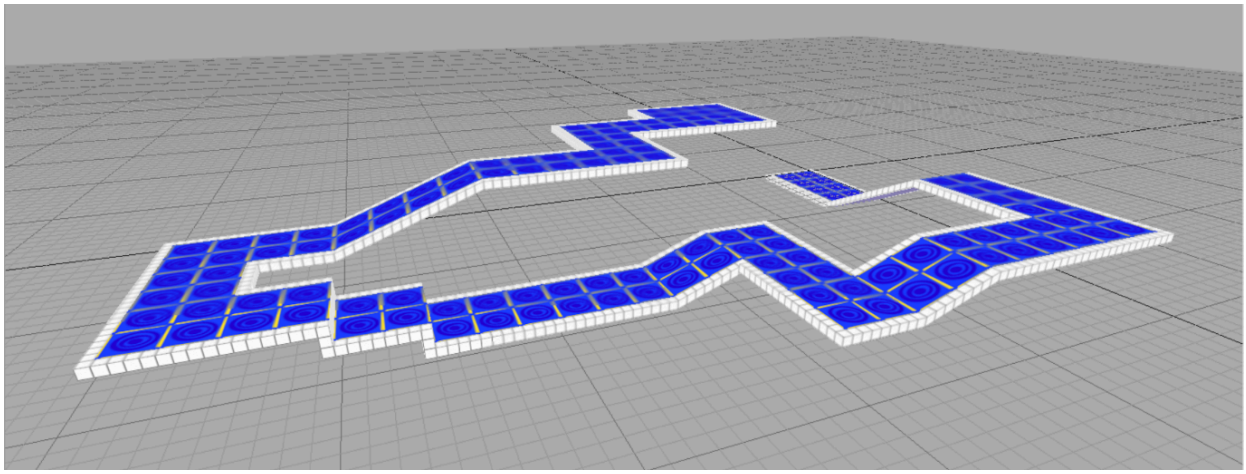
📖 Constructor Guide

Constructor Guide



Written by Nockess
2 February 2021 - 14 July 2025

Table of Contents

Intro & Download

So you want to try building your own interiors for *Marble Blast*? *Constructor* is an answer for that! This is a multi-part guide that will help explore the many functions of *Constructor*, starting with the very basics (including setting up) and building towards some more complicated and niche techniques, and exporting your maps and putting them into *PlatinumQuest*.

With the GarageGames website no longer available, you can find working copies of *Constructor* for both Windows and macOS both pinned in the #level-building channel in the community Discord and at the links below:

Windows: https://drive.google.com/file/d/1dWDA1-8bFZAApT-feKwmJlULOZ1gPkwt
macOS: https://drive.google.com/file/d/11qBQcQL-ItNh1HXB9yRjRjxLR5tyqOmM

Note: Some computers cause *Constructor* to crash excessively, rendering the program nearly unusable; this generally happens with newer hardware. The link below includes a file that will remedy that somewhat, and also allow *Constructor* to run on **Linux**. This file goes in the main Constructor folder, where `Constructor.exe` is found.

`opengl32.dll`: https://drive.google.com/file/d/1ooDVVmi_bu-ZIgCZvBLfUDFfOBacyMAJ

The guide also includes some color-coding for some of the terminology—each term's first *relevant* mention will typically be in bold lettering.

- Windows and subsequent Tabs/Sections
- Buttons
- Fields/Options
- "Texture Names"
- Key/Action

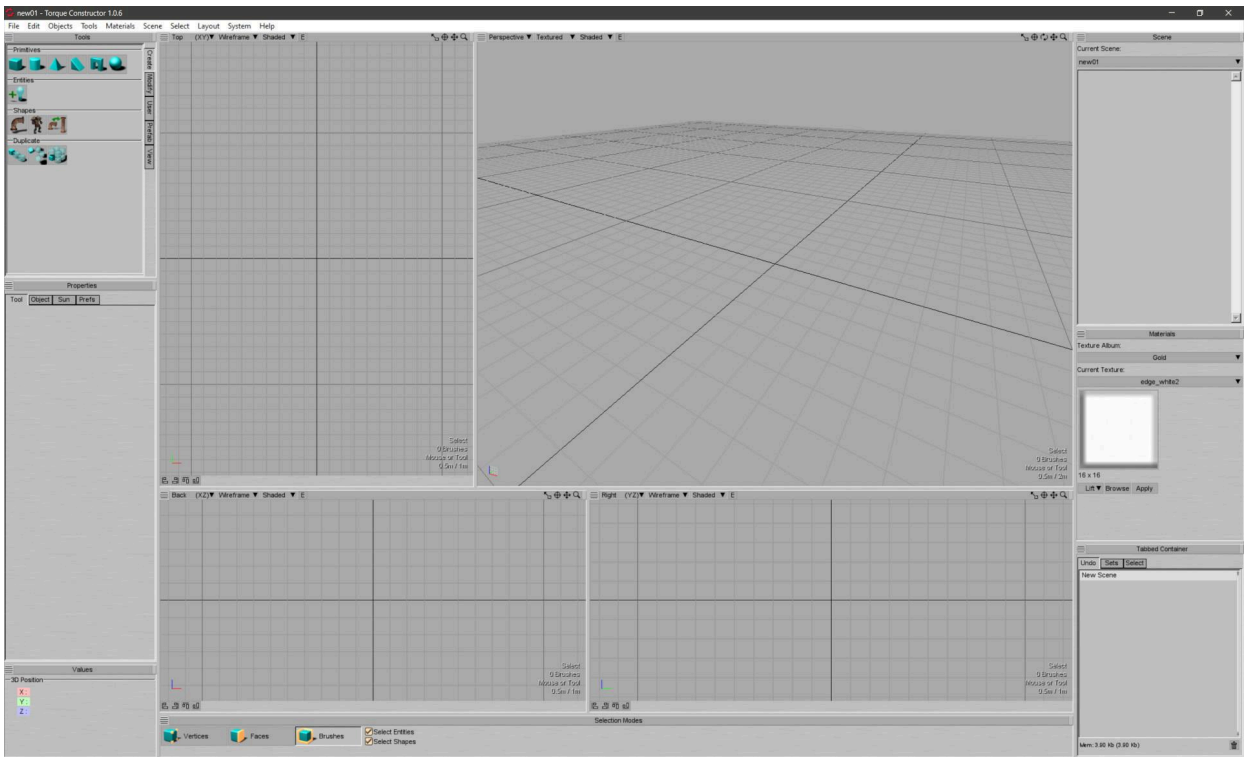
Ready? Start with Part I. Here we go!

🏝️ I: Constructor's Interface and Setting Up

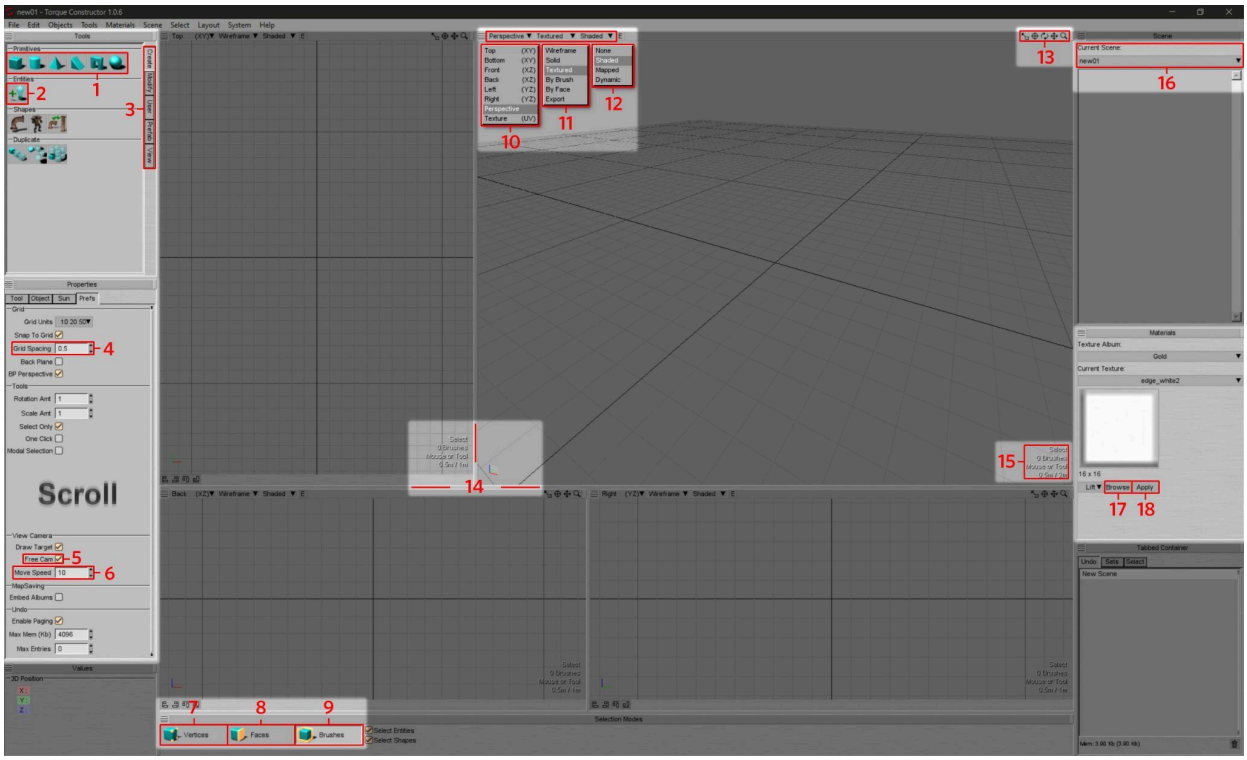
Part I: *Constructor*'s Interface and Setting Up

Before getting started, a brief overview of what you're looking at:

💻 Constructor's Interface



This is the default interface of *Constructor*. There's quite a bit to look at here, but there are only a handful of things that you'll actually use on a normal basis when building your levels. Below is another screenshot of the interface, but everything that's listed has been numbered for convenience.



Tackling everything on the left side first, starting up top in the **Tools** window. Pretty much everything you find here is related to making something.

1. These are the buttons to create brushes, which are the building blocks of your level. The only button here that is used is **Build Cube**, the one on the very left. You're totally free to use any of the other five options, but Build Cube goes a very long way to the point where that should be your main focus.
2. This is the **Add Point Entity** button. The primary use of this is to make moving platforms, which will be covered later on.
3. There are five tabs to the right: **Create**, Modify, User, Prefab, and View. The latter four have other features that will be more useful later on after you've gotten comfortable with the fundamentals of *Constructor*. Especially in the beginning, you'll find yourself sticking to the Create tab the most, given that's where the Build Cube button is.

Lower down is the **Properties** window, where we'll focus on the **Preferences** tab for now. The Tool and Object tabs will come into play once you start building. There are three things to mention here:

4. This is your **Grid Spacing**. Generally speaking, keep this at a power of 2 at all times (0.125, 0.25, 0.5, 1, 2, etc.); it is best to think of 0.5 as the "default".
5. Scroll down and you'll find the Free Camera option for the 3D Perspective View, which should always be kept on.

6. This is your movement speed for the 3D Perspective View. 10 is fairly comfortable as a default, but there's no harm in increasing or decreasing this for particularly larger or smaller levels.

.

The method of **Brush Selection** is at the bottom, which can either be controlled by clicking the buttons or pressing 1, 2, and 3 on your keyboard.

7. **Vertices** will only allow you to select the corners of your brushes, which is useful for making shapes that are not rectangular.
8. **Faces** will allow you to select a single side of a brush so you can move it or apply a texture to it.
9. **Brushes** allows you to select and move whole brushes, or apply a texture to all of its faces.

At the top of the screen, right in the middle, are the **Viewport Settings**, which you can find in the top left of any view you have open. (Speaking of which, each of the windows that display your levels are called **Viewports**).

10. The left drop-down is the point of view, where you can choose to move around in a 3D perspective or see your project in a 2D view.
11. The middle drop-down is the project's surface appearance in *Constructor*—most of the time, this should be kept at **Textured**.
12. The right drop-down is the shading options, which you can turn on to give your project some lighting effects. This doesn't actually change how your level looks in-game, and having them turned off gives your map the same amount of lighting on every side, which is usually best when you're building your level.
13. There are a few handy buttons over on the top right as well. From left to right, you can maximize and re-shrink the respective viewport (far and away the most commonly used out of these), focus on a selected brush, freely rotate your camera, shift your camera without changing rotation, and zoom in and out.

A couple more miscellaneous things to mention:

14. On the edges of each of the viewports are the Viewport Bounds. By clicking and dragging them, you can make one big window, or have two medium-sized windows, or have a 2-by-2 formation seen in the above image, or whatever you prefer.
15. This is the selection info. You can see how many brushes you have selected at a time, and the bottom numbers are your current grid spacing versus the current scale of the grid.
16. This shows the active project that you're editing. If you have more than one project open at a time, you can open the drop-down menu and change maps.

Finally, the **Materials** window has everything you'll need to know about the textures you're using.

17. The **Browse** button will pull up the **Texture Browser**.
18. The **Apply** button applies the current texture to the brush or the face you have selected.

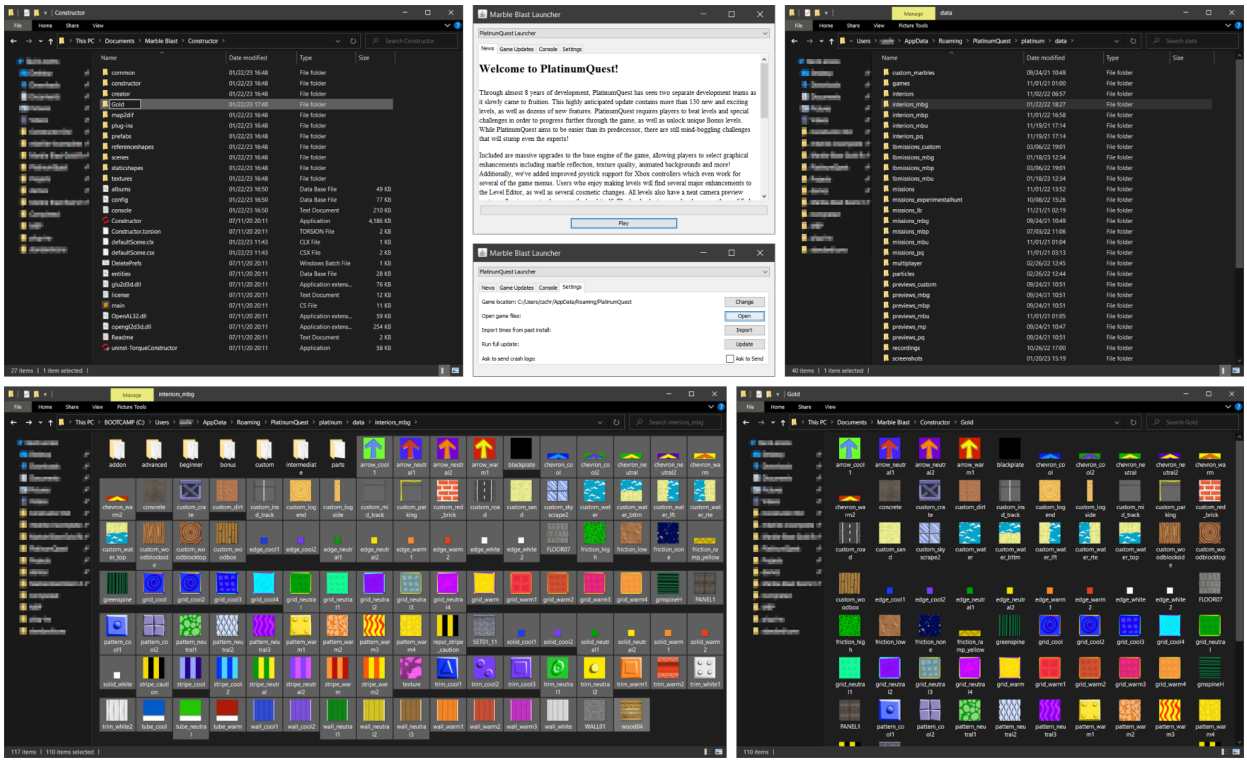
That is just about everything that will be most handy to you now, and while there's still plenty to point out, the later parts will cover those things. Stay with the Materials window on the right side, where you can set up the texture you'll use to build your levels.

⚙️ Setting Up (Adding Textures)

The provided copies of *Constructor* are already set up with the necessary textures. There is a folder and an album for *Marble Blast Gold*, *Ultra*, *Platinum*, *PlatinumQuest*, and even one for the extra multiplayer textures. If your plan is only to make levels for those four games, where you won't need to add any more textures, you can skip the rest of this section of the guide and find a link at the end for the part that dives into building. On the flipside, knowing how to do this will be helpful if you either want to add textures from different games, add your own textures, or if you simply just want the knowledge.

This explanation will only include instructions for installing *Gold* textures, as those are the textures used for the entire guide. If you're able to install the *Gold* textures correctly following these steps, you'll be able to do the same for the other mods. It is important that you do not mix and match mods though. Keeping the folders separate prevents overwriting similar textures that happen to have identical names, such as "edge_white2".

First thing's first. Create a new folder in your main *Constructor* directory where you'll store the *Gold* textures and your *Gold*-themed levels. We have to go fetch the textures from the *PlatinumQuest* game files, which can most easily be found by using the launcher, and going to the Settings tab to the Open button. Once the file window opens, go to platinum/data/interiors_mbg to find the textures. Copy all of those into the folder you just made.

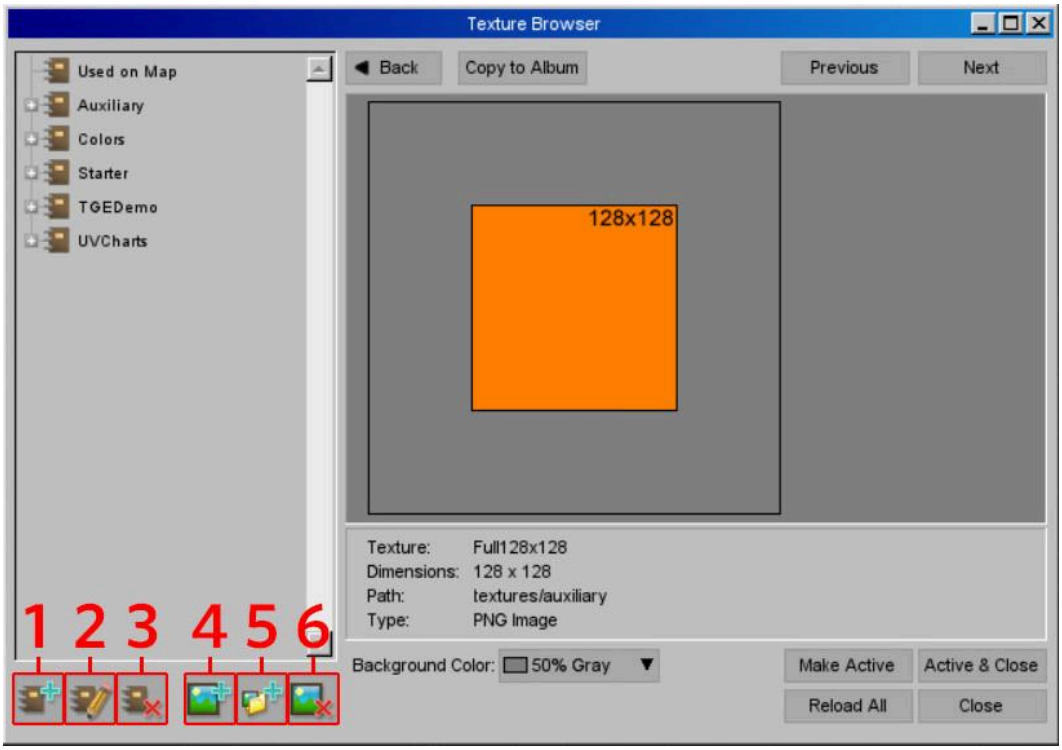


(Read images left to right, top row to bottom row.)

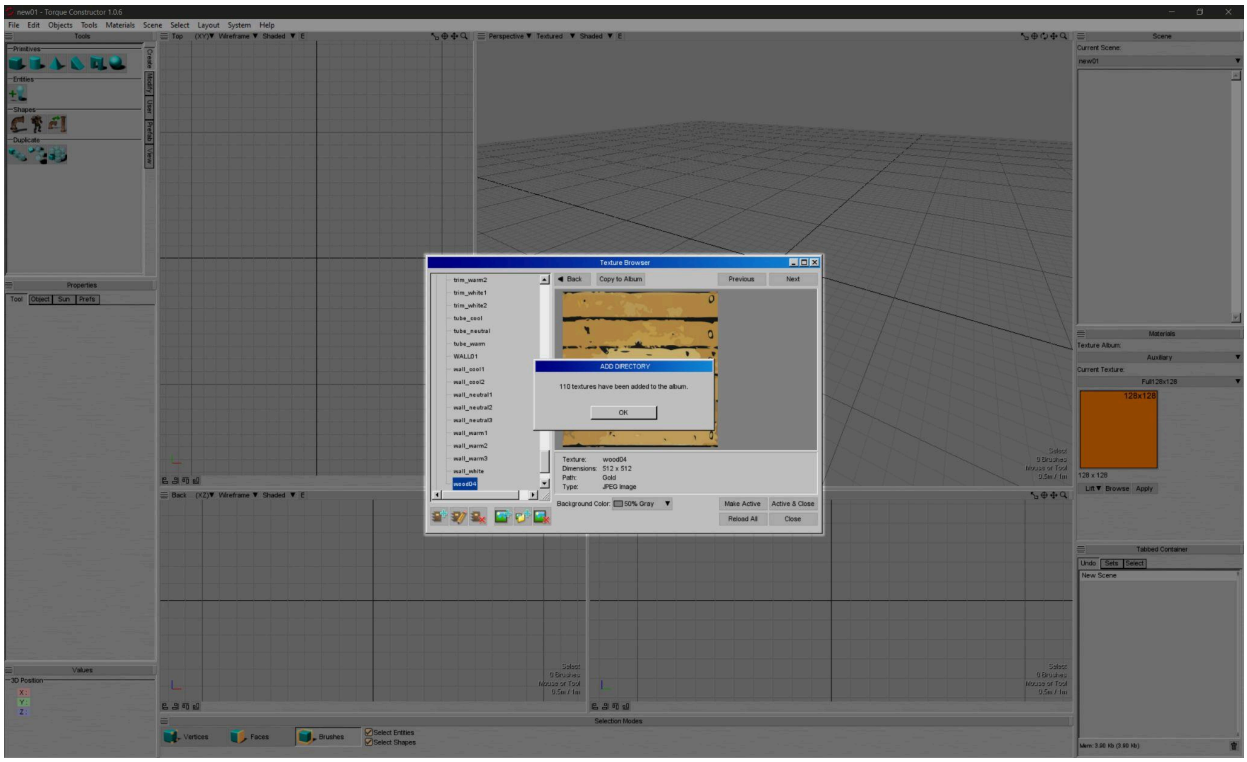
Return to *Constructor* and open the Texture Browser by clicking the Browse button in the Materials window. These six buttons in the bottom left are the main focus of this new window. From left to right, you have:

1. Create a new album.
2. Edit the name of an album.
3. Delete an album.
4. Add a texture to an album.
5. Add a folder of textures to an album.
6. Delete a texture from an album.

Create an album for the *Marble Blast Gold* textures. It is crucial to NOT include any spaces in the name of your album or it will not work. It's best to stick with a simple name like "Gold" or "MBG".



There are two ways to add textures: either one-by-one, or from an entire folder. Obviously it is more efficient to add the entire folder of textures we have into this album all at once rather than individually, so hit the fifth button to bring up the window that will set you up to find the folder you made with the textures. Be patient while the textures load! Constructor may freeze for a few moments as it works, but let it do its thing and everything will be added soon enough.



When you're finished adding textures, hit Close, and now you should be fully set up. With a basic understanding of the *Constructor's* default interface, and all of the textures imported for use, you can move onto building a level.

📝 Summary

- There are four different windows to work in, and the one in the top right is the 3D perspective that is used most often.
- The majority of the settings and building options are found on the left side. The texturing options are found on the right.
- The provided copies of *Constructor* already include the needed textures for the four core series games (*Gold*, *Platinum*, *Ultra*, and *PlatinumQuest*), but the user can add more textures and create more albums to store them in.

🧮 II: Basic Building and Polishing

Part II: Basic Building and Polishing

This part of the guide will cover movement, basic building and texturing, and exporting. If you've followed from the very beginning, you should have an idea of what's in front of you in the interface, and you should have all the textures installed and ready to use.

🧭 Moving Around

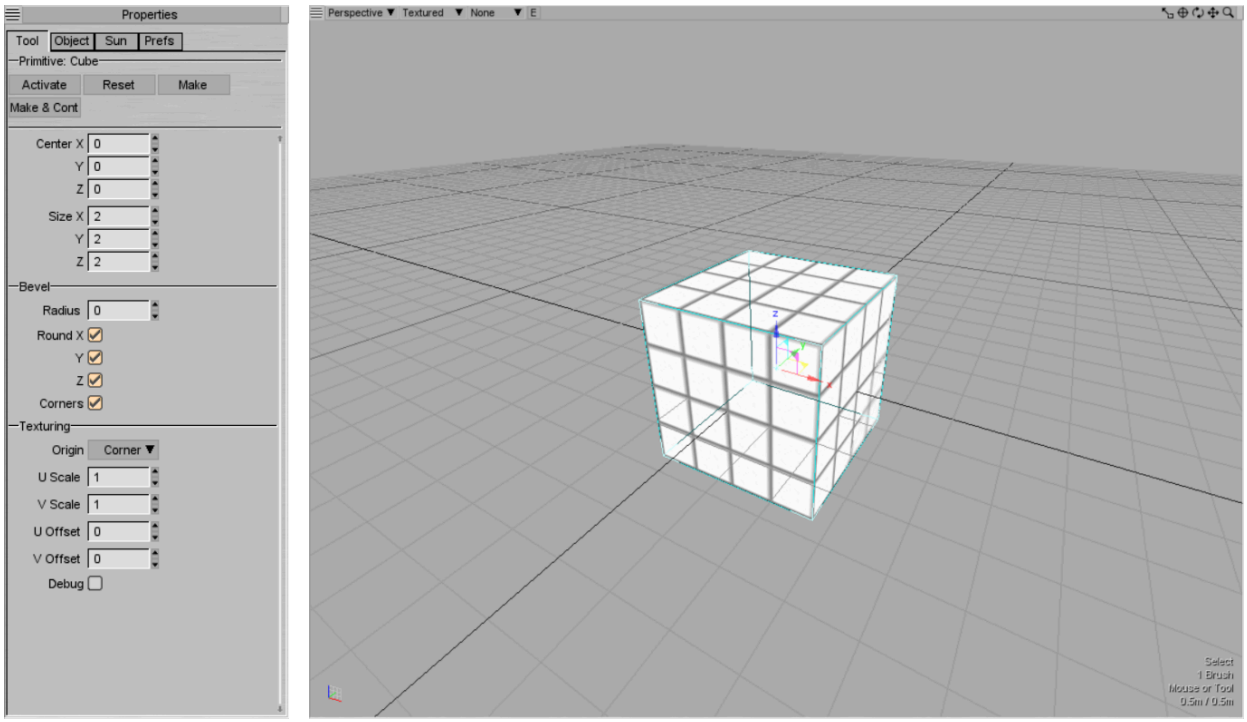
Moving around is easy to get comfortable with given its simplicity and similarities to movement in *Marble Blast*.

In the 2D views, hold right-click and drag your mouse to move around. In the 3D perspective, hold right-click and use the WASD keys to fly around, and use your mouse to move the camera around. You can also hold Shift to move faster, and like mentioned in the interface, you can change the speed of your movement in the Preferences tab in the Properties window.

The scroll wheel zooms you in and out in every view. The farther you zoom in, the less of the grid you will see in the 3D perspective; you may find it helps with visibility, particularly with brushes "below sea level".

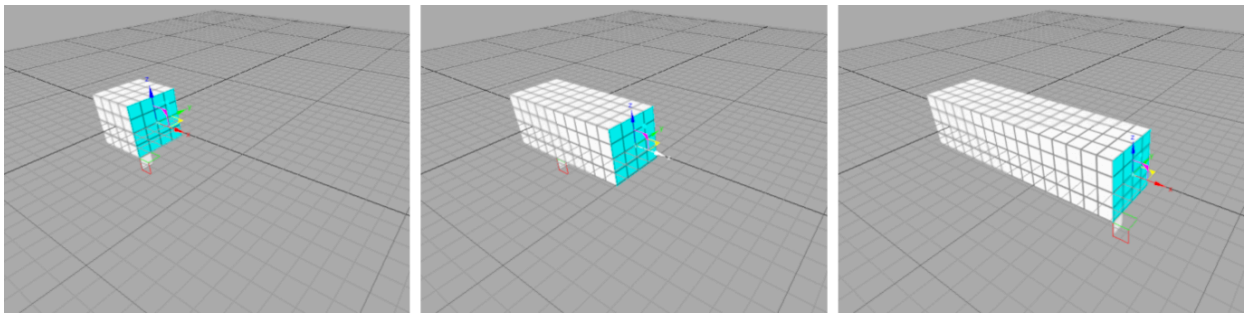
🏭 Building

Now it's time to start for real! First, select a texture to use. Using a **trim** texture (most commonly called "edge_white2") is often a great way to begin, so find the Build Cube button in the top left.



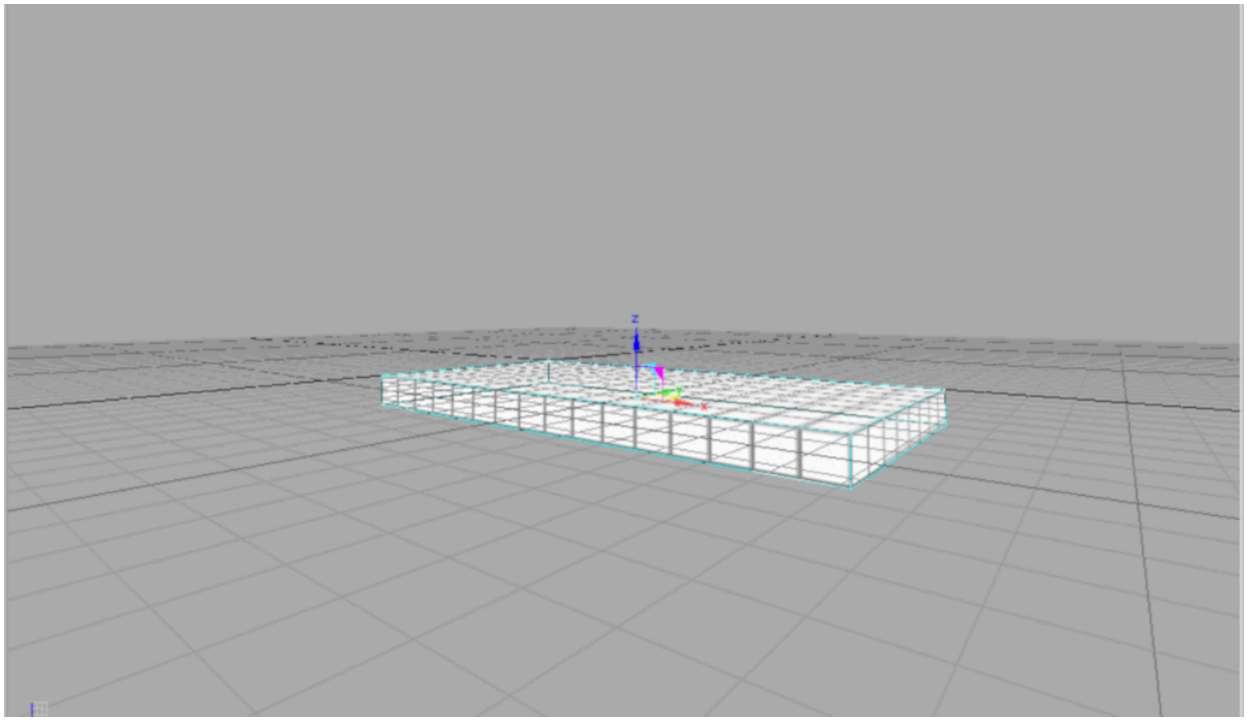
The Properties window has automatically gone to the Tool tab, which you'll use all the time. Start by setting the size of the **brush** to 2x2x2, and setting the center of the brush to 0, 0, and 0. Also make sure that the U and V Scale in the Texturing section is 1 and 1. To finalize the cube, hit either the Make button or the Build Cube button again.

Select Faces mode (either by clicking the button or pressing 2), and resize the brush to be 16 edge tiles long, 8 edge tiles across, and 1 edge tile high. Resizing a brush in Faces mode is as simple as selecting a face, and dragging it in or out using the arrows on the axis. No huge deal if your brush is off-center in any way, but if you care about neatness, it's good to have one of the four corner (vertical) edges on 0, 0, 0, otherwise called the origin. Move the brush by selecting Brushes mode and using the axis.



For the sake of being neat, it's not a bad idea to position the brush right below the grid. This is when you can use the 2D view to tell when the brush is in the right place. A good

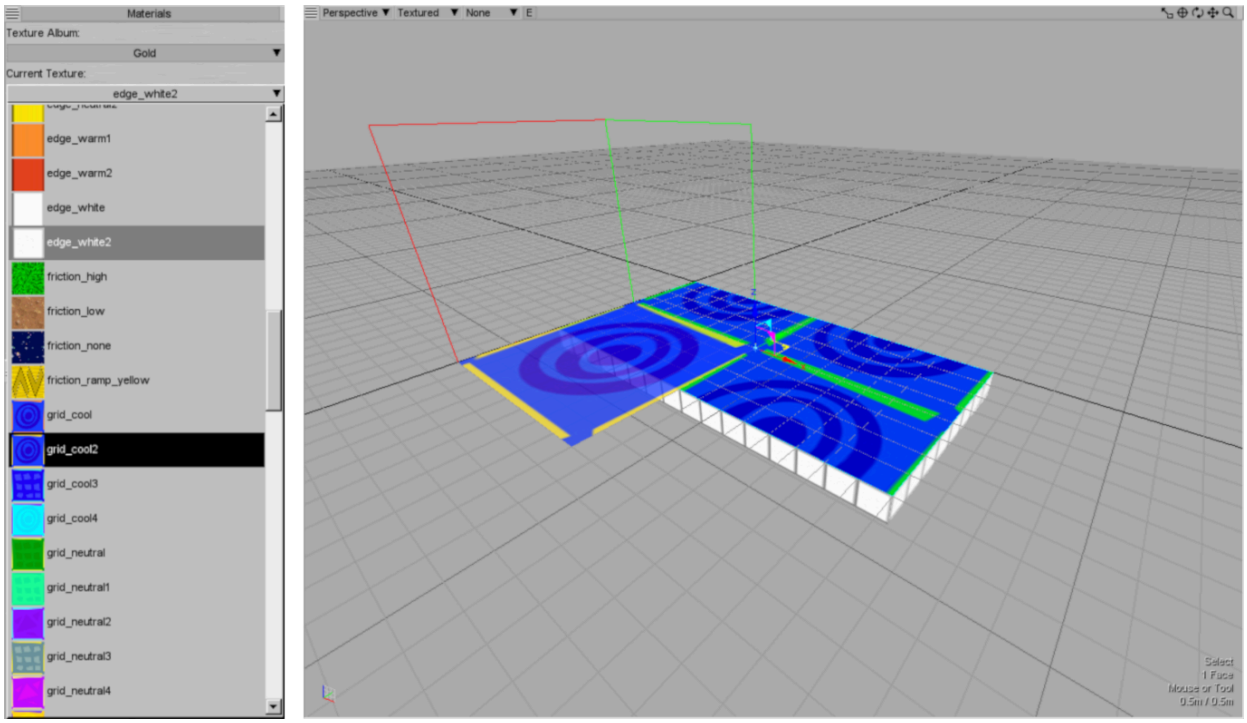
thing to keep in mind is no matter how far you zoom in or out on any of the 2D views, you will always see the axis that intersects with the origin.



🎨 Texturing

This isn't quite what we want our platform to look like, so let's work on changing that.

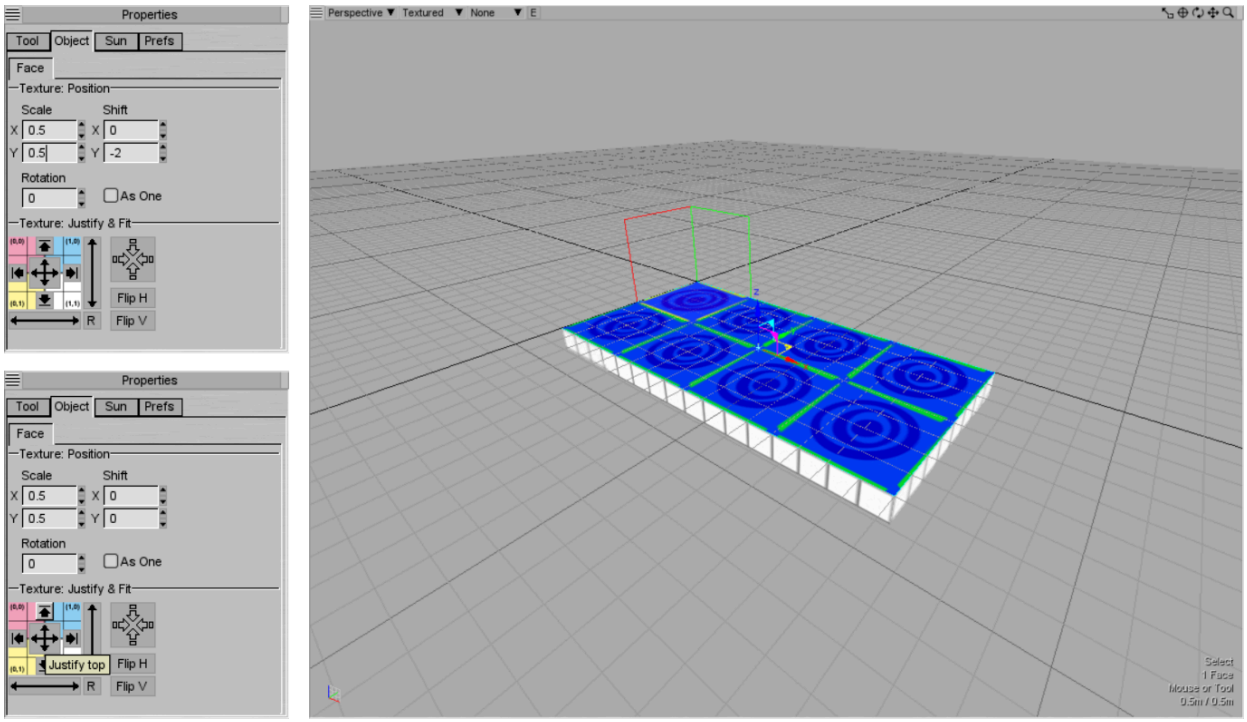
Going back to the Materials window, first select a floor tile texture, which is what you're used to rolling on when you're playing. Contrary to last time, instead of hitting Browse, use the drop-down menus for a more efficient process. The top one is for the texture album, which is already set to Gold from earlier, and the bottom one is for the textures themselves. Open the drop-down menu by clicking on it, and selecting the texture you want to use. Make sure you're still in Faces mode when you're applying the texture, or you'll change the look of the entire brush. Hit the Apply button, and watch the brush immediately look a bit more like a normal platform!



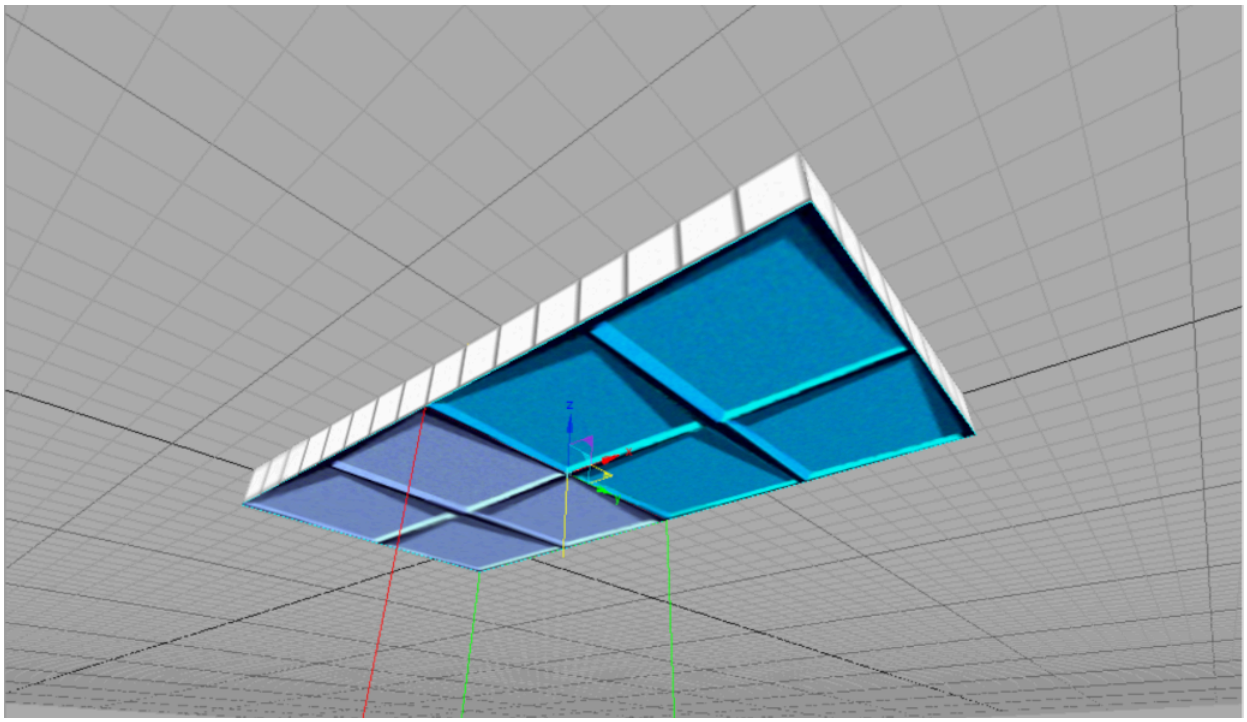
Texture: grid_cool2

The floor looks a bit bigger than what we want it to be though. Fortunately, when you have a face selected, texturing options should show up in the Properties window in the Object tab.

Set the **scale** to 0.5x0.5, which is what almost all floor tiles should look like. If your texture still doesn't look quite right, use the Up and Right Justify arrows to align the textures, which will change the **shift**.



Deselect the face by clicking in empty space in any of the viewports. For the sake of being neat again, you can choose to texture the bottom of the brush as well. Traditionally, that is a wall texture (scaled to 1x1) or the same floor tile texture on the top side.

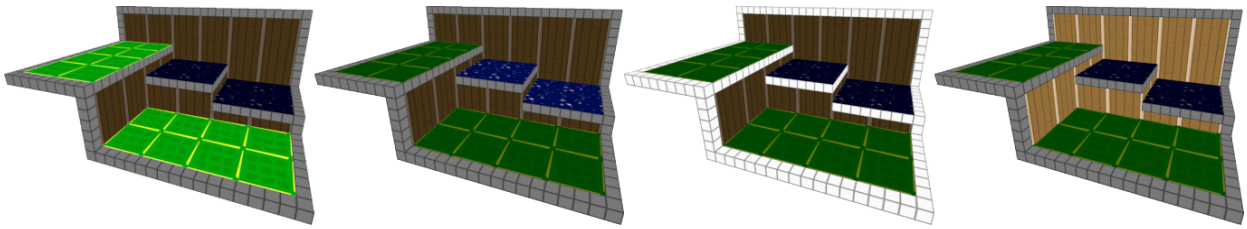


Texture: pattern_cool2

A tip for texturing: it's never a bad idea to position your first brush so you can set the shift to 0 and 0, and then build from there. If clean texturing and alignment are important to you, you can periodically select every face in your project (Ctrl/Cmd+A) as you build and set the shift to 0 and 0 again.

🔍 Texture Scaling

Something good to keep in mind are what the best scales to use for textures are. Of course, all texturing is subject to personal taste, but there are definitely conventions that you can follow, especially to match the look of the original game (and, for this reason, the rest of the guide will follow these conventions). The listed values apply for both the X and Y scales.



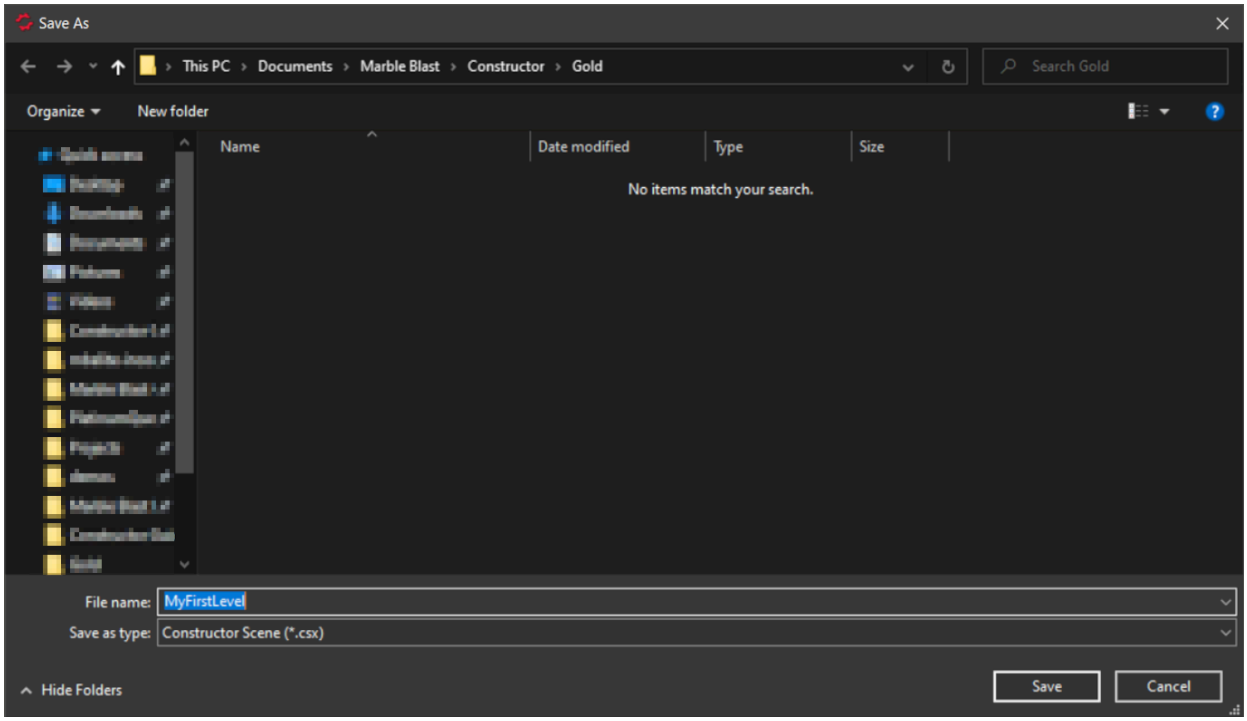
Floor Tiles: 0.5
Frictions: 0.5
Trim: 1
Walls and Wall Tiles: 1

Something that's good to keep in mind that the length of 4 edge tiles is equal to the length of 1 floor tile. Textures that are variants of trim or walls should usually be matched to fit the scales given above.

📁 Saving

Like any other creative project, it is crucial to save periodically! The two options are to save as a .csx file or a .map file—for now, do the latter and save as a **.csx** file. There will be more detail about the different file types in Part III.

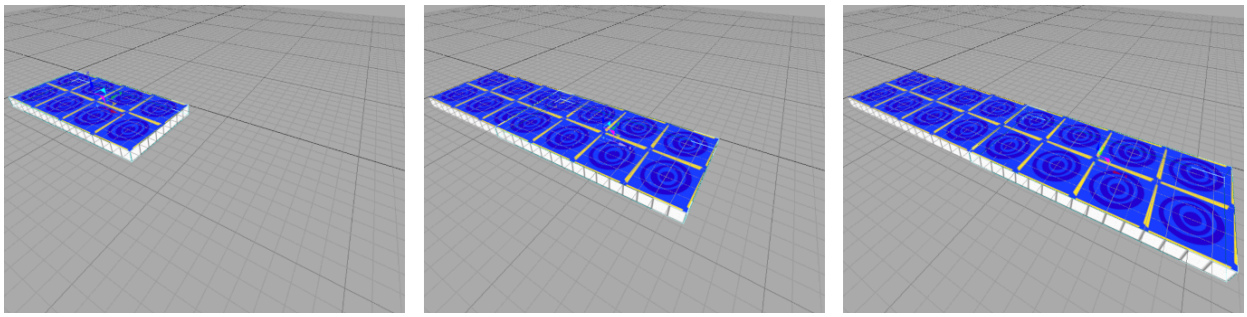
Save your project into whichever folder contains the textures you used. In this example, since the project has been using textures from *Marble Blast Gold*, the project is saved into the *Gold* folder.



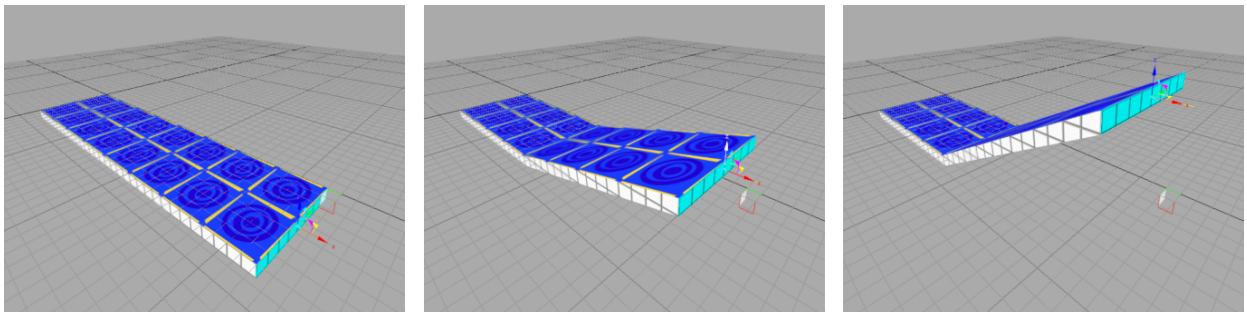
🔨 Continue Building

That's a great start so far—time to continue and finish the level!

Now, a new platform. That can't be done with a single brush, so go to Brushes mode for duplicating the brush. You can either use traditional copy and paste methods (Ctrl/Cmd C+V), which will place the new duplicate in the exact same position as the original, or you can hold Shift and click-drag on any of the three axis as a slightly faster method, which is what is shown above.



This new duplicate will be turned into a ramp. Select the side face on the end of the brush and move it on the Z axis to make a slope. The example shows the edge being moved up by 8 trim (the equivalent to 2 floor tiles).



The sides of the ramp could look better, but come back to that later on (Part IV)—for now, finish the level.

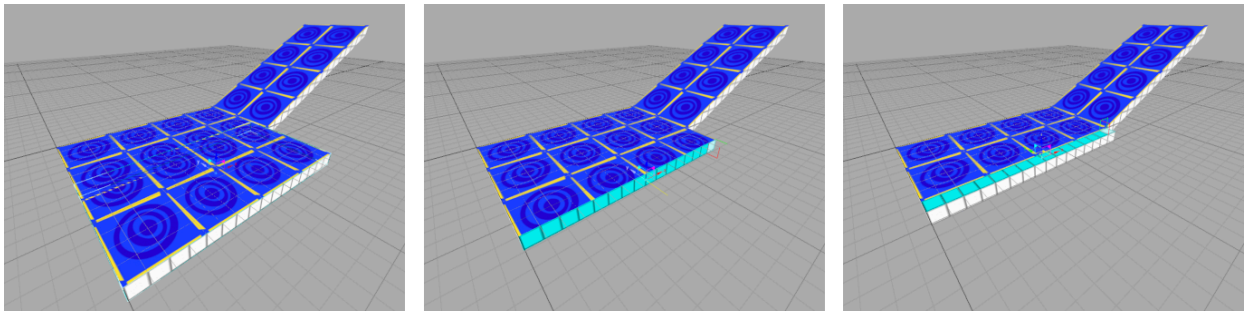
Important tip! The square bracket keys are a great shortcut for adjusting your grid spacing. `]` will double your grid spacing, while `[` will cut it in half. Familiarize yourself with these, generously.

🧹 Polishing

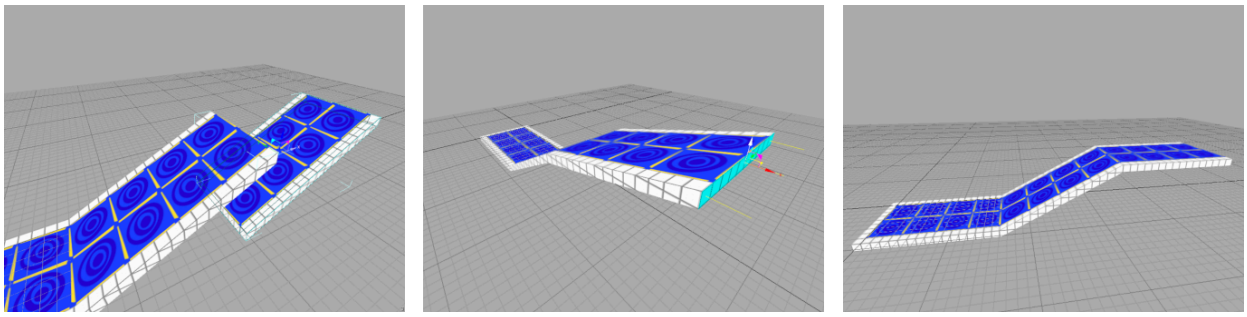
A couple more things to quickly explain before you have everything you'll need to know as you get comfortable with the very basics of building.

Platforms are traditionally surrounded by trim to give them a much more polished look. To start, duplicate your platform using the Shift-and-drag method, resize the platform to be one trim wide, and make sure all six sides of the brush are set to the trim texture, which can most efficiently be done by applying the trim texture when you are in Brushes mode. Be sure to set the scales of all the faces (at least the ones you can see!) to 1x1,

and if you need to, align everything with the justify arrows. Keep lining everything with trim all the way to the end of the slope.

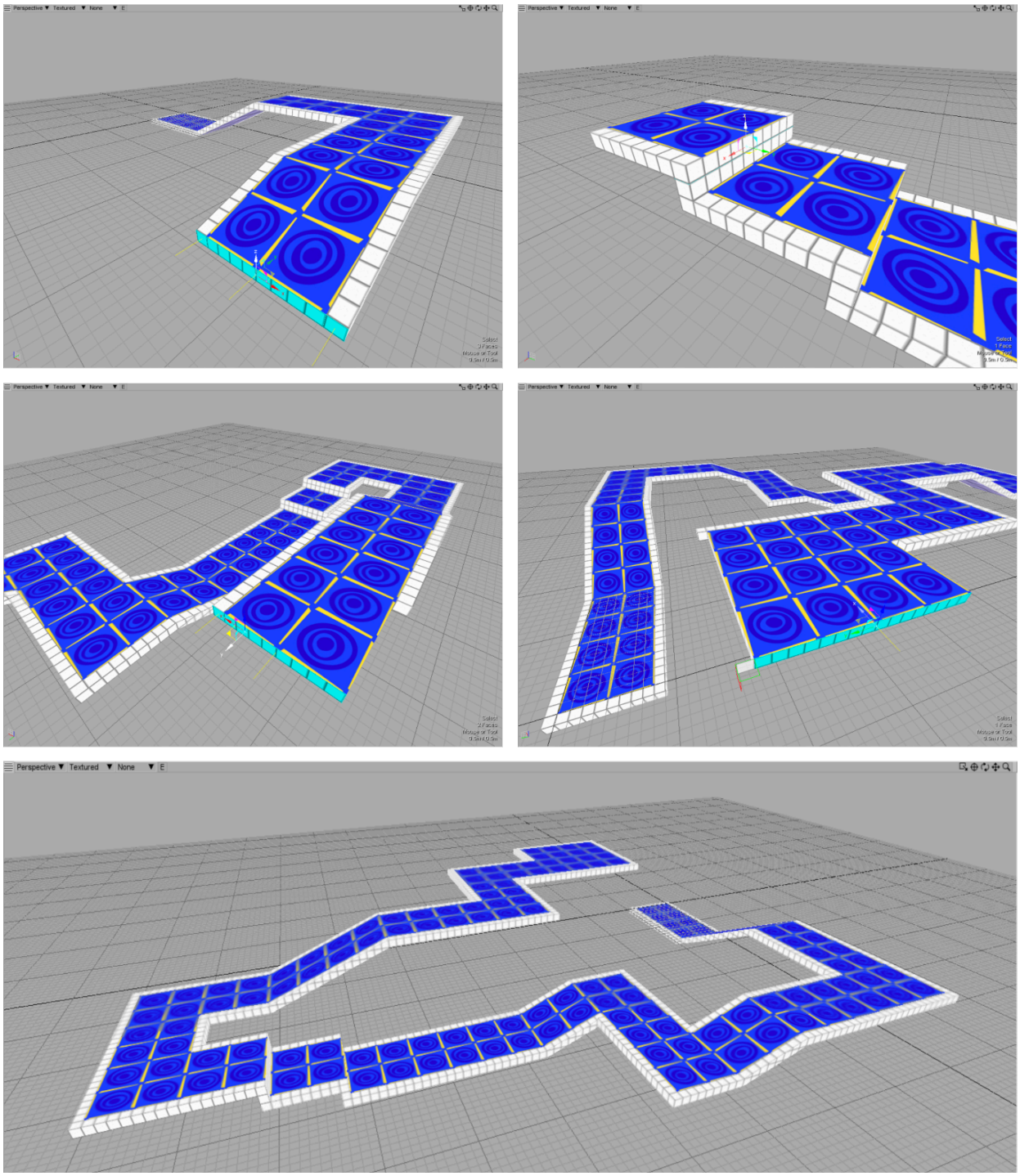


It is possible to duplicate more than one brush at a time, simply by selecting multiple brushes before duplicating normally. If you use the three brushes that make up the ramp to continue the path that's starting to form, you can make the new part of the platform flat again by selecting all of the faces on the end and dragging them down.



That is essentially everything to cover regarding the very basics of building levels, and you are welcome to export now if you choose. However, if you're feeling up for it, keep building for a little while and see what you can create with your new skills! Take some time to practice duplicating and moving faces around to make an interesting path, and continue to become more comfortable with Constructor's workflow.

The pictures below show the progression of how the pictured example project was finished. When you are ready, it's time to put your project into *Marble Blast*.



📝 Summary

- To move around, hold right-click and use WASD (in the 3D perspective) or drag the mouse around (in the 2D perspectives). Scroll to zoom in and out.
- Use the Build Cube button to create brushes, and use Faces mode to resize them afterwards.
- To change a brush's appearance, open the Materials window to select a texture; the entire brush can be changed to that texture (in Brushes mode) or just a singular face.
- With a face selected, the size and positioning of a texture can be adjusted in the Properties window.
    - Use 0.5 scale for floor tiles and 1 scale for trim.
- Use the square bracket keys to quickly change your grid spacing.
- Save!

📬 III: Exporting and Loading Your Interior

Part III: Exporting and Loading Your Interior

This part of the guide will cover exporting and loading in the level you just made into the game. If you've followed from the very beginning, you will have learned about *Constructor*'s interface and built your first project.

📑 Different File Types

Before exporting, some information on the two file types, `.csx` and `.map`.

Generally, one would find it best to use **`.csx`** most of the time, as it is most forgiving by far when it comes to exporting a project for *Marble Blast*. They are like detailed blueprints, in that they explicitly describe the shape of each brush using exact vertex positions and which faces connect them.

On the other hand, **`.map`** files describe shapes in a more "abstract" way, by using planes (flat surfaces) to define the shape. When these planes intersect, they form a convex brush—a solid where every face curves outward. Because of this method, .map files can't represent complex or broken shapes (malformed brushes, mentioned later)—only clean, convex geometry.

If that was a lot to read, here's a quick summary:

- `.csx` - exact, flexible, stores full geometry (positions and faces)
- `.map` - simplified, only supports clean convex shapes, stores the planes that defines them

Finally, **`.dif`** files are what *Marble Blast* recognizes as platforms, or to use the correct lingo, the "interiors" in levels.

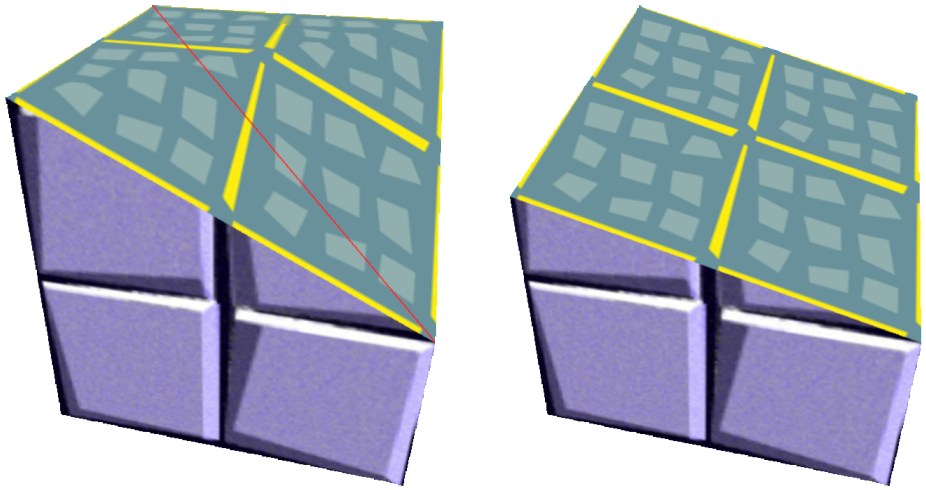
Historically, `.map` was the preferred file type for most of the time that people have been making levels as the only conversion tool relied on `.map` (called map2dif). However, thanks to the efforts of RandomityGuy in 2022, there is now a tool to convert `.csx` files for use in *Marble Blast*.

Before exporting, a quick note about malformed brushes:

⚠️ Malformed Brushes

Malformed brushes will have a very obvious red line going through the middle of them. If your project is saved as a `.csx` file, these are no problem—they will convert without any issues when using csx3dif.

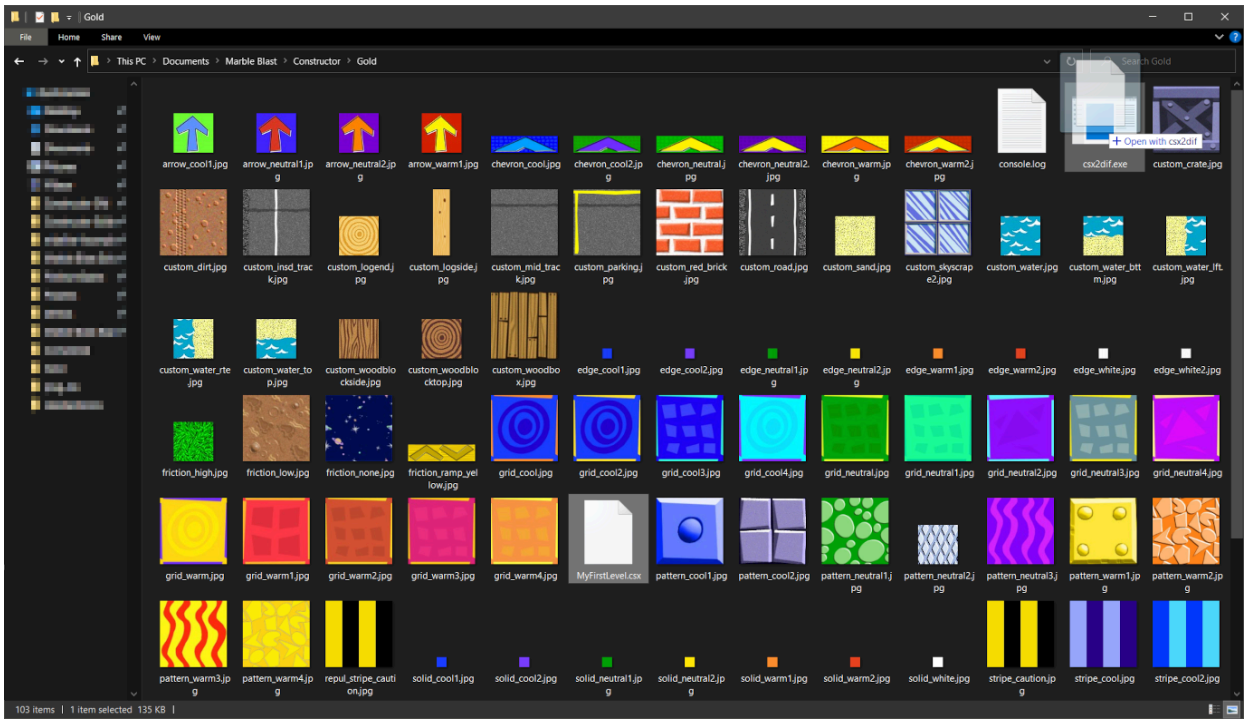
However, they do not agree with .map files. If a malformed brush exists in your project, it will disappear upon reloading your file. They can often be fixed with some careful use of the Knife Tool, but this is less of a concern with `.csx` files generally taking priority over `.map`.



🏅 *csx3dif*

For Windows users, **`csx3dif.exe`** can be found in each of the folders provided in the copy of *Constructor* at the beginning of the guide.

Navigate to the folder where you saved your project, and drag it onto `csx3dif.exe`. A new file will appear next to the `.csx` file you just converted, but now as a **`.dif`** file.



UPDATE PHOTO!!

If you're feeling a bit crafty, you can also convert `.csx` files just with a double-click. If you open the context menu (by right-clicking on the file), and navigate to Open with, you can select (any copy of) `csx3dif.exe` and obsolete the need to click and drag.

RandomityGuy has also produced a website version of *csx3dif*, which is the functioning alternative for macOS users. There is a prompt to choose the .csx file you would like to convert, and the website will return to you a downloaded file.

🥈 ***Map2Dif***

Using *Map2Dif*, while being the primary method that latest through years and years of level building history, can be a bit of a train wreck.

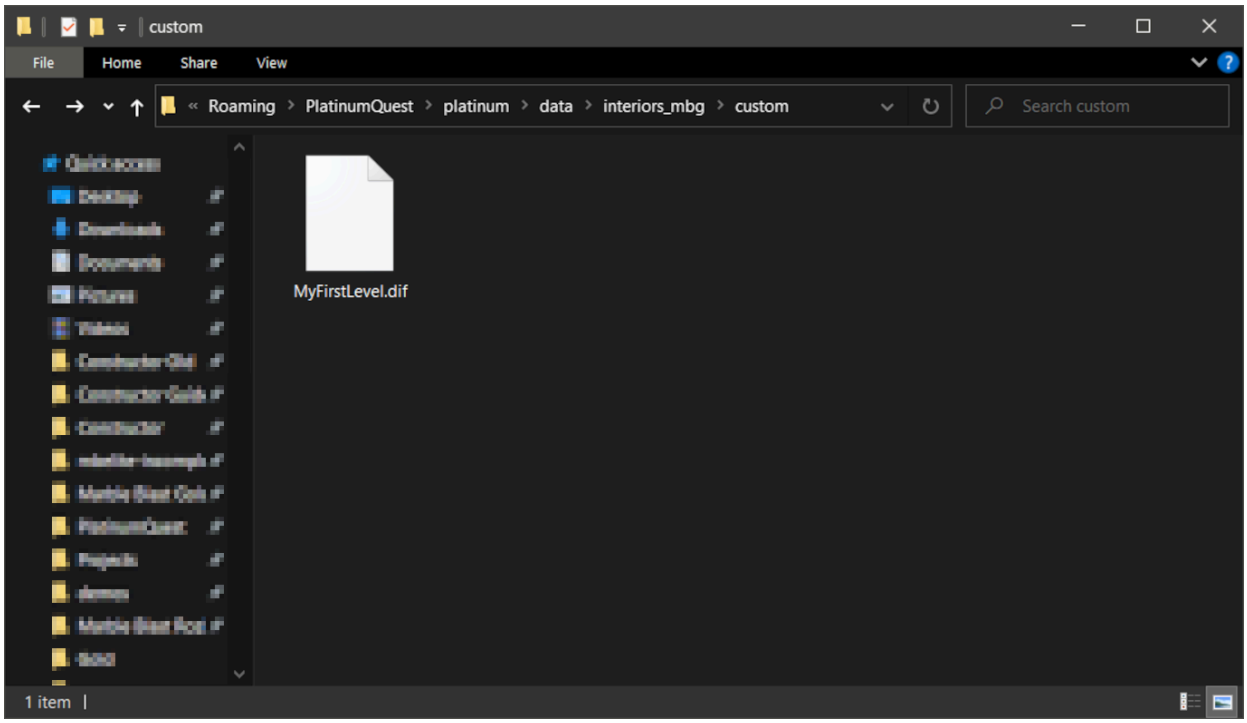
Frankly, it's a fine solution if the map you've built doesn't have super complicated brushes and instead consists of more basic shapes. However, there are two things to try and avoid: unusually large brushes (it's safe to stay under lengths of 32 tiles or so) and curvy objects like spheres and toruses. *Map2Dif* has a history of being fragile and unreliable (truly an understatement in some cases), but again, it was the main way to convert levels for years. It works perfectly well to some extent!

With that said, exporting with *Map2Dif* is the same process as with csx3dif, by dragging the desired .map file onto `map2dif_plus.exe`.

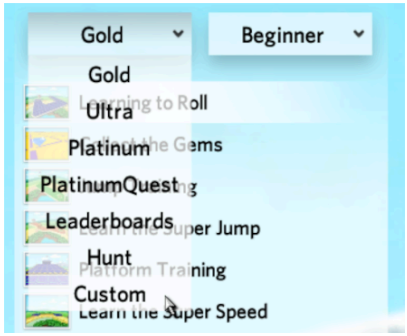
🎢 Loading An Interior

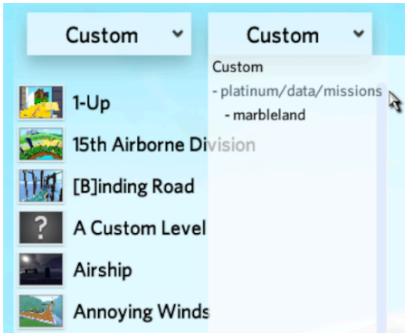
To navigate to *PlatinumQuest*'s directory, there is a button in the launcher called Open in the Settings tab.

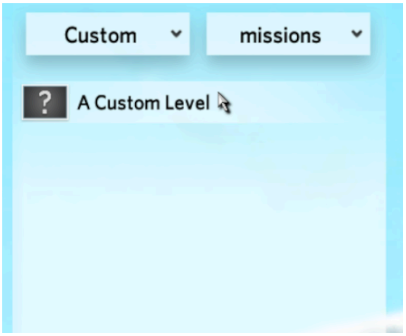
Inside, open “platinum” and then “data”, and there are several folders for interiors. Four of them have acronym suffixes for the four main games: “interiors_mbg” for *Gold*, “interiors_mbp” for *Platinum*, “interiors_mbu” for *Ultra*, and “interiors_pq” for *PlatinumQuest*. If you followed the guide exactly, open “interiors_mbg”. From there, you should find a folder called “custom”, which is where your .dif file goes (if the folder does not exist, create one yourself).



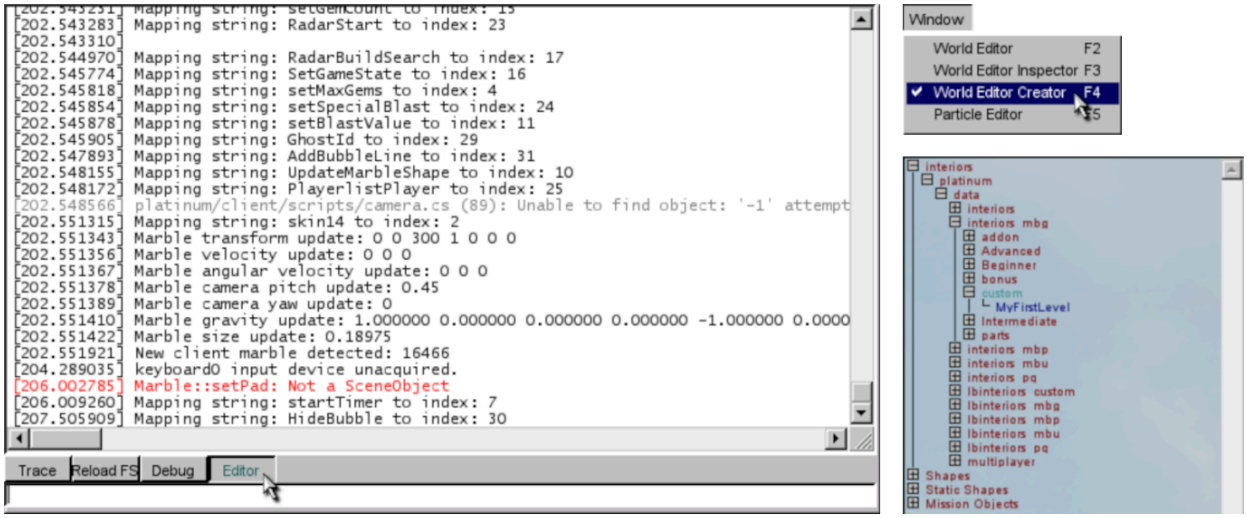
Finally, to load your interior! Open *PlatinumQuest*, and hit Play; within the "Custom" category, there is a subfolder inside "platinum/data/missions" called "templates". Select the template mission according to the interior you made (again, *Gold* if you followed the guide exactly).



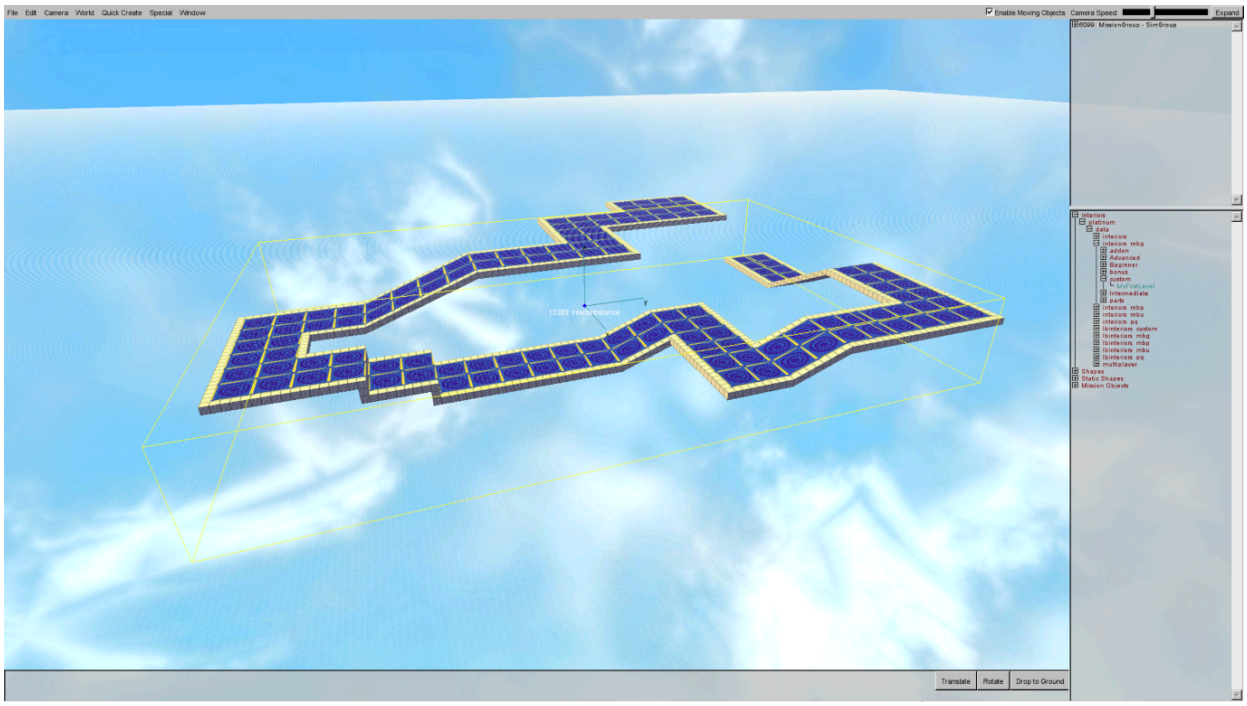




Open the console with ~ (tilde), and press the Editor button to activate the level editor. Hit ~ again to close the console, and open the editor with F11.



Open the Camera menu and select Toggle Camera to start flying around. Open the Window menu and select World Editor Creator. That will show the creator menu, where you'll search through the Interiors tree view to find your interior; click on it to load it into the mission.



In order to not overwrite the template mission, hit Save As in the File menu.

Everything from there is all about the level editor! This guide continues only within *Constructor*, so the rest in-game is up to you.

All of the fundamentals of building a level with *Constructor* have been covered by now, so everything afterwards is for specific concepts that are more complicated. For the meantime, feel free to become more familiar with the basics, and challenge yourself with what you already know. Move on when you're ready!

📝 Summary

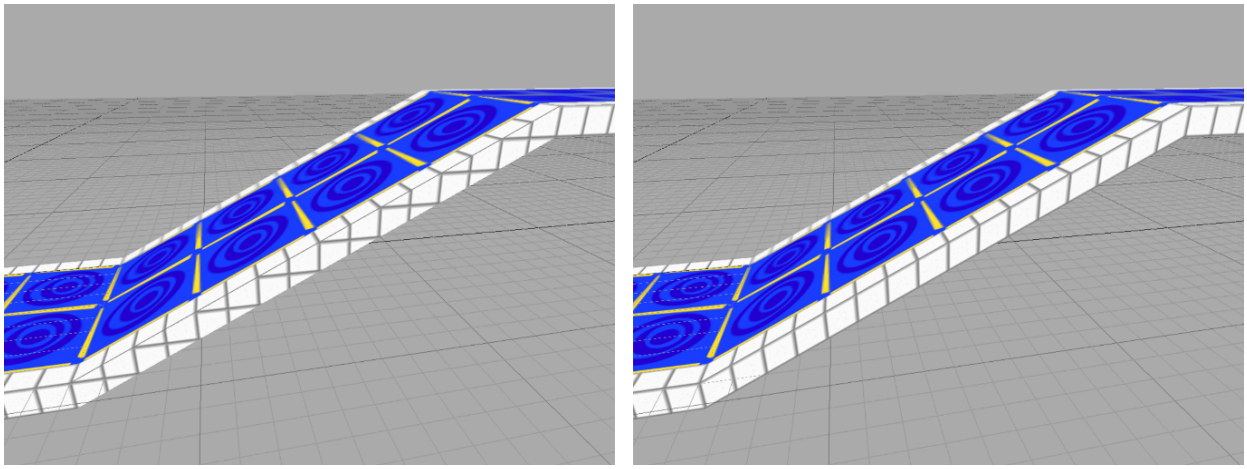
- There are two main file types, **`.csx`** and **`.map`**, but `.csx` is used far more for its wider range of capabilities.
- Malformed brushes (brushes with a red line through the middle) will convert using *csx3dif*, but they will disappear and not convert from `.map` files.
- To convert (export) a project, click and drag the file onto `csx3dif.exe`, which will create a **`.dif`** file that can be used in-game; there is also a website alternative.
- Copy the new .dif file to the folder labelled "custom" inside the interiors folder according to which game you created your level for.
- Create a new level using one of the supplied Template missions.

🖌️ IV: Ramp Trim

Part IV: Ramp Trim

This part of the guide will cover skewing trim to align it with sloped platforms.

Of course, this is not necessary to do on every ramp, but it certainly looks more polished than not doing so. Take a look at this side-by-side of the exact same ramp:

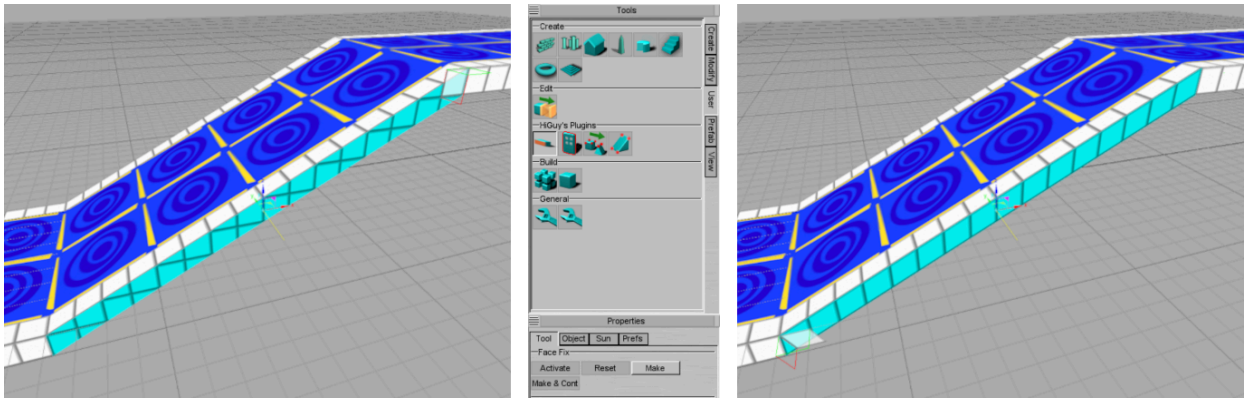


There are two methods for this. The first one, using HiGuy's *Face Fix* plugin, essentially does the job automatically. The other method is doing it manually, using the Advanced Texture options: this is great knowledge to have for situations where *Face Fix* either isn't enough, or simply makes a mistake.

🪄 ***Face Fix***

Constructor has a variety of plugins available for use, which are found in the **User** tab in the Tools window.

Select the face you want to skew, and press the ***Face Fix*** button under "HiGuy's Plugins", which has an icon of a slanted version of the Build Cube. Simply press Make, and... that's everything! It's that easy.

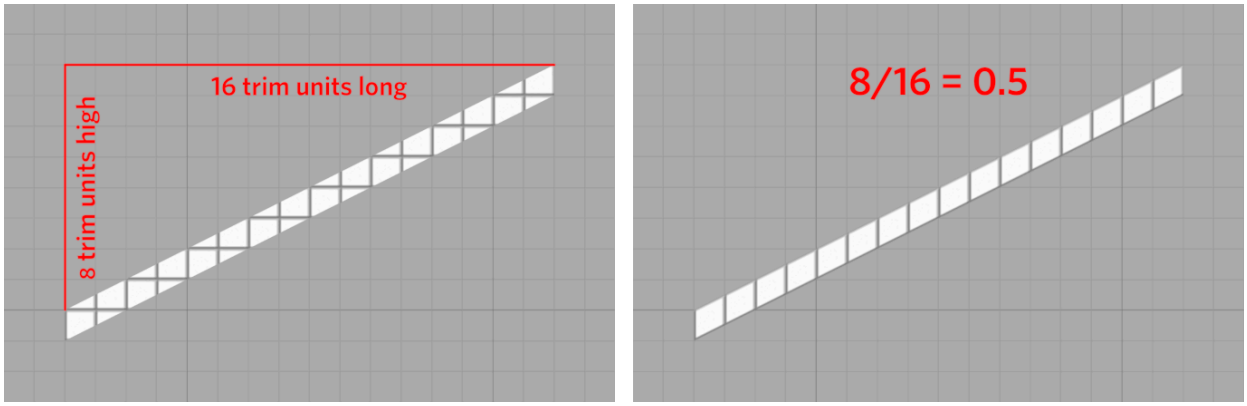


This tool can be used on multiple faces at a time; it is not necessary to go one-by-one.

🔧 Advanced Texture Options

This method is significantly more complicated than the simple press of a button, but easy to understand after only a couple practices and still plenty efficient.

First, focus on the 2D view that gives you a head-on view of the face you want to skew. Count the grid lines (with 0.5 grid spacing) and divide the horizontal trim units by the vertical trim units. This ramp has a slope of 0.5.

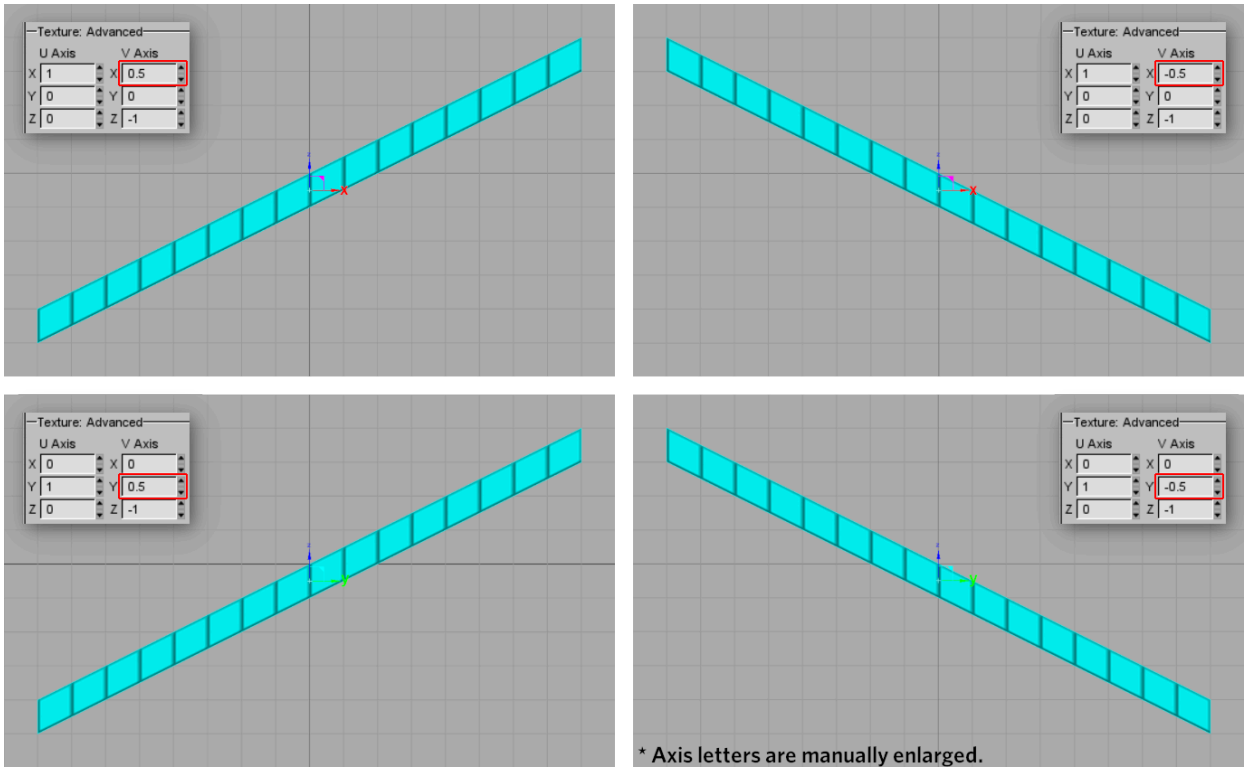


Select the face you're looking at and go to the Advanced Texture options at the bottom. The only two fields that are actually important for skewing the sides of ramps are the first two under V Axis, the X and Y values.

Fully understanding these options can take some trial and error, but here's the basic gist using the slope I have for my ramp:

1. If the ramp goes up in the same direction that the X arrow is pointing, type the slope as a positive number into the X field.
2. If the ramp goes up in the opposite direction that the X arrow is pointing, type the slope as a negative number into the X field.
3. If the ramp goes up in the same direction that the Y arrow is pointing, type the slope as a positive number into the Y field.
4. If the ramp goes up in the opposite direction that the Y arrow is pointing, type the slope as a negative number into the Y field.

(Match the field to the parallel axis; if the ramp goes up towards the arrow, type a positive number.)



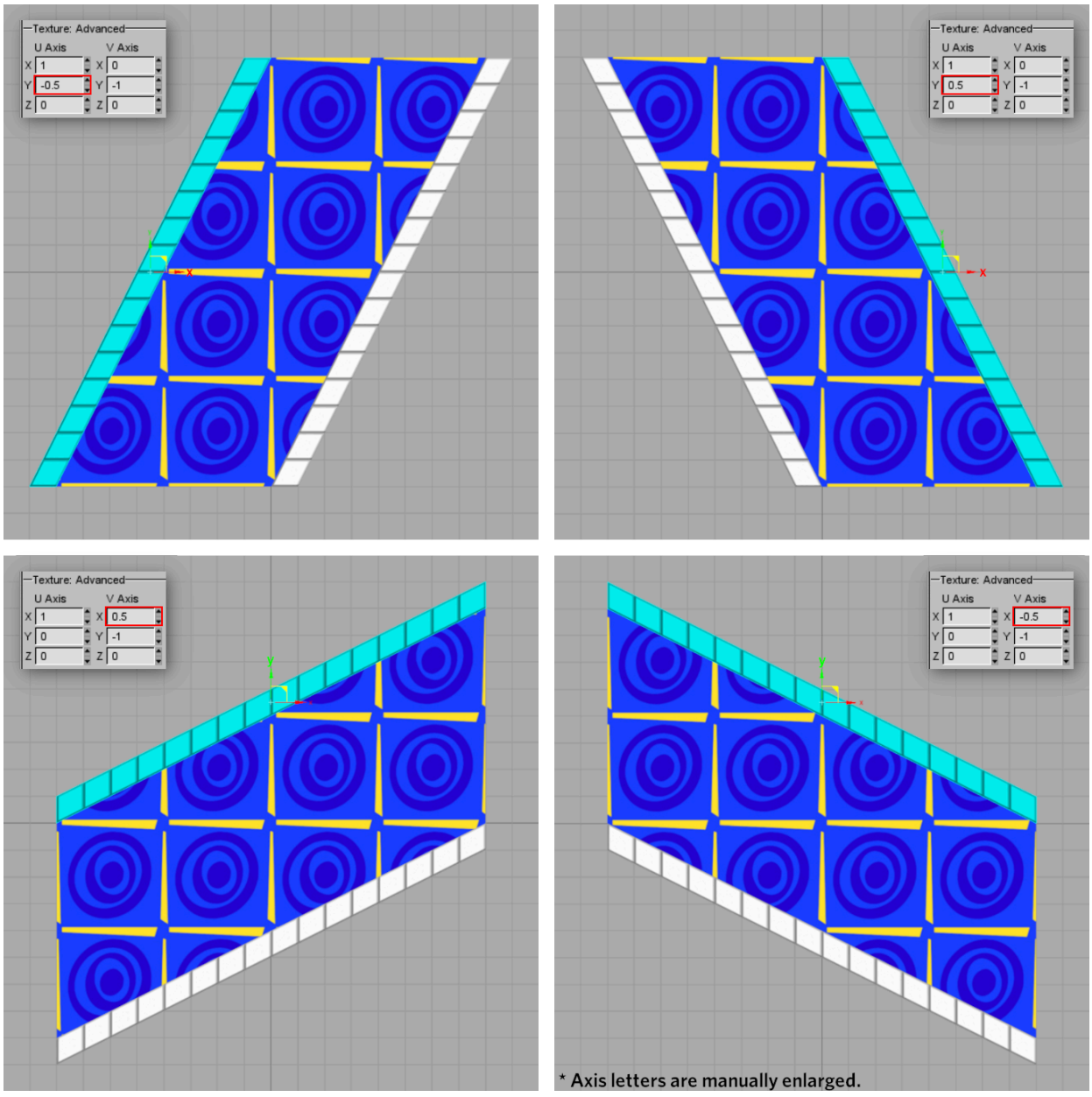
(The order of the images arranged the list above go left to right, top row to bottom row.)

Repeat the process on the other side of the brush, typing in the same slope value in the same field. If you want to speed this process up a little bit, you can select both sides and texture them at the same time.

With a texture facing straight up, and looking at the top-down 2D view:

5. If the ramp goes from southwest to northeast and the short edges are parallel to the X axis, type the slope as a negative number in the Y field in the U axis column.
6. If the ramp goes from southeast to northwest and the short edges are parallel to the X axis, type the slope as a positive number in the Y field in the U axis column.

7. If the ramp goes from southwest to northeast and the short edges are parallel to the Y axis, type the slope as a positive number in the X field in the V axis column.
8. If the ramp goes from southeast to northwest and the short edges are parallel to the Y axis, type the slope as a negative number in the X field in the V axis column.



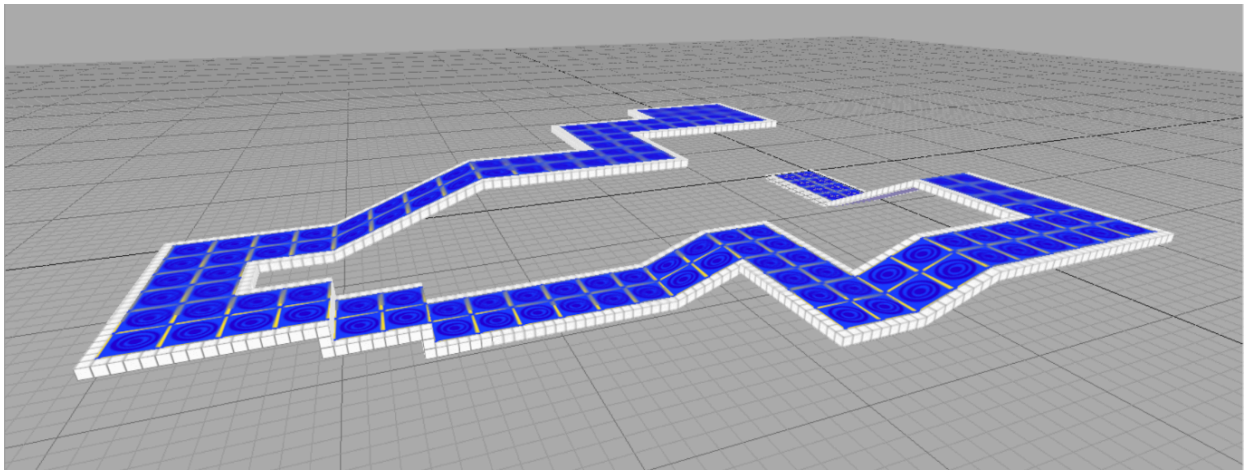
(The order of the images arranged the list above go left to right, top row to bottom row.)

With a texture facing straight down, follow the same instructions for the values that go in the U Axis column, and do the inverse for the values that go in the V Axis.
The reason for this is the value in the Z field of the V Axis column, where it says 1 for top faces and -1 for bottom faces.

Truthfully, it's not necessary to remember all of that information—trial and error will suffice, and eventually it will make more sense naturally.

Lastly, sometimes you might have to fix the alignment, which can be done just like how it's done normally with the Justify arrows.

🏰 Build Something Beautiful!



At this point, you've learned all the fundamental skills to build a basic level and clean it up.

Obviously, there is plenty more to learn—circular platforms, moving platforms, tips for detailing, tips for building more efficiently...—you can still do so much with what has already been taught to you, so practice and become more comfortable with the basics! Feel free to play and around and experiment; here are some ideas to consider:

- Towers for scenery (connected to the platform, or surrounding the level)
- Gaps in the floor
- Blocks or small structures to weave around for obstacles.

You have many, many possibilities at your fingertips already. Best of luck!

📝 Summary

- *Face Fix* is a very efficient tool.
- If done manually, the correct value can be found using slope formula (rise/run).
- The axis parallel to the face being adjusted, and the direction the respective arrow is pointing, determines which field to enter a value into and whether it will be positive or negative.

🥧 V: Pie Slices

Part V: Pie Slices

By now, you should have a handle on the fundamentals of *Constructor*: flat platforms, ramps, and basic texturing. There is still a lot you can learn!

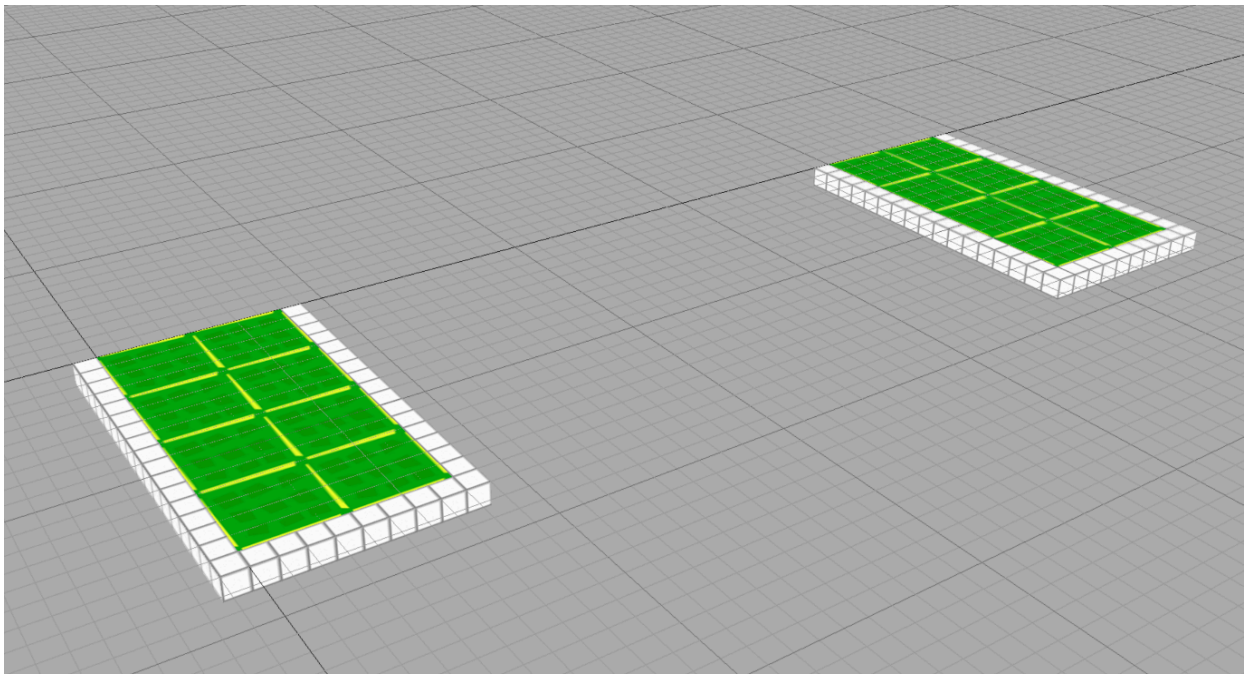
Going forward, there will be less explanation of the fundamentals within new concepts, so it is encouraged to track back to earlier parts of the guide if you get stuck.

This part of the guide will cover *Pie Slices*, the tool used to make circular shapes.

🏭 Building a Pie

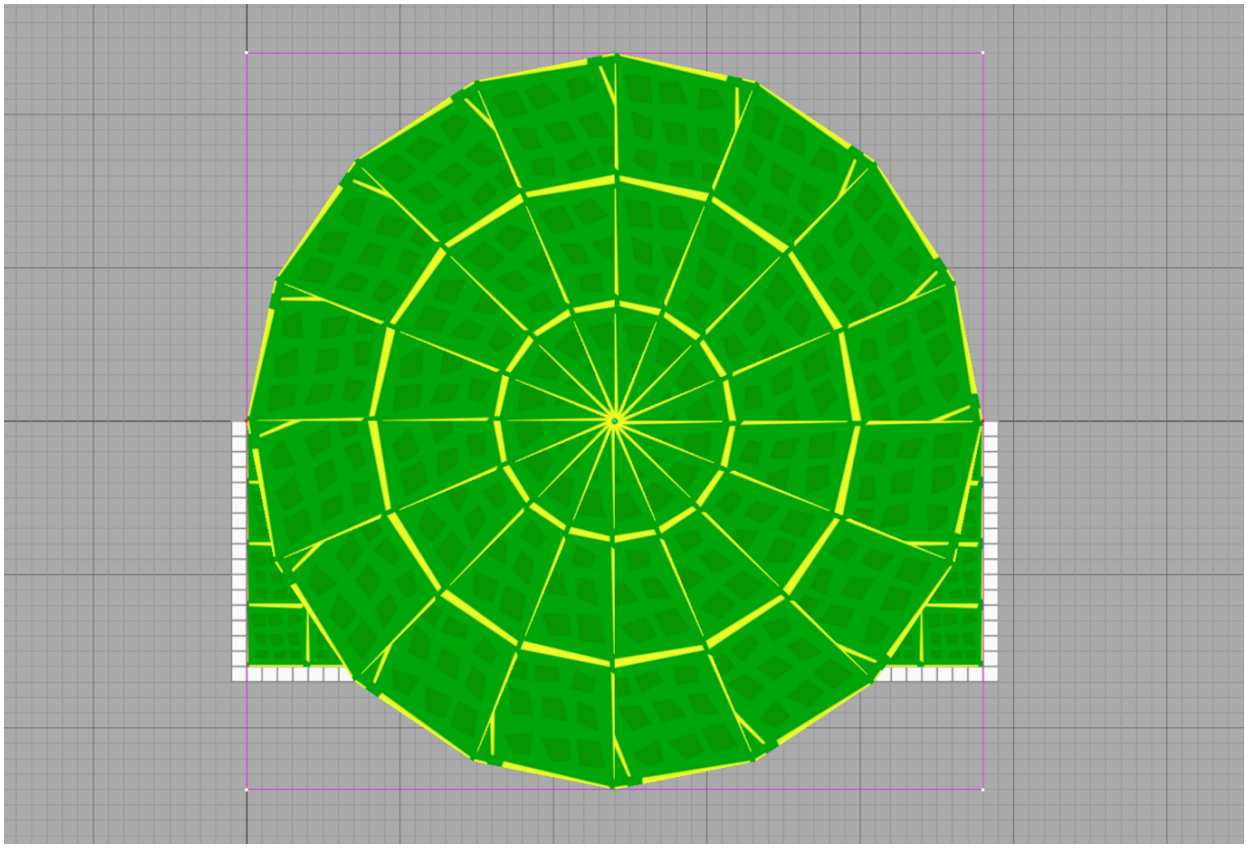
Here are two platforms that will be connected with a circular platform. The open ends (the ones without trim) are facing the same way, so this needs a 180° turn—half of a full circle.

If you would like to follow along exactly, you can find a copy of this project here. The gap between the two platforms is 7½ floor tiles wide, or eight tiles if the trim is ignored.



Texture: grid_neutral

After choosing a desired texture, start with selecting the **Create Pie** tool, which is in the User window of the Tools window. Use the top-down 2D view for this—your cursor should have turned into a crosshair, so click and drag anywhere to create a pie (circular platform). The dots on the sides (red and green) and corners (white) of the borders can be helpful in resizing the pie to fit with the two platforms. Make sure the pie is aligned with the floor tiles of the platforms, not the trim.

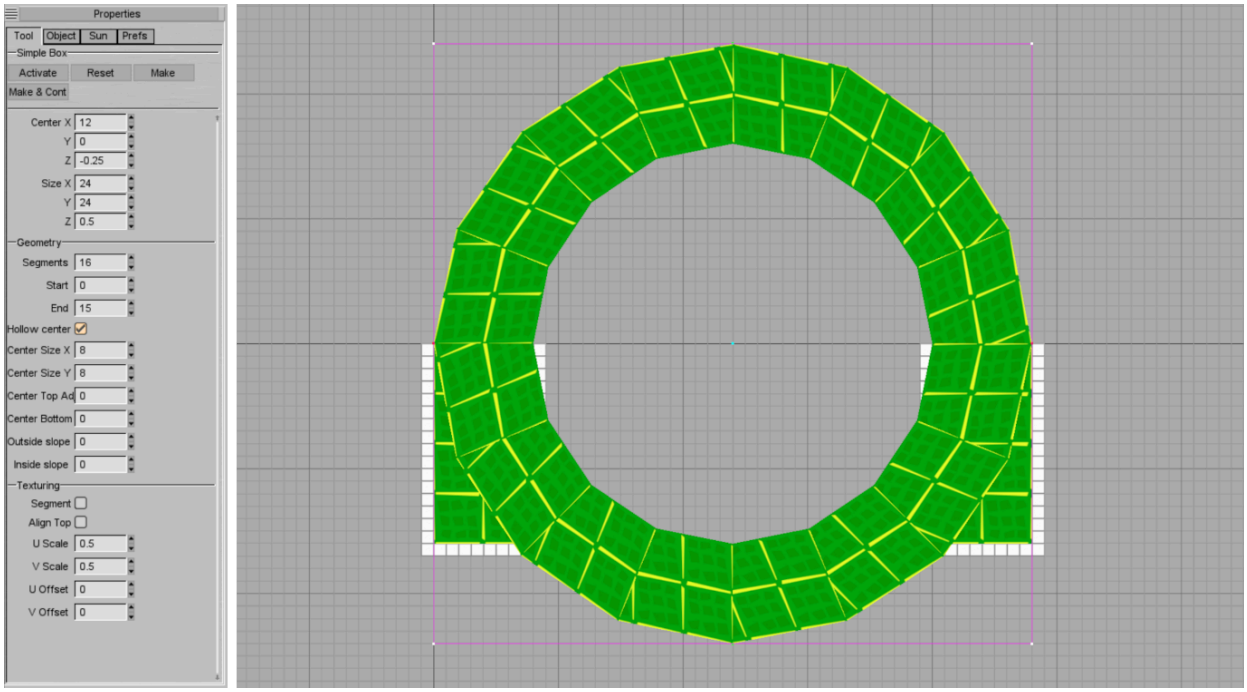


Remember the vertical sizing too! Use the blue dots in one of the side-view 2D perspectives to adjust the height and depth, or look at the Z value in the Size field in the Properties window and change it to 0.5.

It looks weird in the beginning, but at least it fits. Completing the desired shape must happen before fixing the texturing.

In the Properties window, check the Hollow Center box. Change the X and Y values to align with the inner sides of the two platforms being connected—one floor tile is 2 units. Be sure the size of the hole does not exceed the radius of the pie, or the entire thing will disappear.

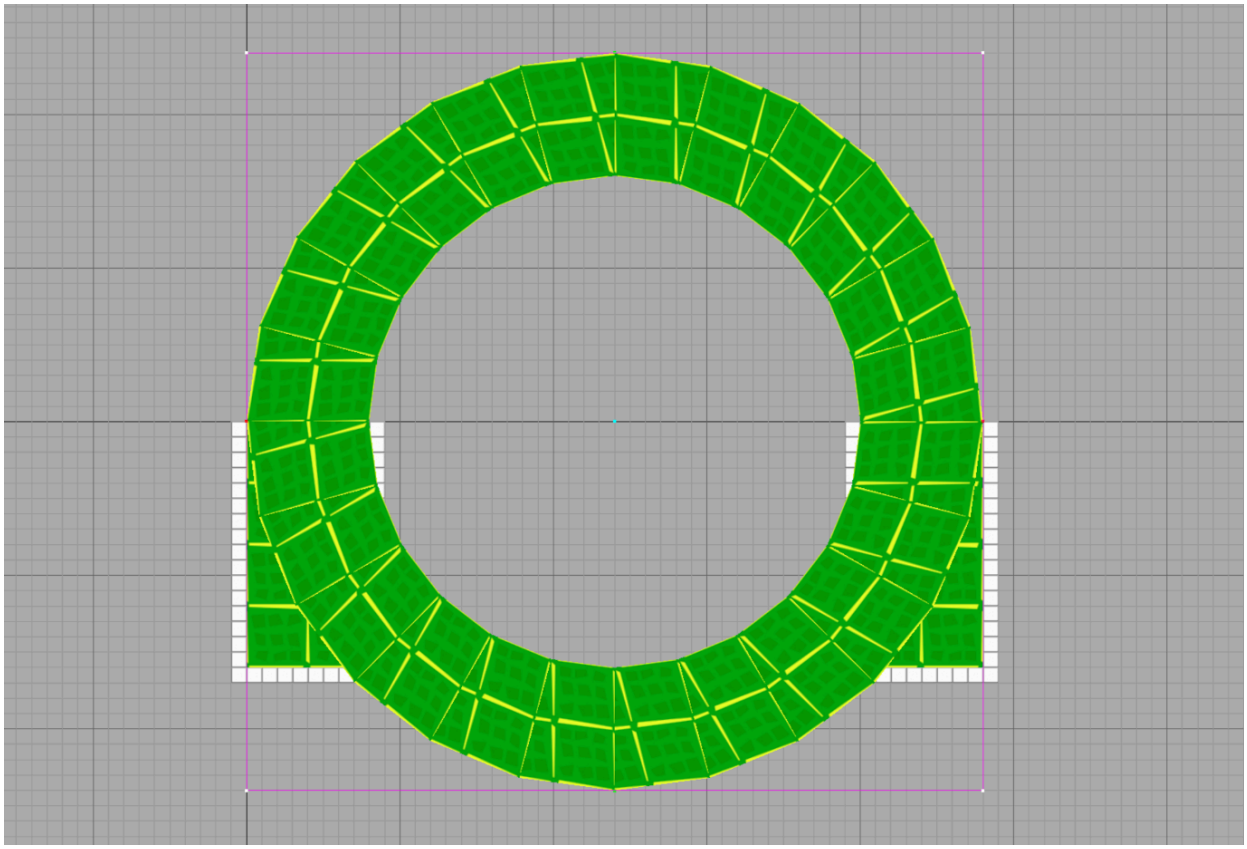
Change the U and V Scale options at the bottom to 0.5, like floor tiles are normally scaled.



Lastly, before finalizing the build of the pie, is the number of **segments** it's made of, which is also in the Properties window. Some philosophies that are best to follow (although not necessary):

- Try to keep the number of segments as a multiple of 4, so the whole pie can easily be split into halves or quarters (for a U-turn or a quarter turn—just a good practice).
- Furthermore, pay attention to the inside edge of the pie, so the inside width of each segment is roughly as wide as a floor tile.
    - The benefit is for the trim that will run along the inside edge, in which each segment will be roughly four trim units long.

In the photo below, the pie is made up of 24 segments.

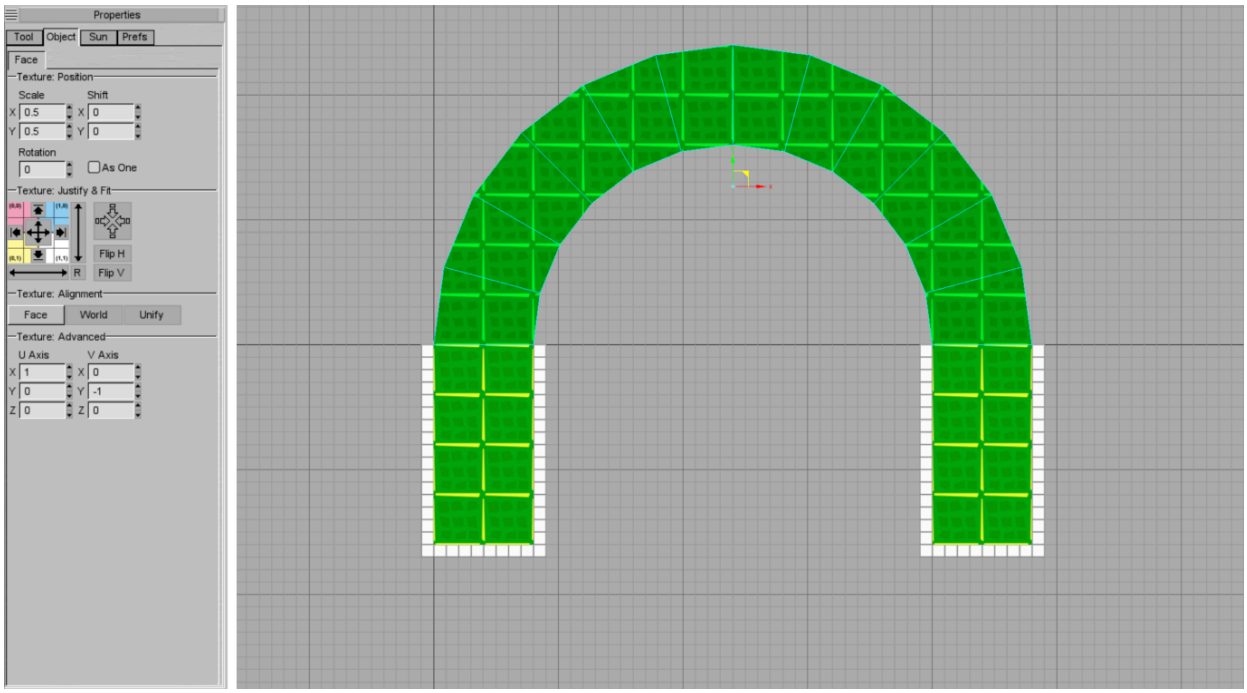


🎨 Texturing a Pie: Flat Sides

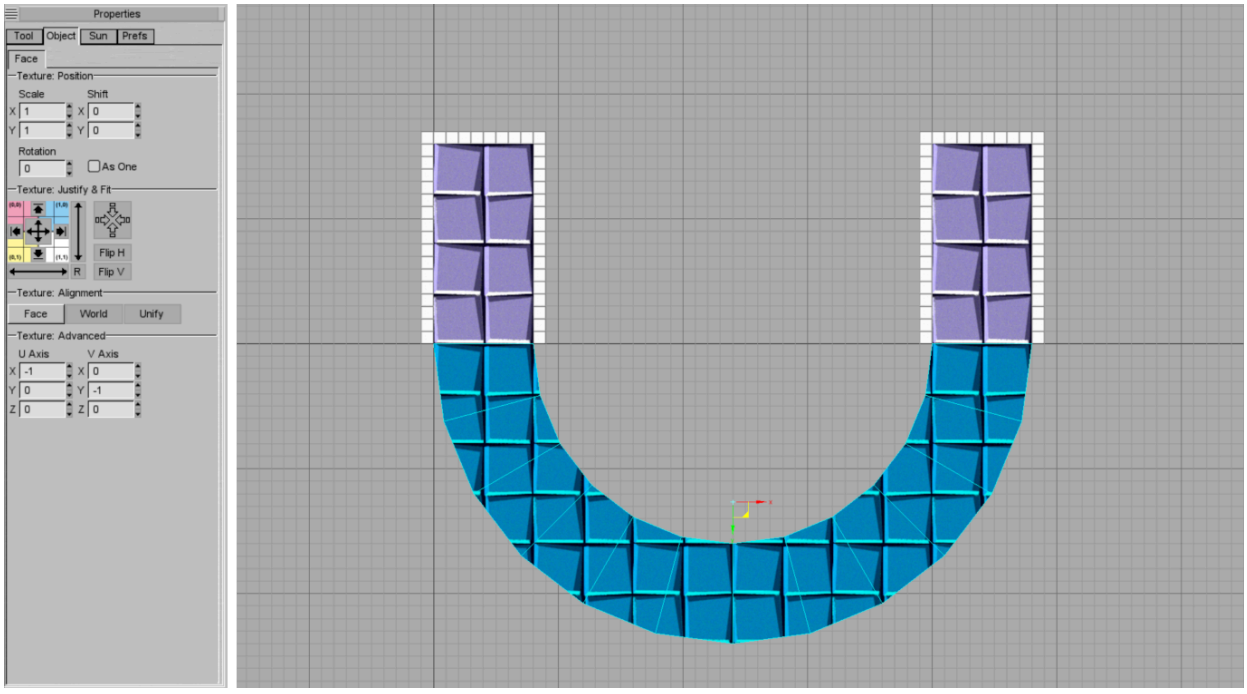
Finalize the build by clicking Make or pressing Enter. Now, pies made using *Pie Slices* are all made up of individual brushes, so the ones that do not need to be there (the entire bottom half) can be deleted.

In these situations, texturing is simple, since every top face will be set to the same shift, including the two end platforms. The photo below shows every top face having their shift be set to 0 and 0, like explained in Part II. You may have to press the **Face** button first (in the Properties window; different from the Faces mode button).

This also confirms that the two platforms being connected are also aligned properly, and the resulting curve will not leave a partial tile.



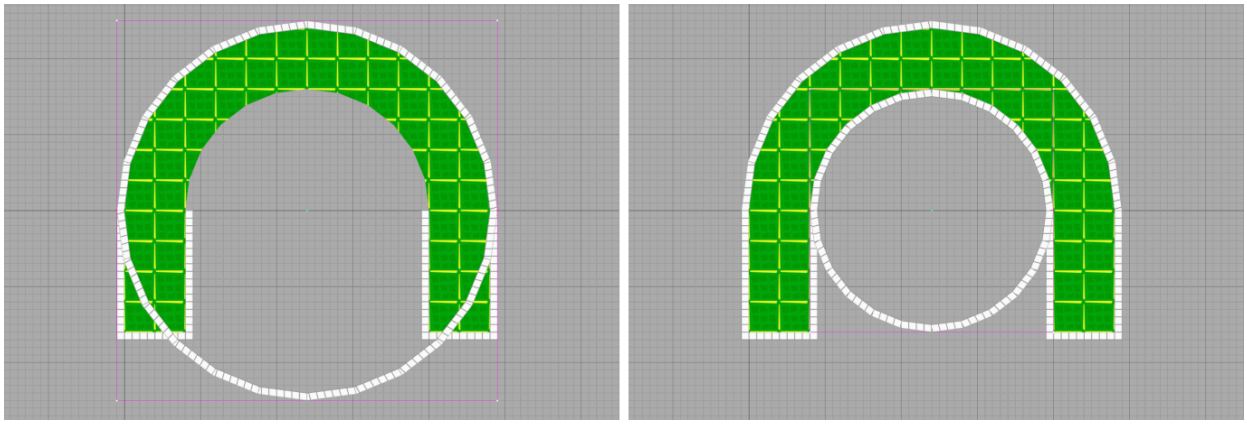
The same can be done on the underside as well.



⛏ Trimming Pie Slices

After the main path comes the surrounding trim, starting with the outside edge. The first several steps are the same: draw and align the pie, change the hollow center to fit around the path, change the U and V scale to 1 so the trim texture is the correct size, finalize the build, and delete the unnecessary segments. Make sure the number of segments stays the same.

Do the whole process one more time for the inside edge.



The quick formula to find the hollow center of a trim pie:

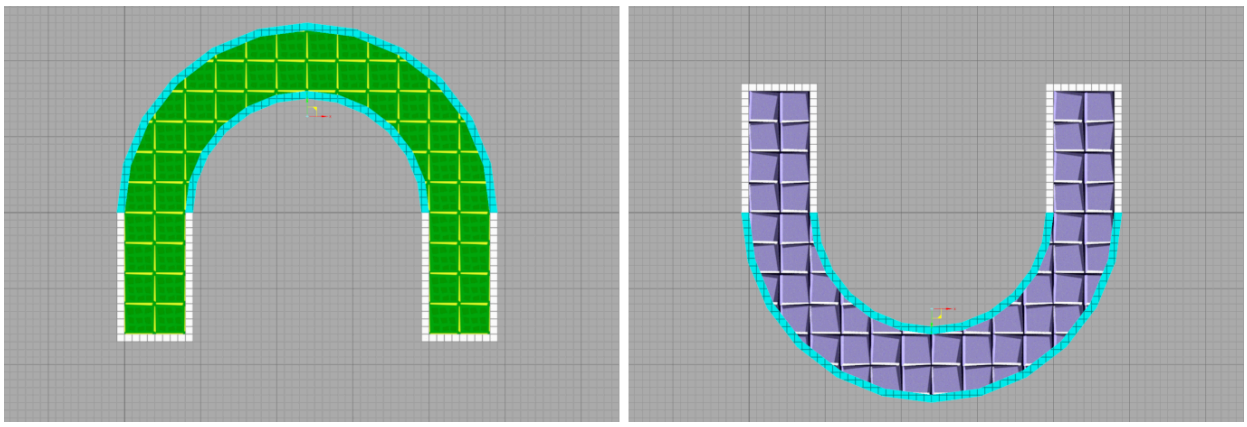
- Start with the size of the pie.
- Divide by two (a circle's radius is half its diameter).
- Subtract 0.5 (one trim piece is equal to 0.5 *Constructor* units).

Here is that formula being used for both sets of trim, the outside on the left, and the inside on the right.

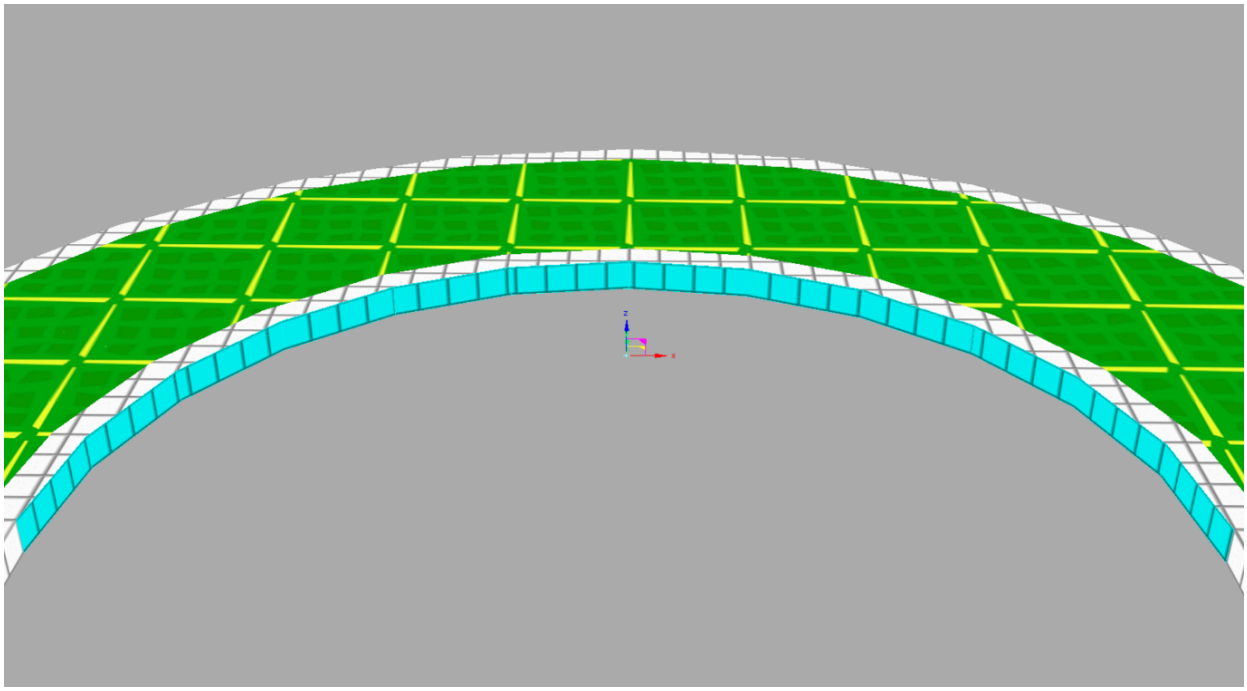
🎨 Texturing a Pie: Curve Sides

Texturing the sides of pies isn't quite as simple. Given the non-straightforward nature of circles, each side face will have a decimal value in both its scale and shift. The goal is to find that exact scale and align it correctly.

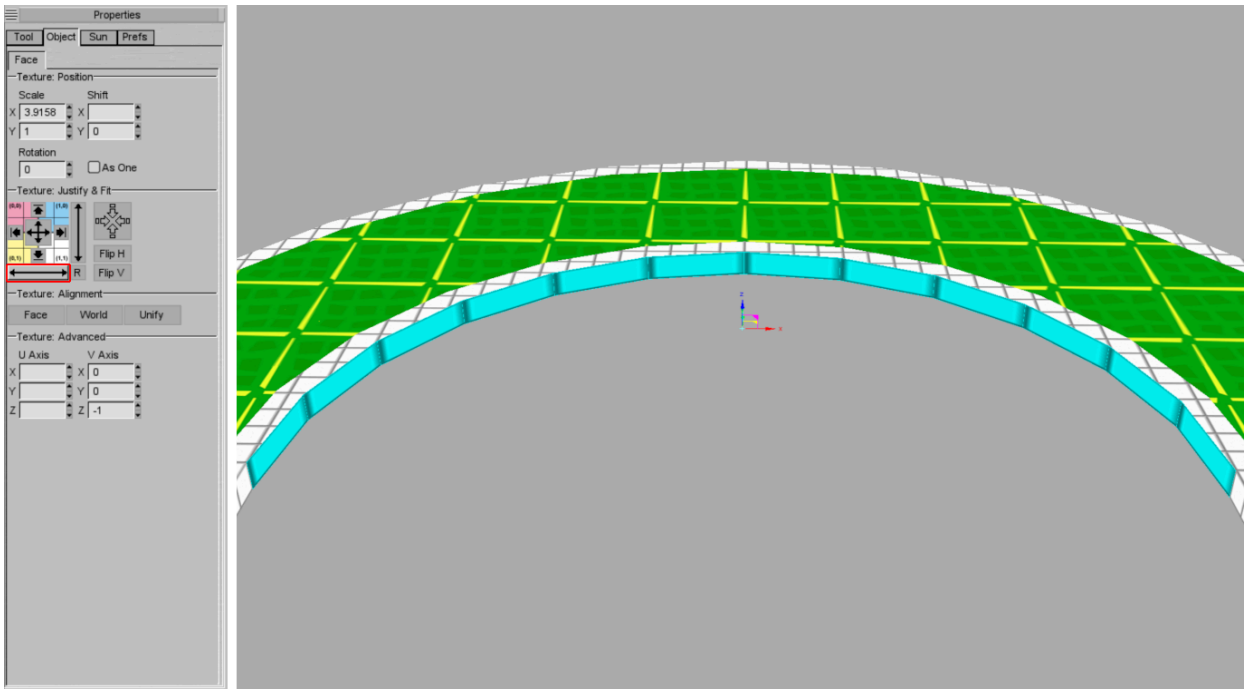
First, the top and bottom of the trim can receive the same treatment as the floor tiles: set the shift to 0 and 0. (If you need to, hit the Face button first.)



Now to texture the inside and outside trim faces of the curve. These have to be done separately, because the sizes will differentiate. Start with the inside faces by selecting all of them and hitting the Face button to reset their orientation to match the sides of the trim on the other platforms.

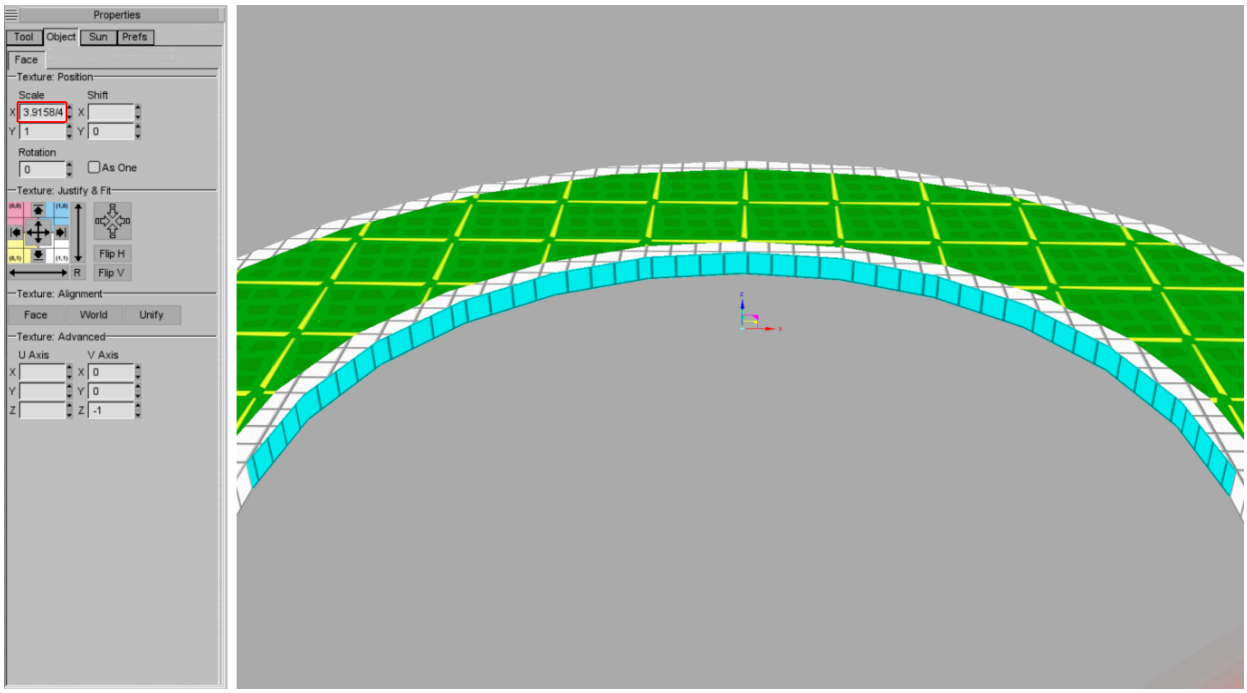


The critical piece to having the trim faces align correctly is the **Stretch to Fit Width** button: this will set the X scale exactly to the width of the face. (There is indeed another button for length, but that isn't important now.)

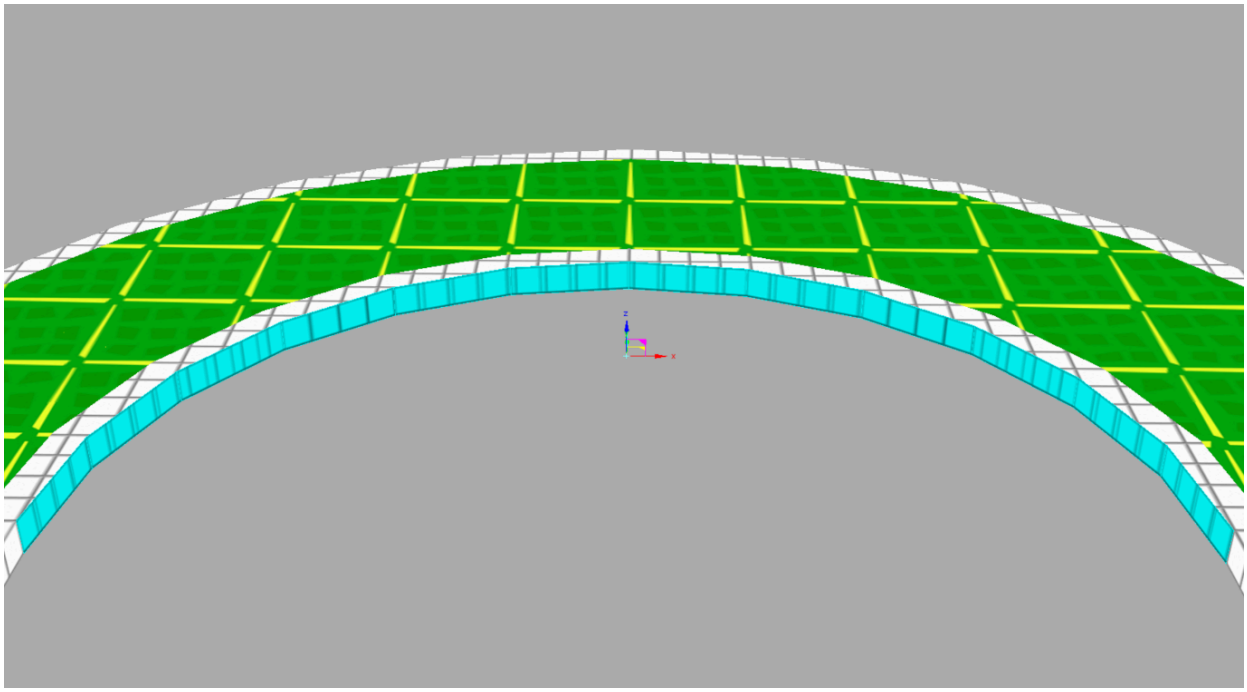


Now it should be more clear why it's suggested to choose the size and number of segments in the pie so the width of the inside faces are roughly equal to a floor tile, which is the same as four trim tiles. The X scale of the selected faces are close to 4 (or, simply, a whole number), so the solution to fixing it is to divide the value by 4 to bring the size as close as possible to 1. There will be four evenly-sized trim tiles that perfectly fit the width of the brush.

You can write in these fields like you're using a calculator! Typing "/4" will suffice.

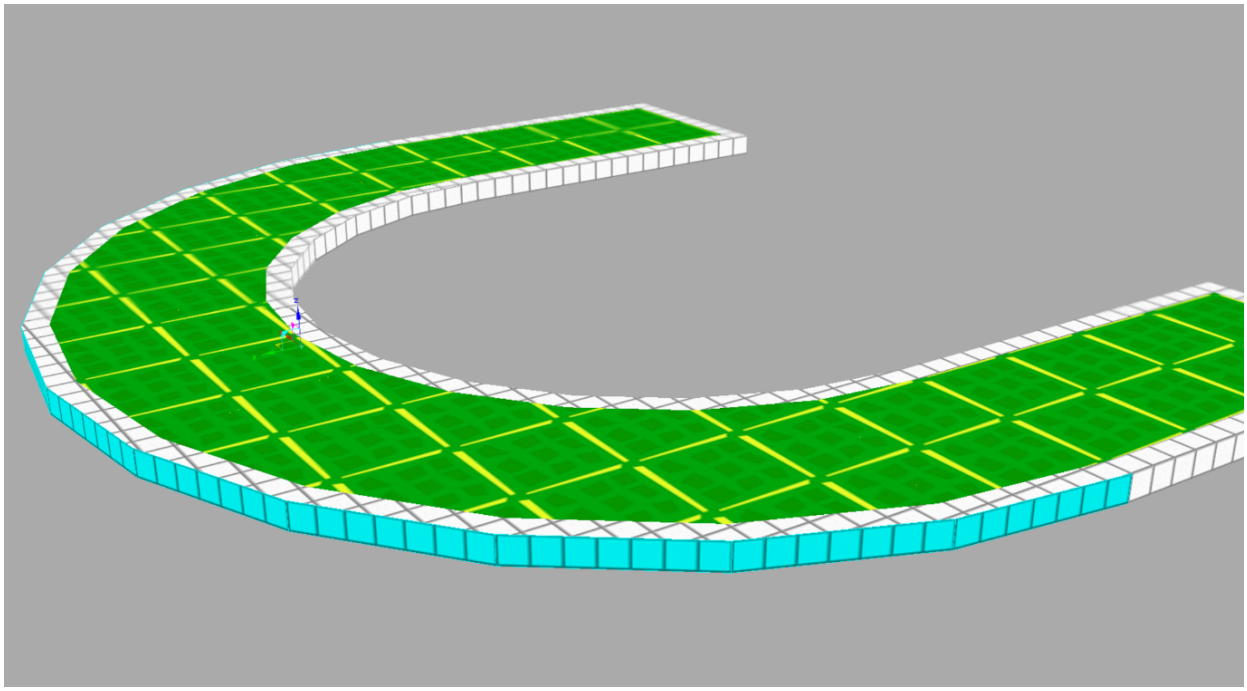


This is a case where setting the X shift to 0 will not result in the desired look. so use the Justify Right button instead.



Bravo! That should look nice. It was a lot, but fortunately there's a chance to practice again with the outside edge. The process is the same:

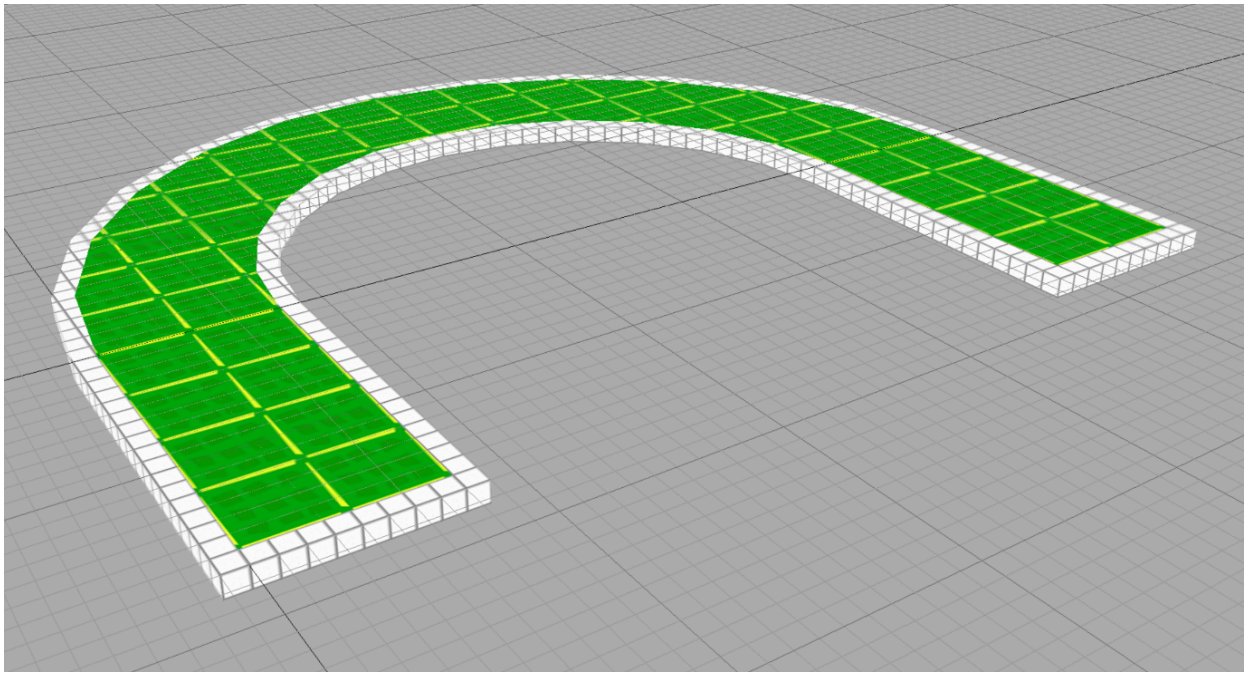
1. Select all the faces and hit the Face button.
2. Resize the faces using the Stretch to Fit Width button.
3. Divide the stretched size by the nearest whole number so the new size is as close as possible to 1.
4. Fix the alignment with the Justify Right button.



Something to keep in mind: sometimes the decimal value after using the Stretch to Fit Width button won't be as close to a whole number as this example with the inside faces, and if you followed this guide exactly, you will find that this is the case with the outside faces. The stretched size is about 6.5, which is annoyingly halfway between 6 and 7—at this point, it doesn't make a huge difference which you divide this by.

In the photo above, each face consists of 7 trim units.

💭 More to Come



That's a sweet basic curve!

There is a wide open world when building with curves, and future parts of the guide will explain better trim texturing, being more efficient in building curves, and creating loops. All three are closely related, and some aspects will actually make building curves a cleaner and faster process, but it's important to have a fundamental understanding of building curves in the first place.

Next: Moving Platforms

📝 Summary

- Draw a pie with the Create Pie tool in the top-down 2D view, aligning the borders with the edges of the platforms being connected.
    - Adjust the hollow center to fit the width of the platforms.
    - The number of segments should be a multiple of 4, and is determined by the size of the pie and how close the inside faces can be to the length of a floor tile.
- The Face button resets the orientation of the selected faces, so the top and bottom faces can be aligned correctly
- The inside faces (of the trim) are textured by:

    1. Selecting all of them and using the Stretch to Fit Width button to resize them to the exact width of the face.
    2. Dividing the stretched size by the nearest whole number so the new size is close to 1.
    3. Fixing the alignment with the Justify Right button.
- The same process applies for the outside faces. If the stretched size isn't close to a whole number, round up or down.

[WIP] VI: Moving Platforms

WORK IN PROGRESS

Part VI: Moving Platforms

This part of the guide will cover moving platforms, the last major piece of building levels. Continuous *and* triggered moving platforms will both be covered.

🏗️ Before Starting

So, moving platforms behave a little tediously in Constructor—there is a certain order to the steps, and making a mistake can result in quite a bit of inconvenience, so here's a little housekeeping before starting:

- Set up your moving platforms as static brushes. (If you look in the Properties window...)
- If you will have a triggered moving platform, make a static brush that will later become a trigger.
    - This typically includes two triggers: one to activate the moving platform, and another to call it back.
-

Before Making Moving Platforms:
First thing to always do is set up your moving platforms as normal static brushes, as well as make static brushes for where you want your triggers to be. Obviously important to make a trigger to activate the platform, but you can make a trigger to call it back too. Save a copy of your .map with everything set up, and then make another copy that will actually have the moving platforms.

Constant Moving Platforms:
- Select moving platform, hit "Other" on the left and then "Door_Elevator".
- Spawn a path node marker by clicking the Add Point Entity button (lightbulb with a plus sign), and scrolling down to find "path_node". Place it in the middle of your starting platform.
    - Never a bad idea to set the duration (in milliseconds) and the smoothing (almost always 2) so those properties are copied.
- Duplicate the path node by shift-click and dragging, place it where you want your platform to go.

- Make a duplicate of the path node that's at the end of the path.

Triggered Moving Platforms:
- Follow every step from making a constant moving platform.
- After you finish placing the path nodes, select the brush that's supposed to be a trigger that you intend to activate the platform, and hit "Trigger" on the left (a couple options above "Other".
    - If you made a second trigger to call the platform back, make that a Trigger brush type too.
- Up in that same box is an "Edit Properties" button:
    - Select the DoorElevator that is your triggered platform
        - "InitialTargetPosition" should be 0.
        - "datablock" should be "PathedDefault".
        - You can add those two properties yourself (there's an "Add Property" button if they're missing for whatever reason.
    - The "TargetTime" of the first trigger should be the same value as the duration of the first path node in the path.
    - The "TargetTime" of the second trigger (if you made one) should be 0.

In-Game:
- Open the mission, spawn your interior, hit F3, hit "Create Subs".
- Save and reload the level.

Notes:
- In cases where you have to remake the moving platforms from scratch, Constructor doesn't like it when you simply delete the moving platforms and make them again (it messes up the order of everything in the "Edit Properties" window. This is why I highly suggest saving a copy of your .map with all the moving platforms and/or triggers ready and set up.
- It actually doesn't matter where the path nodes are placed in relation to the moving platform they go to. The two things that do matter are that the moving platform is placed where you intend it to start, and where the path nodes are in relation to each other. You can move all of the path nodes at once to a completely different part of the level and the moving platform will still move in the exact same way in-game. (I usually have my path nodes on the corner of my moving platforms instead of the center.)
- Sometimes the moving platforms themselves don't spawn, but the markers and triggers do. I don't know what causes this. Sometimes I have to remake the moving platforms all over again, sometimes I even have to install a fresh new copy of Constructor. It sucks, I know.

- If you've already spawned your moving platforms in-game and you make a change to the path, you have to delete the lines of code in the .mis file for every part of the moving platform (the MustChange, the path, etc.) before you hit "Create Subs" again.
- Constructor is still a piece of shit, this is what I do every time I make moving platforms, but of course not everything will go 100% smoothly, so prepare to either be very patient and/or very sad.