create sectors by drawing walls and connecting the last wall drawn with the first wall. Build will automatically turn the enclosed space into a sector with predefined attributes, or those of adjactent sectors, whatever applies.
Now you're ready to draw the very first sector in your level. To get into wall drawing mode press the **Space** key and a message will appear in the status bar, telling us that you are in sector drawing mode. A white wall will go from the closest point of the grid where you pressed **Space** up to the current mouse cursor position. Move the mouse a bith north (up) and then press **Space** again to place a vertex there. Move the mouse a bit east, press **Space**, move it south, press **Space** again and then move the mouse cursor on top of the first point. Press **Space** once again to close the sector and to end sector drawing mode. If you missed the first point or generally if you think that you need to place a vertex at a different place without having to quit the whole sector drawing mode use the **Backspace** key to delete the last inserted vertex, unanchoring the last wall drawn. This is also referred to as „backtracking". You can press **Backspace** repeatedly to delete one point after another. When you delete the first point drawn you will exit sector drawing mode without making any changes to your map. After having successfully completed drawing the first sector it will look similar to the one seen in figure 2.

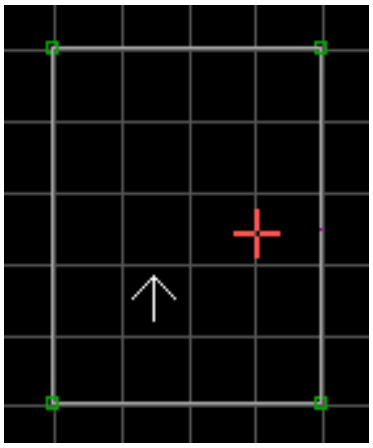


Figure 2: The first sector

Notice that all points were placed at the closest intersection corner of the grid because grid snap is on by default. If you do not like your walls or sprites snap to grid, press the **L** key. The mouse cursor will change its color to white as long as grid snap is disabled. You only need this in very rare situations and turning off grid snap can make certain operations such as drawing sectors very difficult, so leave it on for now. Instead, press the **G** key to cycle through the different grid sizes. Use large grid sizes to plan rooms and basic layout while a smaller grid is preferred for designing detailed objects.

8) Viewing your Level in 3D Mode

After creating your first sector you might want to view it in 3D mode. With Build you do not have to save the map and load it in the game to see it in 3D. Press the **Enter** key on the numeric keypad to enter 3D mode, your one room level will be displayed in a first person view like in the game. The number in the upper left corner is the frame rate, the number of frames the engine draws per second. If this drops below 15 at a certain viewpoint you should rework the visible part of this area because many players will find it too slow to be fun to play, no matter how good it looks. However, if you're building levels on a slow computer, like a 486, you can be sure that most players will get a higher frame rate than you. The same applies if your 3D view is set to a very high resolution, like 800x600. You can walk around in it using the **arrow** keys. Walking sideways depends on the Build version you are using, usually you use **Alt** or **Right Ctrl** to strafe on, or **,** (comma) and **.** (period) to strafe left and right. Speed up the movement speed by holding down the **Shift** key.

If Build does not switch to 3D mode but instead displays a message „Arrow must be inside a sector before entering 3D mode." move the white arrow into one of your sectors. Since there's currently only

one sector, move the cursor into this sector because your viewpoint may not be outside of the level boundaries when switching to 3D mode. To quickly move the white arrow simply move the mouse cursor (the red crosshair) to where you want to place the white arrow and press the right mouse button, or keep it depressed while you move the mouse. Press **Enter** again - don't forget, that's the numeric keypad **Enter** key! If the arrow doesn't fit completely into the sector you should make the sector larger. To get back to 2D mode press the **Enter** key once again.
In 3D mode you can toggle between 3 different height modes by pressing the **Caps Lock** key. Experiment with these movement modes to find out which one works best in which situations. The movement modes change the way you move up or down which is done by pressing the **A** or **Z** key. Finally, with **Ctrl+A** and **Ctrl+Z** you can look up and down in 3D mode.

9) Inserting & Deleting Points

If you want the sector to be a little more complex than a rectangle you can insert points by moving the mouse cursor near to the wall you want to split. The wall will begin to blink when highlighted. There will always be a wall or sprite that blinks because Build automatically highlights the closest wall, vertex and sprite in 2D mode. This will be more important later when editing wall and sprite attributes is discussed. Press the **Insert** key now to split the wall at the point where the small white dot is visible. A vertex will be added on that wall there, splitting it into two. You can move a vertex by moving the cursor over it so that it starts to blink. Press the left mouse button and keep it depressed while you move the mouse to the new location of the vertex. To delete the new vertex move it directly on top of one of its neighboring vertices sharing the same wall in the same sector. A message will appear, saying „Point deleted.".

10) Creating Circular Walls

Build provides a simple function that changes any wall into a circle. When a line is highlighted (blinks) press **C** to get into what I call circle drawing mode. A bunch of yellow vertices will appear. Change the size of the circle by using the mouse and increase or decrease the number of vertices with the **+** (plus) or **-** (minus) key on the numeric keypad. To finally draw the circle on the map press the **Spacebar**. Note that with this function it is possible to accidentally draw walls that cross with no vertex at the crossing point, or some of the walls might be placed in different sectors. Make sure that all of the vertices are placed into one single sector and do not cross or touch other walls.

11) Saving your Work

To save your first one room level to disk make sure you are in 2D mode (press the keypad **Enter** key to return to 2D mode if necessary). Hit the **Esc** key and the status bar will display a message that looks like this: „(N)ew, (L)oad, (S)ave, save (A)s, (Q)uit". The capital letters in brackets represent the hotkeys to press. **N** starts a new, empty map. **L** will load a map from disk. **S** will save the current map under the same filename, overwriting the file. **A** will ask for a new filename before saving the map, but if the filename already exists it will overwrite it, so be careful. **Q** will quit the Build editor.
Since this is the first time you save the map use the save (**A**)s function in order to give your map a different filename. Otherwise if you use (**S**)ave it will be saved as 'NEWBOARD.MAP', unless it was saved before under a different filename.
I recommend to make incremental backups of your level, in case that your map got screwed up during the last modifications. This also allows you to try out new things with the option to go back to an earlier version of your level. Since there's no undo feature in Build you should use incremental backups as means for undoing changes.

12) Creating Adjactent Sectors

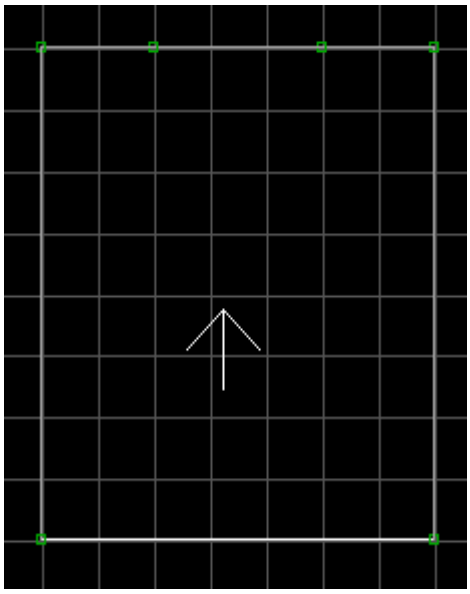


Figure 3: Preparing the adjactent sector

Since I have mentioned them before I should explain how to create adjactent sectors. An adjactent sector is any sector that shares at least one wall with another sector. Use the rectangular sector you built at first and insert two new points at the northern wall. Use the **Insert** key twice on this wall to split the wall twice. You may find it necessary to change the grid size with the **G** key in order to place the points at the desired locations, like in figure 3. You can use the **A** and **Z** key to zoom in or out your view in 2D mode. Use the **Z** key to zoom out if you want to get an overview of your map. Use the **A** key to zoom in if you need to make detailed adjustments. These keys do not modify the size of a level, they simply change the „distance“ from which you view the map.

In this example the north wall was split and two vertices were added. The left vertex will be the starting point for the new adjactent sector. Move the mouse cursor on top of it and press **Space** to start sector drawing mode. Make sure sector drawing starts directly on top of the vertex or else the whole procedure will fail. Next one more vertex is added to the north and one more east of this vertex to create another rectangular sector. Finally press **Space** on the right vertex on the north wall and then on the left vertex to close the sector. Sector drawing mode will end and the wall that both sectors are

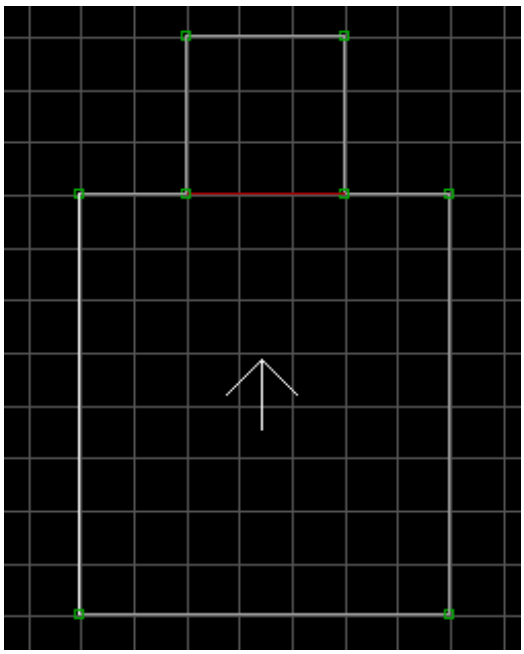


Figure 4: The completed adjactent sector

sharing will change its color from white to red, as seen in figure 4. White walls are the outer bounds of a level and behind a white wall there is virtually nothing. The space behind all white walls (non-sector space) is called NULL space, meaning that the player normally will not be able to access this area and the Build engine will not render the 3D view outside of the level boundaries. This is also the reason why the white wall turned red. It has valid player space (sectors) on both sides, thus it is called a two-sided wall. It can be walked through unless the designer flags it as impassable by pressing the **B** key on the wall in either 2D or 3D mode. If you enter 3D mode you will not be able to see the red wall because there is no wall displayed there. For the moment the red wall is only there to seperate the two sectors.

13) Changing Floor and Ceiling Heights

Currently the two sectors look exactly the same and it is not easy to see where one sector ends and the other one begins. In order to change the floor and ceiling heights of the new sector enter 3D mode and point with the mouse cursor on the floor or ceiling of the smaller sector. Now hold down the left mouse button and press either the **PgUp** (Page Up) or the **PgDn** (Page Down) key. You will notice that the floor or ceiling of the sector will raise or lower a bit every time you hit one of these keys. You do not necessarily have to hold down the left mouse button while doing this. However, as the sector's shape changes, it might be possible that a different surface or wall moves into place under the mouse cursor. Because operations in 3D mode will usually affect the closest wall, floor/ceiling or sprite lock the object you point at by holding down the left mouse button before pressing any other key. That way all operations will only affect the locked object, no matter what comes in front of it while modifying it.
Raise the floor of the new sector and lower the ceiling by pointing the mouse cursor at the floor, holding down the left mouse button and then press the **PgUp** key several times. To lower the ceiling point at the ceiling, hold down the left mouse button and press the **PgDn** key a few times. If you have finished doing so you will see a ledge that seperates the two sectors. This ledge goes along the red border wall which you can see in 2D mode. The red wall can have textures on its upper and lower part, depending on the adjactent sectors floor and ceiling heights. These textures are referred to as upper and lower texture of a 2-sided wall. Only 2-sided walls can have upper and lower textures. In figure 5 a different upper and lower texture has been added as well as another floor and ceiling texture to better show the step and ledge created by raising the floor and lowering the ceiling.

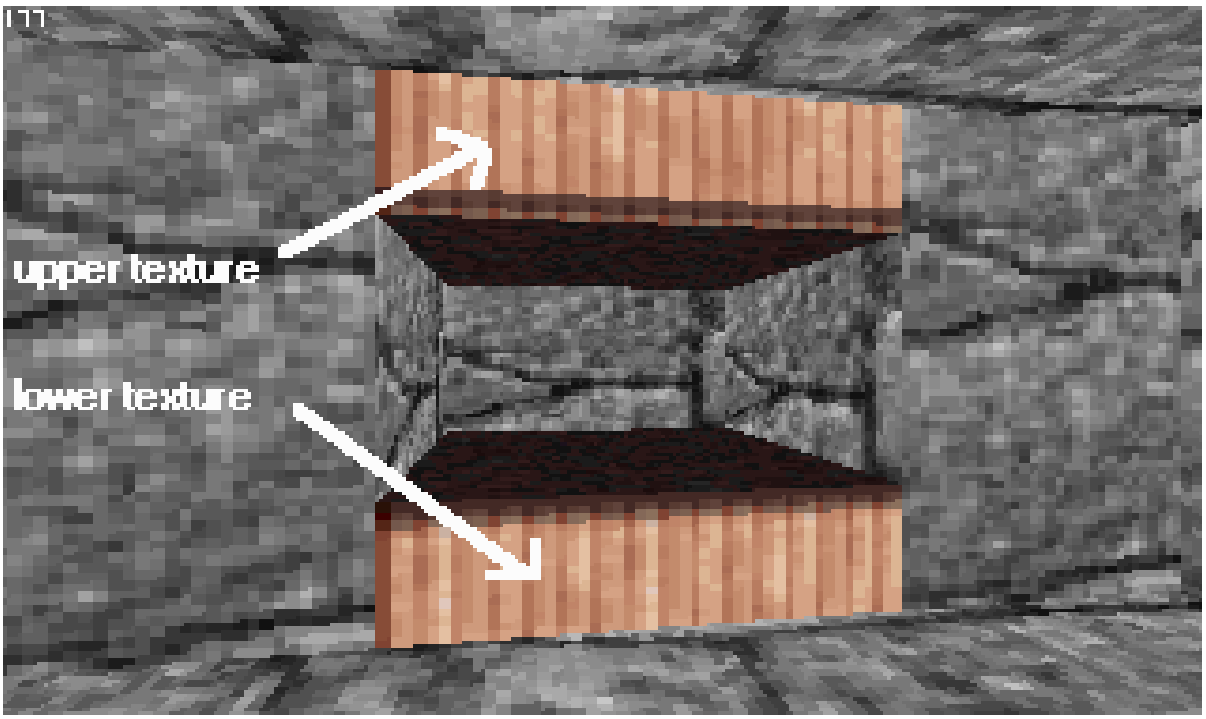


Figure 5: The adjactent sector with upper and lower textures.

14) Splitting and Joining Sectors

Sometimes it is better to split an existing sector into two rather than adding another adjactent sector. Splitting a sector is a very simple operation. All you do is draw a wall from one vertex of the sector to another one. The wall may not cross other sectors! Use the existing start sector and split the larger sector by pressing **Space** while your mouse cursor is on top of the vertex in the lower right corner and then move the cursor to the vertex in the upper left corner and press **Space** again. The wall will turn red immediately - you have just split this sector into two. Take a look at the example in figure 6.

If you prefer to have one sector instead of two you can easily join them. This is done by placing the mouse cursor into one of the two sectors and then press the **J** key. Now move the mouse cursor into the other sector and press the **J** key again. The two sectors join into one and the red border wall will be removed. Note that the remaining sector will take over the attributes of the sector you first pressed **J** in, the other sector's attributes are gone.

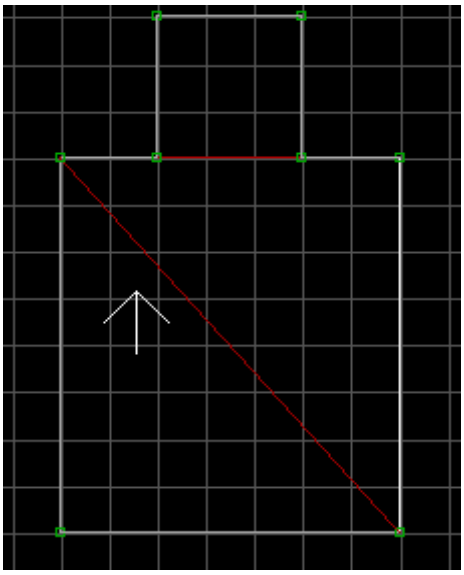


Figure 6: The start sector was split

15) Creating Child Sectors

You may wish to create a sector within a sector. This is called a child sector. To create a child sector simply draw walls only inside the parent sector as if you were drawing a single sector. These walls may not overlap with any other walls. Then press **Alt+S** to turn the area enclosed by these walls into a sector. Figure 7 is an example of a completed child sector.

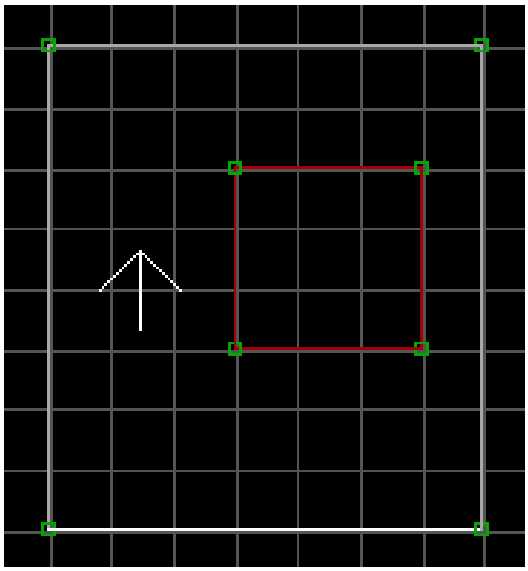


Figure 7: A child sector

When you have a look at the enclosed area in 3D mode and not the completed child sector you will notice that there is a solid column instead of a child sector. This is because the first step in creating a child sector is to carve out an area from an existing sector. This area is NULL space which the player can not enter, it is, so to say, rock solid. This operation is useful to create columns or other solid objects which go from the floor to the ceiling. Go back to 2D mode now so you can finalize the making of the child sector of this column. Move the mouse cursor into the enclosed area and press **Alt+S**. The walls of the enclosed area will turn red, indicating that you now have created a child sector which you will be able to raise/lower or texture differently in 3D mode. Try it!

16) Creating Island Sectors

Peninsular (island) sectors are similar to child sectors except that they share one or more white walls with their parent sector. Take a look at figure 8. You create a peninsula sector by starting at one vertex that is on one of the parent sector's walls and continue drawing inside the sector. To close this sector you only need to place the last vertex on another existing vertex of the parent sector. This is basically the same as splitting a sector, except that you create the splitting wall with more wall segments. You can get the same result by first splitting the sector with one wall going from one vertex to the next vertex and then inserting vertices on this wall. However it is not always possible to draw a wall inside a sector that does not cross any child sector. So it is important to know that you can make the splitting wall more complex.

17) Moving Sectors

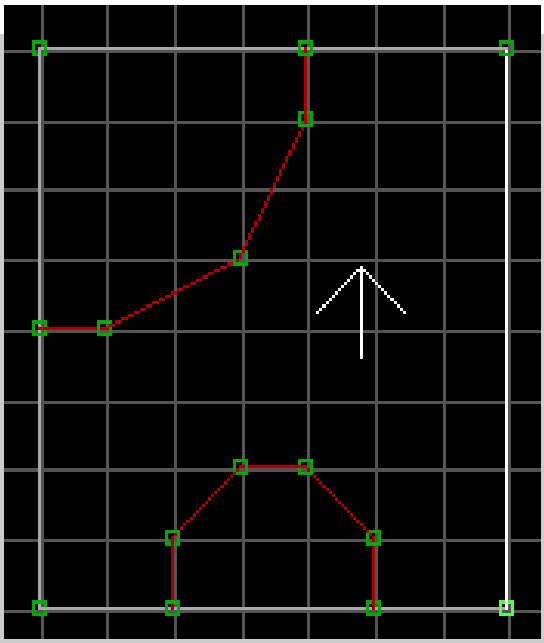


Figure 8: Two example peninsula sectors

You can drag vertices to change the shape of a sector. By dragging each vertex of a sector you can even drag the whole sector. This method is very time consuming though so there's an easier way to move sectors from one place to another. Select all vertices in order to move them all at once by holding down the **Right Shift** key and drawing a rectangle over the vertices to select. Once you have let gone the **Right Shift** key all the vertices located inside the green rectangle will begin to blink. Move one of the selected vertices with the left mouse button and the rest will follow accordingly. Once the sector is in place press **Right Shift** again to deselect all vertices.

18) Deleting Sectors

Do not delete whole sectors by dragging its vertices on top of each other until they're all gone. This is considered bad design and might lead to unpredictable results, including corrupted maps. These errors may not be visible at first but when you discover that the map has gone corrupt after putting in endless hours of work you will wish you wouldn't have done so at first. This is one more reason to not only save often, but to save under different file names as well. When developing maps save them incrementally by changing the filename every time before doing heavy modifications, starting a new area or simply trying out something which may not work at all or may not fit to the theme of the map. The proper way to delete a sector is to move the mouse cursor into the sector in 2D mode, then press **Ctrl + Del**. This will remove the sector as well as properly adjust any adjactent sectors if necessary.

19) Copying Sectors

To copy a sector select it first by holding down the **Right Alt** key and drawing a rectangle around the sector to be copied. The sector needs to be completely in the selection rectangle to become selected. After the **Right Alt** key has been released the selected sector(s) will flash green. Now move the mouse cursor in the green flashing area and hold down the left mouse button. Press the **Ins** key to make a copy of the selected sector(s). Move the mouse to the place where you want to move the copy of the selected sector(s) while still holding down the left mouse button. Release the mouse button to place the copy at the current location and press **Right Alt** once again to deselect all sectors. Do not leave two sectors directly on top of each other, this can cause problems!

Copying sectors is usually an easy task but not so for child sectors. These are sectors who only have red border walls. Copying such a sector the same way described above will lead to an invalid copy of the child sector. After moving the child sector it will appear as a sector with only white border walls. Try to make a child sector of this sector will provoke an error message and trying to delete it may even corrupt your map.
To copy a child sector prepare its destination first by drawing a „template sector" that is going to hold the copy of the child sector. Actually the so called „template sector" is not a sector but walls carving out a NULL area of the parent sector. This „template sector" must be larger as the sector to be copied, of course. The „template sector" should have the exact same amount of vertices and walls. Why this?
When copying a child sector you only copy the inner parts of the 2-sided walls, in other words the one side of the 2-sided walls facing inside the child sector. Because this will be a 1-sided wall after copying Build shows them as white walls which are NOT carving out NULL space of a parent sector. In fact leaving the child sector's copy may corrupt your map! That's the reason why there's need for a „template sector" like in figure 9. Once the copy of the child sector has been pasted into the „template sector" begin to move the vertices of the „template sector" on top of the vertices of the copied child sector so the walls turn red.

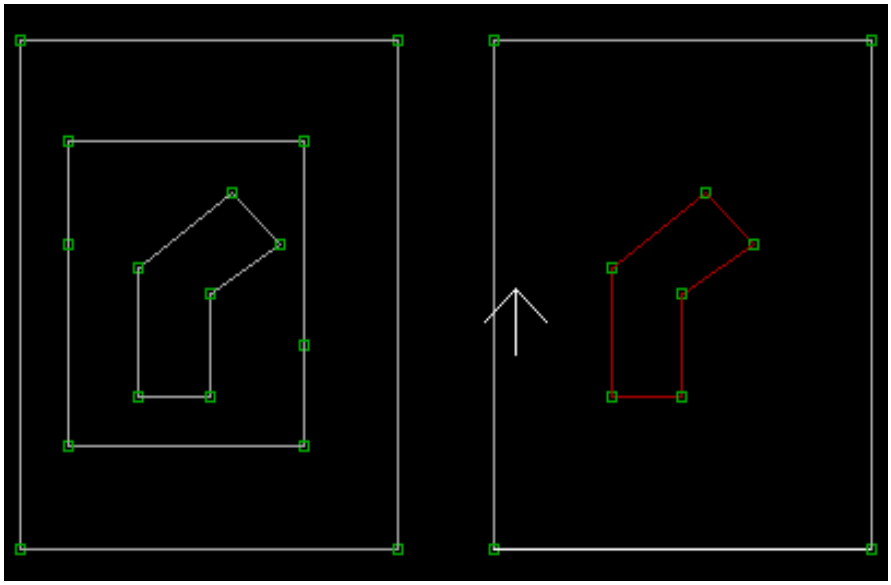


Figure 9: The copy of the child sector on the left side is enclosed by the "template sector". In this example the vertices of the "template sector" still have to be placed on top of the vertices of the copied child sector in order to complete copying the child sector.

Sector copying also works between different levels. Use the instructions for normal sector copying but after selecting and copying the sector with the **Ins** key hit **Esc** while the copy is still blinking green. Load a different level by pressing the **L** key and the selection will be copied over in the new level. Finally move the selection in place and press **Right Alt** to deselect the copy. Make sure that the current level is saved before loading a new one because Build will not ask you to save changes when loading a different level.

20) Rotating Sectors

To rotate a sector or a group of sectors you need to select the sector(s) first by using the **Righ Alt** key and enclosing the sectors with the selection rectangle. While the sector(s) is/are flashing green use the **.** (period key) and **,** (comma key) keys to rotate the sector(s) by 90 degrees. Fine tune the sector's rotation by holding down the **Shift** key while pressing either the **.** (period) or **,** (comma) key.

21) Sloping Floors and Ceilings

Sloping floors and ceilings can only be done in 3D mode. Point to the floor or ceiling you want to slope and press either the **[** or **]** key. Remember that you can always „lock" on the currently selected surface or sprite by holding down the left mouse button. In case that the slope does not go in the desired direction know that the slope anchors on the first wall of a sector. Fortunately you can change which wall is recognized as the first wall (respectively the anchor wall) of a sector. This is best done in 2D mode. Highlight the wall you want to change to the sector's first wall and press **Alt+F**. Take care on which side of the wall the cursor is when using **Alt+F** on a 2-sided (red) wall. A 2-sided wall can be a sector's first wall on both sides as well as only on one side, or not at all. Once you have changed the sector's first wall go back to 3D mode and take note how the slope changed. You might want to align the slope to an adjactent sector's floor or ceiling, do this by pressing **Alt+[** or **Alt+]**. If it doesn't work reposition the viewpoint so that you look in the direction of the slope.

22) Sectors over Sectors

Sectors over sectors means that you can have two or more stories in a level which are connected in some way and overlap on the 2D map but may not overlap in 3D space. That means one sector's ceiling needs to be lower than the other sector's floor so that one sector is located above the other in 3D space. Moreover at any time the player may not be able to see both the sector above and the sector under it because this will cause graphic problems (the so called Halls of Mirrors - „HOM" effect). Always make sure that a solid sector blocks the player's view from a sector that is virtually on top of another. One of the best methods is to use stairs that go around a corner. Drawing sectors over sectors is very simple, just draw one over another on the 2D map. None of these overlaying (or stacked) sectors may share a common wall or vertex though. Look for examples of sector over sector situations in Shadow Warrior's Bath House and Water Torture levels or Duke Nukem 3D's E3L8.MAP and E3L10.MAP levels.

23) Applying Textures to Walls

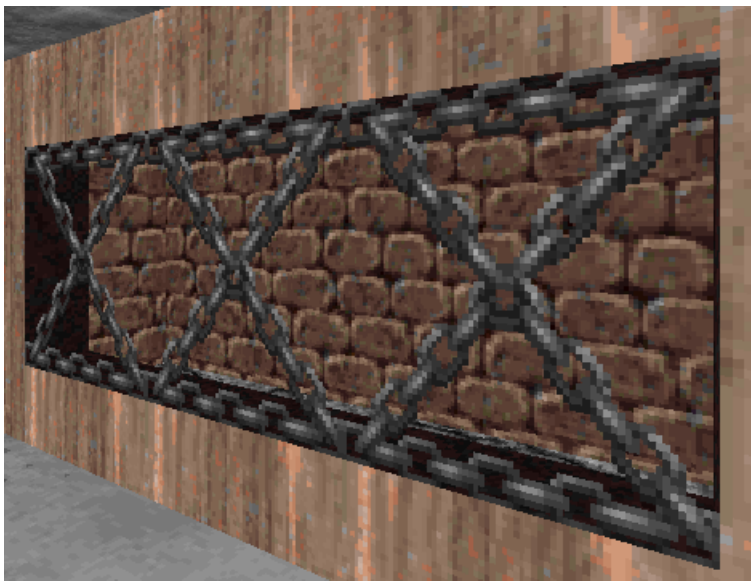
Grey bricks over and over again are getting boring? So add some color to the level by adding some more textures. First go to 3D mode and place the mouse cursor over the surface which you want to give a different texture. Press the **V** key to display a list of textures. Only one? There is only one texture? Hold your breath because this is just the list of the textures currently used in your level - to get the complete texture list press **V** a second time. Use the cursor keys and **PgUp/PgDn** keys to scroll through the list. A white rectangle, the texture selector, will be around the currently selected texture. To go to a specific tile number quickly press the **G** key to insert the tile number to go to.Press **Return** to apply the currently selected texture to the surface, or hit **Esc** to cancel. There's no need to go to the texture list every time in case you want to apply the same texture to all walls in the room. Use the **Tab** key to copy the texture under the mouse cursor into the clipboard and then press **Return** on the surface to which to apply the copied texture, or press **Ctrl+Return** to apply this texture to all walls in this sector. To only copy the shade of the texture in the clipboard use the **Shift+Return** key combination, pasting only the shade value to the surface under the mouse cursor and not changing its texture.

24) Upper and Lower Textures

Some 2-sided (red wall) walls may have a lower and/or upper part (step and ledge) that you can texture differently. For example if there's a stairway do not use the step texture as the upper texture as well because more often than not it is not looking very good. In order to edit these two parts of the wall independently press the **2** key on the lower texture. By doing so the lower texture will change to the basic grey brick texture. Now apply a different textures to this surface the same way described above by pressing the **V** key once, or twice if you want to get the full texture list, and finally pressing **Return** to apply the selected texture to the surface.
Example walls that have both an upper and a lower texture visible are walls that form a window. The texture above the window is the upper texture and the texture below the window is the lower texture. Because the Build engine draws textures from top to bottom by default the upper and lower window textures might not align properly with neigboring textures. To fix this press the **O** key on either the upper or lower texture to change the drawing orientation. Which surface's orientation should be changed depends on the sector heights in this area so the best way to get it looking good is to try it out.

25) Masked Walls

Masked walls are 2-sided walls which have a middle texture applied to it. Masked walls with no upper or lower textures look just like normal walls if all three textures are the same but you can walk through the wall if the whole is big enough for the player to pass through. This trick is often used to cover secret areas, such as wall images that are supposed to be walked through to grab some goodies there. Usually the middle texture is a partially see-through texture though like the bars in figure 10, showing a realistic looking cage. In the texture list you can determine which texture is partially see-through by watching out for purple areas in the texture. This purple color is the „see-through area“ of a texture where the Build engine draws the walls, sprites, etc. Which are virtually behind the masked wall. If you apply a partially see-through texture to a single sided wall you will see the purple area because see-through textures only work correctly on 2-sided walls or as sprites.
To turn any 2-sided wall into a masked wall go to 3D mode and place the mouse cursor on the lower texture or below the wall if there's no lower texture and then press the **M** key to add a middle texture. Depending on how you want to make it look like choose either a solid or transparent texture from the texture list by pressing the **V** key on the new texture that appeared after pressing **M**. The same texture will be applied to the other side of the masked wall. In some rare cases a masked wall is desired on only one side of the wall while the other side stays normal. Instead of pressing the **M** key alone press **Shift+M** to only turn the side of the wall facing the you to a masked wall.



Masked walls can also be made translucent by pressing the **T** key on it. However use this option with care because the translucency calculations take considerably more time than just drawing the masked texture over the background. Don't have too many translucent objects in one scenery or else this area will become slow and jerky in the game and not much fun to play, no matter how good it looks.

Figure 10: A masked wall with a partially see-through texture.

It wouldn't be very realistic if the player would be able to move through the grates in figure 10. Because of this you can make any 2-sided (red) wall blocking the player by pressing the **B** key while pointing at it. The wall will turn from red to purple in 2D mode, indicating that it will block the player from moving through it. You can make only one side of the wall blocking while the other stays normal by pointing at the correct side of the wall and then pressing **Shift+B**.

26) Working with Textures

There are a lot of operations to modify the look of a wall or floor/ceiling texture. This includes changing wall orientation, panning, scaling and aligning walls as well as changing the shade and/or palette values of a texture. All of the operations described below only work in 3D mode.

26.I) *Changing Orientation of Wall Textures*

Normally any texture will be drawn from the top of the wall down to the bottom, tiling the texture image if necessary. By pressing the **O** key on a wall it will toggle the texture drawing mode from „top to bottom“ to „bottom to top“. This is most often used on walls that have either an upper or a lower texture to easily align those parts of the wall with neighbor wall textures. The **O** key is also used on lifts and doors to avoid that the texture moves up or down together with the sector because this usually looks bad. Textures anchored on the ceiling will move when the ceiling moves and textures anchored on the floor will move when the floor moves.

26.II) *Scaling Wall Textures*

Resizing wall textures is done by using the **2, 4, 6** and **8** keys on the numeric keypad. Holding down the **5** key as well while pressing one of the keys above will make scaling walls go faster. To restore the default texture scale and pan values point to the wall texture and press the / (slash) key.

26.III) *Scaling Floor/Ceiling Textures*

There are only two different scale factors for floor/ceiling textures. Toggling between compressed and normal texture mode is done by pointing to the floor or ceiling texture and pressing the **E** key.

26.IV) *Panning Wall Textures*

Panning wall textures is done by holding down the **Shift** key and then pressing one of the **2, 4, 6** and **8** keys on the numeric keypad. The **5** key speeds up wall panning and, same as above, to restore the defaults press the / (slash) key.

26.V) *Panning Floor/Ceiling Textures*

Use the numeric keypad **2, 4, 6** and **8** keys to pan floor and ceiling textures, however there is no need to press the **Shift** key because scaling textures is done via the **E** key. Use the **5** key to speed up panning. Hit the / (slash) key in order to reset to the defaults of the floor/ceiling texture.

26.VI) *Aligning Wall Textures*

To make a set of common textures align properly press the **.** (period) key on one of the textures. Build will pan and scale the other walls automatically so that no seams should be visible anymore. This is a very powerful and useful function that should be used often because seams in a level will destroy the immersion quite easily.

26.VII) *Aligning Floor/Ceiling Textures*

This is a bit different than wall texture alignment. There's no way to automatically align two floor/ceiling textures to avoid seams other than by panning and scaling them manually. By floor/ceiling alignment the orientation of the floor/ceiling texture relative to the sector's wall is meant. To be more precise, relative to the sector's first wall. Say, if there's a rectangle sector at a 30 degree angle and a pattern texture is applied to its floor/ceiling it should align along the 30 degree angle rather than aligning on only the four orthogonal angles north, south, east or west. This can be done by pressing the **R** key on the floor/ceiling to set the drawing mode of the floor/ceiling to relative alignment. By pressing the **R** key the floor/ceiling texture aligns along the sector's first wall. To change a sector's first wall go to 2D mode and press **Alt+F** on the wall which is to become the sector's first wall.

26.VIII) *Parallaxing Floor/Ceiling Textures*

To make a floor or ceiling texture parallax (used for skies) press the **P** key on the floor/ceiling. Note that not all textures look good when parallaxed, usually only those specifically designed for parallaxing, like the various sky textures. Press **Ctrl+P** to change the algorithm used for parallaxing. Finding out which one looks best for a certain area in a level depends on how far away the parallaxed texture should seem to be.

26. IX) *Rotating Wall and Floor/Ceiling Textures*

To easily flip textures by 90 degrees press the **F** key while pointing on a texture.

26.X) *Shading Wall and Floor/Ceiling Textures*

Shading textures is very important when creating realistic looking levels. To apply shade to any surface point at it and hit the + (plus) or - (minus) key on the numeric keypad. Only the floor shade will determine how bright the player and monsters will be when they're in this sector. In addition to the + and - keys there is also the **'** (apostrophe) **+S** key combination. After pressing **'+S** Build asks to insert a shade value. In order to insert negative values assume that 256 is 0 and count backwards. A shade value of 248 actually means -8. Check the negative values by pressing **'+S** a second time but do not change it if it is correct, simply hit **Return** in that case.

26.XI) *Changing Wall and Floor/Ceiling Texture Palette*

This is often used to make one and the same texture look differently so it can be used more often without getting bored by it. Most palette numbers change the look of a texture significantly while others only affect a few pixels. For example palette values are useful to

make textures glow red in lava areas or to apply a blue-ish look to textures in pool areas. Changing the wall or floor/ceiling palette is done by pointing at it and pressing **Alt+P**, then insert the palette number. Some palette numbers might have special functions or might look differently on some textures.

27) Working with Sprites

Sprites are all other textured objects in the game besides walls, floors and ceilings, of course. Monsters are made of sprites just like trees and weapons are. A sprite is an object in Build that can be placed anywhere in the level. The texture of the sprite changes it's attributes, so a sprite with a monster texture will hunt and shoot the player, a sprite with a weapon texture waits to be picked up by the player and a sprite with a bottle texture will remain a bottle until the player shoots it into pieces. Note that in the game you can turn on 3D sprites which will effectively turn many of the 2D sprites into 3D voxel sprites. Voxels are pixels with height information, allowing the designers to create true 3D objects by arranging simple blocks of pixels at different heights. When placing weapons, ammo or health packs in Build they will normally be displayed as voxel sprites but they're still referred to as sprites. Sprites use the same texture set as walls and floor/ceilings, so you are not bound to only use see-through textures but you can also use wood textures to create bridges, for example, or other objects that seem to be solid. By cleverly arranging sprites it is possible to create bridges, even several bridges each above the other. Because this is a design trick it has also a bad side, that being the frame rate. Too many sprites in one area will very effectively slow the game down.

27.I) *Placing and Deleting Sprites*

Placing sprites can be done in either 2D or 3D mode. Point the cursor at the position you want to place the sprite and press the **S** key. In 2D mode you will see a small blue or purple point appear with a small line indicating the angle of the sprite. In 3D mode there's one special case though. When pointing the cursor on a wall and not the floor or ceiling while pressing the **S** key will place a sprite flat on the wall's surface, making it look like a wallpaper or picture. This is useful to make rooms more interesting by plastering the walls with pictures, blood stains or cracks. Delete a sprite by pointing at it and pressing the **Del** key, again this works in both 2D and 3D mode.
Note that Shadow Warrior's Build has a small bug that hides sprites in 3D mode in new levels. To get around this insert a new sprite, save the map and quit the Build editor. Then start Build and load the level. After doing so the sprite(s) should be visible.

27.II) *Moving Sprites*

Moving sprites in 2D mode works the same way as moving vertices by selecting the sprite so it is blinking and then holding down the mouse cursor to move the sprite around with the mouse. It's also possible to adjust the height of a sprite but only in 3D mode. Point at the sprite and press the **PgUp** or **PgDn** key to move the sprite up or down. Use **Ctrl+PgUp** or **Ctrl+PgDn** to place the sprite directly on the floor or ceiling respectively.

27.III) *Changing a Sprite's Texture*

Which texture is applied to a sprite changes what it is, a monster, weapon, candle or a key. Changing a sprite's texture is done the same way as changing textures of walls, floors or ceilings. Press **V** while pointing on the sprite in 3D mode to get a list of all currently used sprite textures and **V** again to get a list of all available textures. Press **Return** when you found the texture of your liking. If you choose one of a set of textures be sure to only choose the first texture in sequence, for example the monster's first moving frame or it will not animate and shoot in the game. Take a look at the table for a list of tile numbers that correspond with each monster's first walking frame in Shadow Warrior. There are also special sprite textures which are used to define sector movements and most of the effects in the game, like exploding walls or remote controlled cars. These will be explained at a later point.

Tile #	Actor
800	Hornet
817	Bouncing Betty
820	Accursed Head
1210	Sumo Boss
1300	Serpent Boss
1400	Coolie
1441	Coolie Ghost
1469	Green Guardian
1580	Little Ripper
3780	Fish
4096	Evil Ninja
4162	Crouching Evil Ninja
4320	Big Ripper
5162	Female Warrior
5426	Zilla Boss

As with any other texture a sprite's texture can be copied into the clipboard with the **Tab** key and pasted onto any other sprite, wall, floor or ceiling by pressing the **Return** key. The next time **S** is pressed to place a sprite the new sprite will have the texture in the clipboard applied to it. This is useful to place several sprites of the same type quickly. Simply copy the texture to the clipboard by pressing **Tab** after placing the first sprite and applying the desired texture to it and then hit **S** repeatedly to create several sprites with this texture with a single keystroke.

27.IV) *Different Sprite Display Modes*

Sprites can be set to be drawn in three different ways: rotating and always facing the player, flat in order to be placed on a wall and flat on the floor in order to place it, you guessed it, on the floor. To toggle through these modes point at the sprite in 3D mode and then press the **R** key. It is helpful to hold down the left mouse button while doing so to lock the sprite so that only the locked sprite will be affected, no matter which surface will come under the mouse cursor while modifying the sprite. Note that some textures look distorted when used as floor/ceiling sprites, or floor/ceiling texture in general.

27.V) *Changing a Sprite's Angle*

This can be done in both 2D and 3D mode by pointing at the sprite and then pressing the **,** (comma) or **.** (period) key. If used together with the **Shift** key it will rotate the sprite 1 degree each time. In 3D mode it is possible to flip the sprite by pointing at it and pressing the **F** key. This will flip the sprite by 90 degrees each time. In both modes there's also the **O** key available that quickly changes any sprite to flat drawing and places it on the next solid wall in the opposite direction of the sprite's current angle.

27.VI) *Scaling Sprites*

This is the same as with any other texture. Use the numeric keypad **2, 4, 6** and **8** keys to scale the sprite. If the **5** key is held down as well scaling will be faster. To restore the sprite's default scale value press the / (slash) key. Scaling sprites is often necessary because many sprites are drawn bigger than they should exist in the game because they were drawn bigger (more detailed) to avoid heavy pixelation if the player gets closer to a sprite in the game. Use sprite scaling to make realistic sprite sizes. Try not to leave any sprite in a level that seems to be too large to be realistic and try making same sprites the same size throughout the level. Note that some sprites such as monsters are not effected by scaling, their size is hard-coded in the game (programmed in the executable file).

27.VII) *Shading Sprites*

Same here, to shade sprites you do the same as with any other texture. Point at the sprite in 3D mode and press the + (plus) or **-** (minus) key, or press the **'** (apostrophe) + **S** key and directly insert a shade value. If you leave the shade value at 0 the sprite's shade value will take on the floor's shading value of the current sector.

27.VIII) *Blocking Sprites*

To toggle whether a sprite blocks the player or not point at it in either 2D or 3D mode and press the **B** key. In 2D mode you will see that the color of the sprite changes from blue to purple, meaning that it will now block the player and not let him/her move through it. If a sprite's color is blue it will not block the player. For example blood stains or very small objects should not be blocking the player while really huge sprites like trees almost always should.
In addition to blocking sprites there's the **H** key that toggles a sprite's hitscan bit. If the hitscan bit is set the sprite's angle line will be thicker in 2D mode, meaning that the hitscan bit is set so that missiles and bullets won't go through it, although the player might, depending on the blocking bit. Some sprites may only be destroyed if their hitscan bit is set. In 2D mode you can set the hitscan bit as well but with a slightly different key combination. To toggle the hitscan bit in 2D mode press **Ctrl+H**.

27.IX) *Translucent Sprites*

To change a sprite's appearance to be translucent point at it and press the **T** key. There are three modes of translucency: none, 75% and 25%. Note that translucent objects will slow down the level considerably when you place too many in one area. Be careful when and where you use this feature!

28) The End

This is the end of the „Learning Build“ chapter. After reading this chapter make a couple of test maps, trying out everything you can think of or are interested in. Please take a look at the original game maps and learn from them, for example how the designers used sectors, textures, etc. to create realistic sceneries. Use the KEXTRACT utility to extract the maps from the *.GRP file. Type „KEXTRACT SW.GRP *.MAP“ to extract all maps from Shadow Warrior’s GRP file. This might be different for other Build games.

The only thing missing is how to load levels in the game. This is usually done by using a command line switch. For example in Shadow Warrior you would type „SW -map mylevel“ to load MYLEVEL.MAP and to play it in Shadow Warrior. Note that loading maps works only in registered versions of the games! The shareware versions can not load user maps!

Once you feel familiar with the Build editor you should go on and read the references for the game you’re editing. These are descriptions of how to set up certain game effects which assume that you know the basic Build editing functions. Other references explain advanced Build level design techniques.