Simulation RGB Workflows

This page provides a tutorial on the different workflows available for controlling the RGB Grid Channel of the Chaos Phoenix Simulator in 3ds Max.

Overview

This is an Entry Level tutorial which requires no previous knowledge of Phoenix. A basic understanding of 3ds Max would be helpful but is not a prerequisite for being able to follow along.

Requires Phoenix 3.11 Official Release and V-Ray Next Official Release for 3ds Max 2015 at least. You can download official Phoenix and V-Ray from https://download.chaos.com. If you notice a major difference between the results shown here and the behavior of your setup, please reach us using the Support Form.

The instructions on this page guide you through the different workflows available for controlling the RGB Grid Channel of the Phoenix Simulator in 3ds Max.

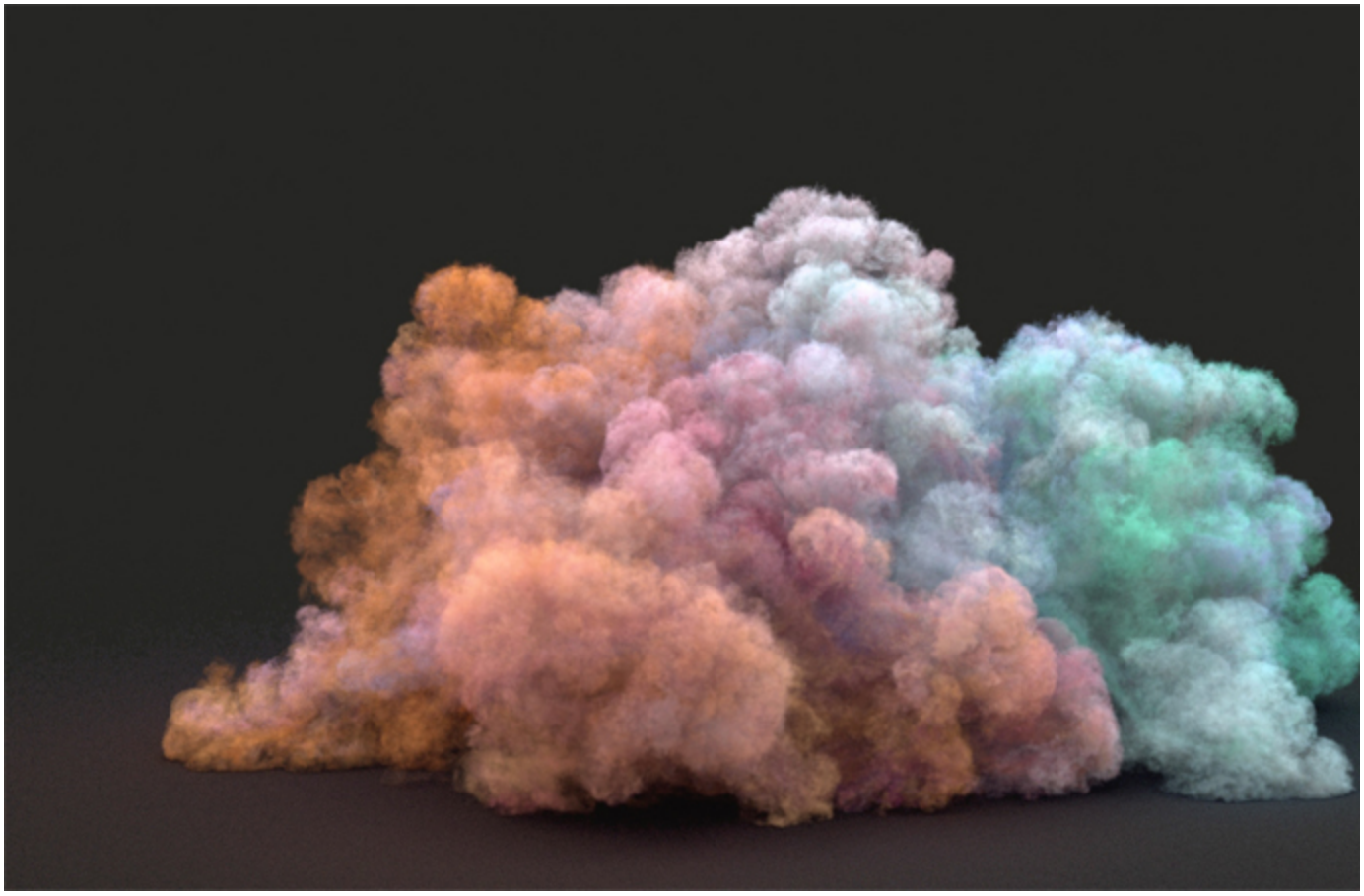
The tutorial is divided in chapters covering different topics such as:

1. Using a Phoenix Source to emit a uniform RGB color.
2. Using a Phoenix Source to emit RGB from a texture.
3. Using a Phoenix Source to emit RGB from the vertex colors of the source geometry.
4. Using a Phoenix Source in Brush mode to change the RGB color of Simulations while the simulation is running.
5. Using a Phoenix Mapper in conjunction with a V-Ray Distance Texture to change the RGB based on the proximity of an object in the scene while the simulation is running.

All these workflows are suitable for both **Fire/Smoke** and **Liquid** simulations.

The Download button below provides you with an archive containing example scenes based on the instructions on this page.

Download Project Files



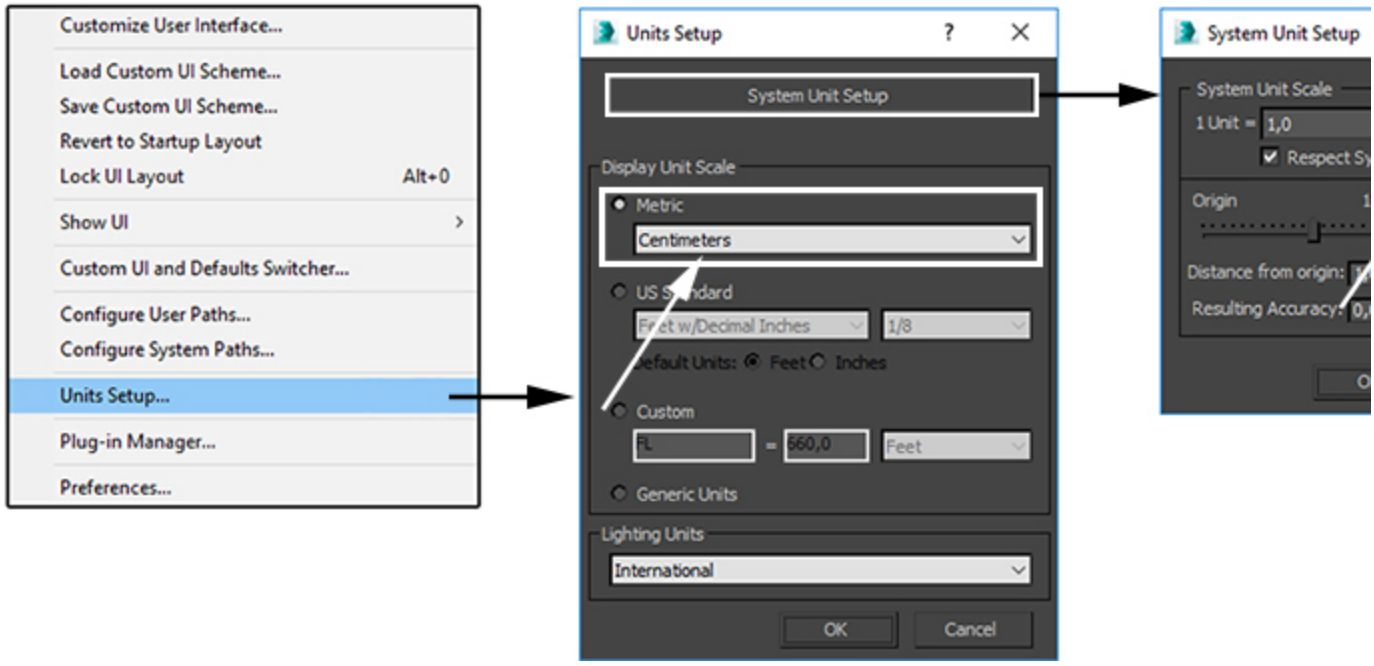
Units Setup

Scale is crucial for the behavior of any simulation. The real-world **size of the Simulator** in **units** is important for the simulation dynamics. Large-scale simulations appear to move more slowly, while mid-to-small scale simulations have lots of vigorous movement. When you create your Simulator, you must check the **Grid** rollout where the real-world extents of the Simulator are shown. If the size of the Simulator in the scene cannot be changed, you can cheat the solver into working as if the scale is larger or smaller by changing the **Sc ene Scale** option in the **Grid** rollout.

The Phoenix solver is not affected by how you choose to view the Display Unit Scale - it is just a matter of convenience.

Go to **Customize Units Setup** and set Display Unit Scale to **Metric Centimeters**.

Also, set the **System Units** such that **1 Unit** equals **1 Centimeter**.

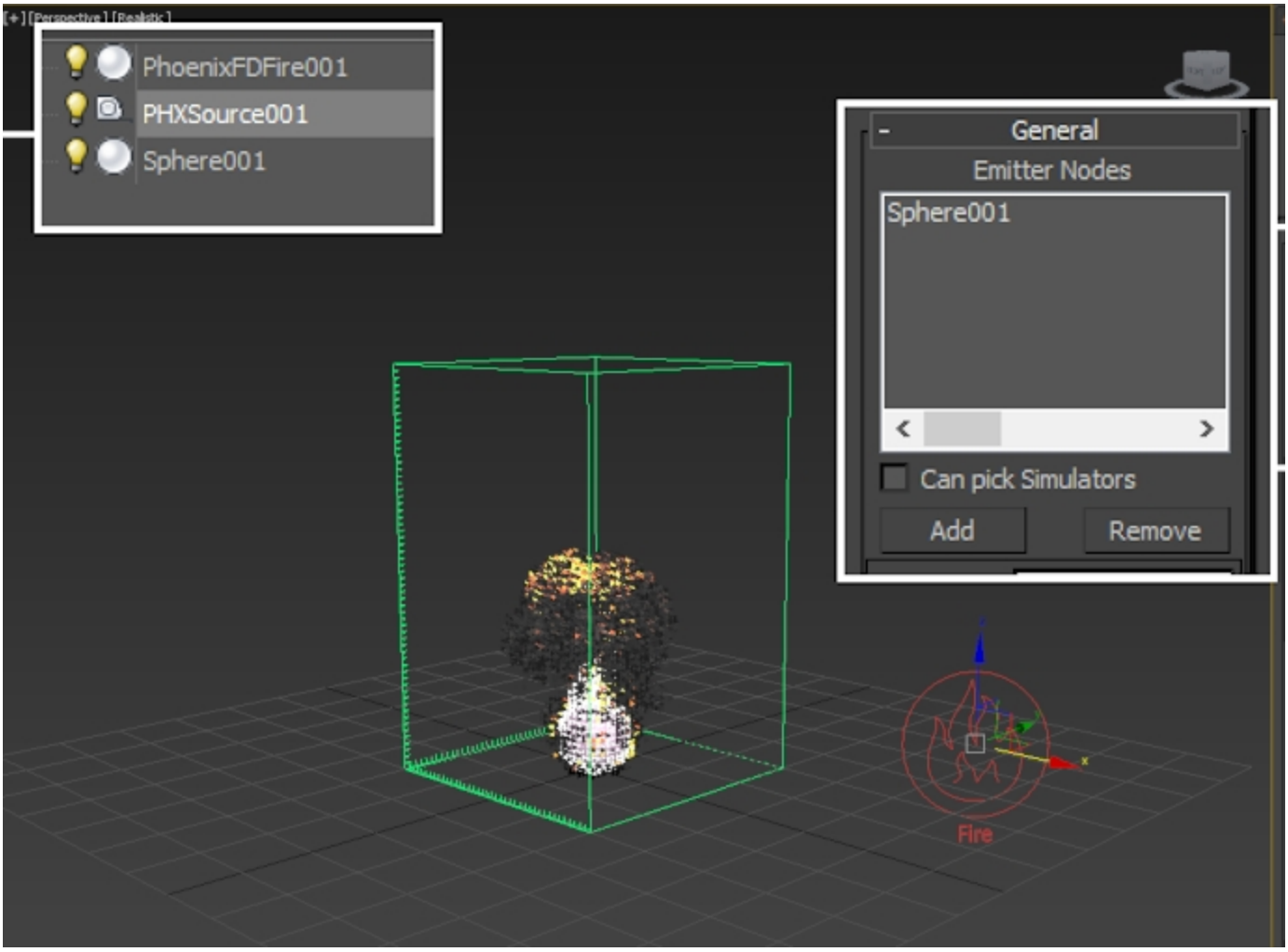


Chapter 1: Emitting uniform RGB color from the Phoenix Source

Create a:

1. **Standard Primitives Sphere** with a **Radius** of **5cm**. Move it **5 units up in the Z** direction so it sits at the origin.
2. **Phoenix Fire/Smoke Simulator** with a **Grid X/Y/Z** size of **40/40/60**.
3. **Phoenix Fire/Smoke Source**.

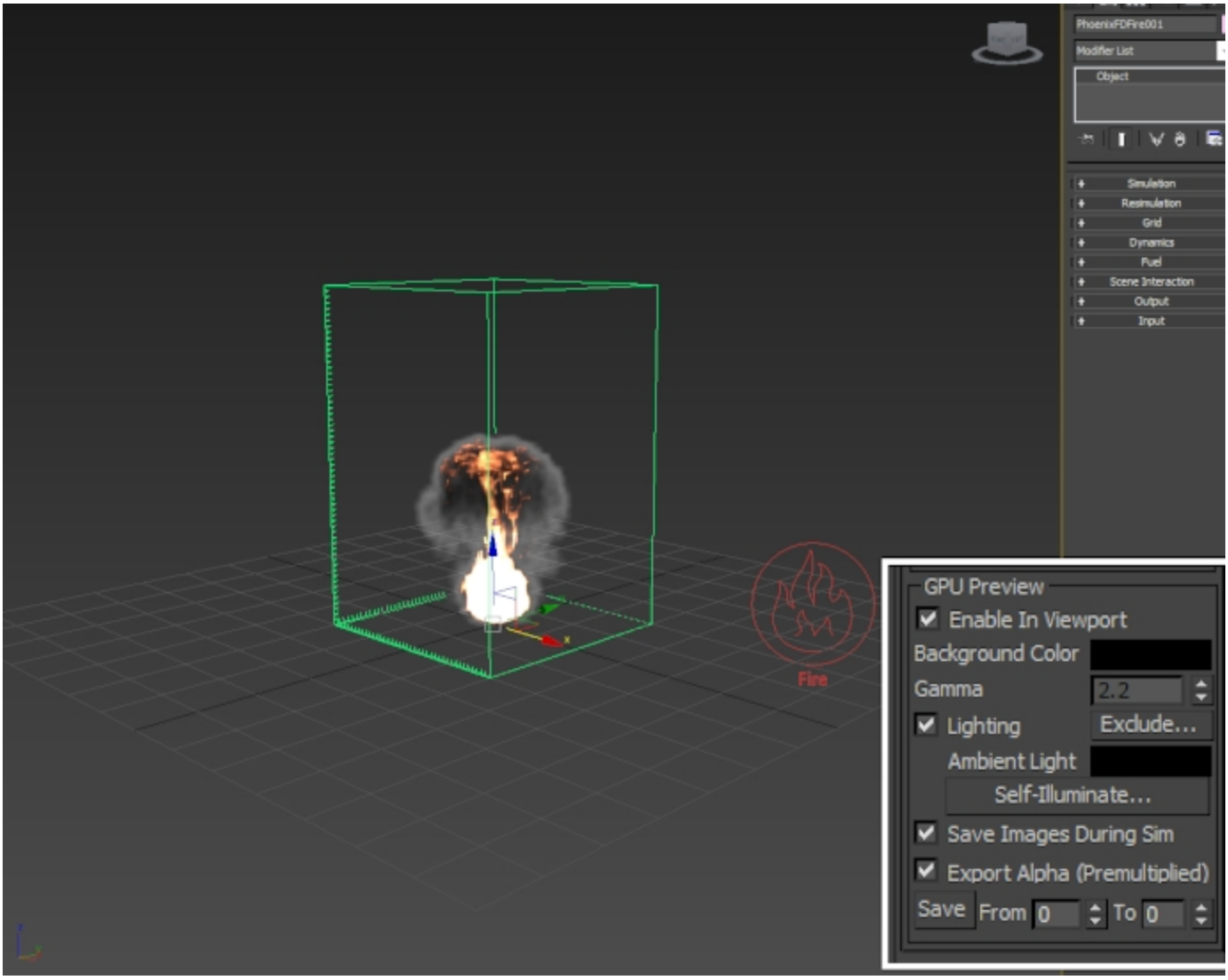
Select the Source and add the **Sphere** to the **Emitter Nodes** list so the **Source** knows to use it as an **emission geometry**.



Start the simulation from the **Phoenix Simulator Simulation** rollout.

Open the **Phoenix Simulator Preview** rollout and **Enable GPU Preview**. Note the presence of Fire in addition to the Smoke.

Since we are trying to focus on shading the **Smoke Color based on** the **RGB Grid Channel**, in the next step we will disable the export of the **Temperature Grid Channel** and also the rendering of Fire from the **Volumetric Shader**.

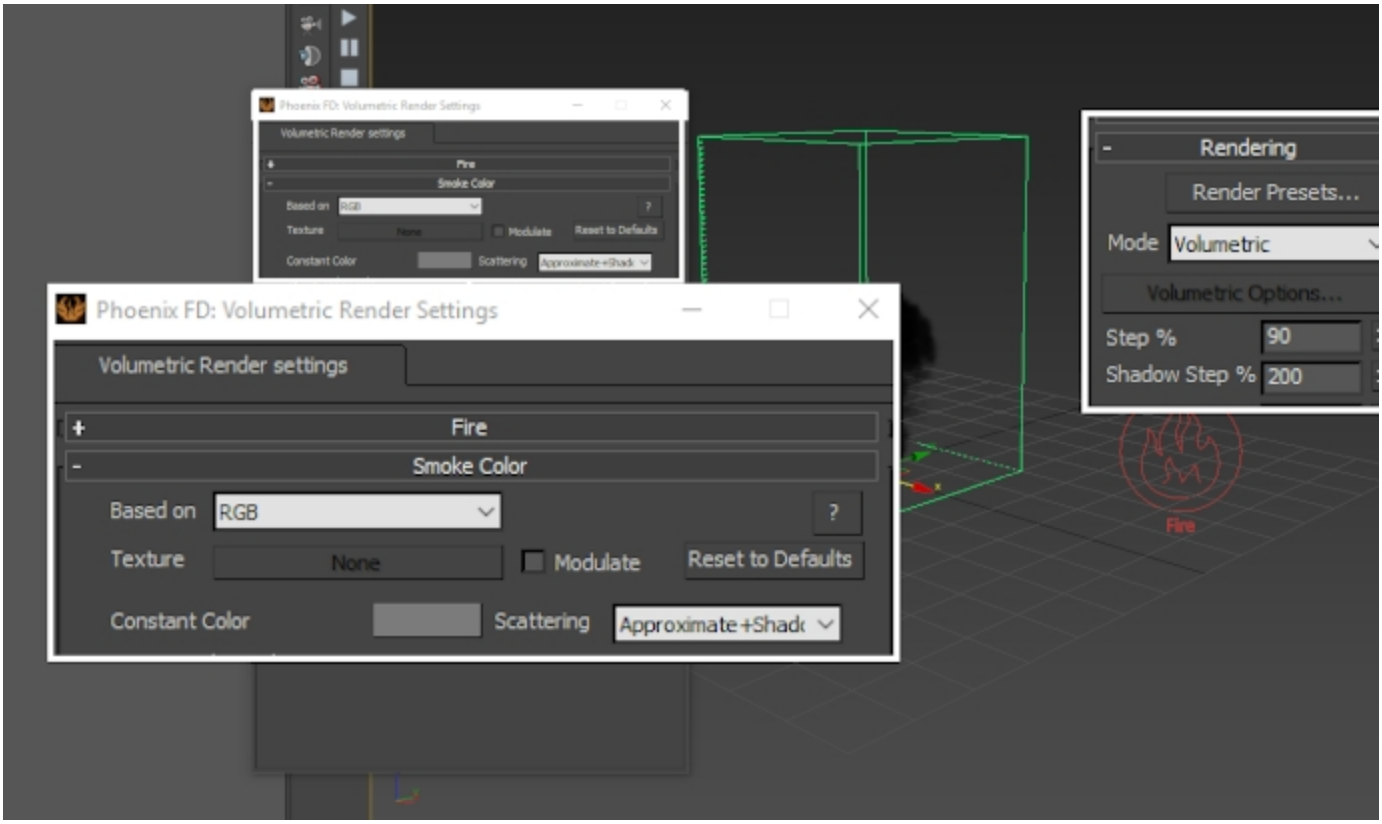


Open the **Phoenix Simulator Rendering** rollout and click on the **Volumetric Options...** button.

From the **Fire** rollout, set the **Based on** to **Disabled**. This will disable the **emissive** component of the **Volumetric Shader**.

Lastly, open the **Smoke Color** rollout and set the **Based on** option to **RGB**. The color of the smoke in the Viewport should go **black**. There are 2 reasons for this:

1. **When setting the Based on option to a certain channel**, **you need to make sure that the specified channel is cached to disk**. This is done from the **Output** rollout **Output Grid Channels**.
2. The default RGB color of the Phoenix Fire/Smoke Simulator is black (0, 0, 0). You need to set the color to another value either from the Phoenix Source, or using a **Phoenix Mapper** (we take a look at the Mapper in Chapter 5).



You can think of the **Volumetric Options** window as the **Phoenix Volumetric Shader**. It holds the majority of the parameters that govern how the supported **Grid Channels** (such as Smoke, Temperature, RGB, Velocity, etc.) are used during the **rendering** stage.

The Volumetric Shader is separated in 3 major sections:

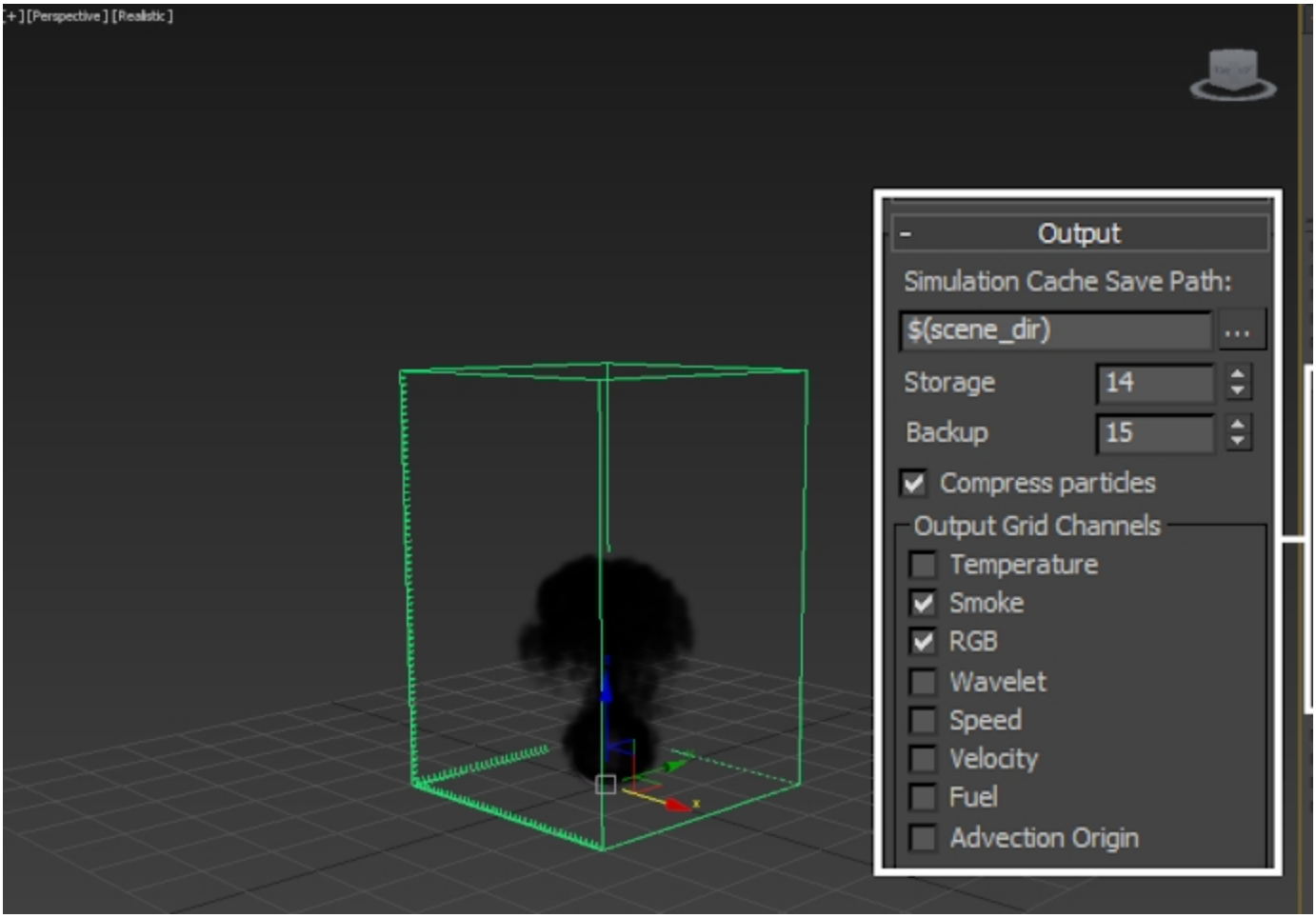
1. The **Fire** rollout controls the **emissive** component of the shader. You can specify what channel is used to emit light, how strong is the emission, what color it is and how it affects the Alpha. If you want to learn more on this topic, you can check the Documentation or go through the Burning Chair tutorial.
2. The **Smoke Color** rollout controls the **shading** of the volume. You can directly set a uniform Constant Color, use a Texture or a simulated Grid Channel - such as RGB. Importantly, the Smoke Color rollout also holds the settings that govern how the volume is affected by external light sources and by it's own emissive component (the Fire).
3. The **Smoke Opacity** rollout is used to specify where the volume is **visible** inside the Phoenix Simulator. While most of the time you'll use either the Simple Smoke, or the Smoke options, it's also possible to set this to Speed, for instance, and tweak the Opacity Diagram below such that "smoke" is visible only in those areas of the container where the Velocity Grid Channel is above a certain threshold.

Go to the **Phoenix Simulator Output** rollout and **disable Temperature** from the **Output Grid Channels** section.

Enable the RGB. You need the RGB Grid Channel stored to disk if you want to use it for the Smoke Color. **This holds true in general - if you want to use a certain Grid Channel in the Volumetric Shader, you need to cache it to disk during the simulation stage**.

The **Output** rollout controls which Grid Channels are saved in the .aur or .vdb cache files on your hard drive. Any time you run a simulation, Phoenix stores those files in the background.

Since we are not utilizing the Temperature grid channel for this tutorial, saving it to disk is unnecessary.

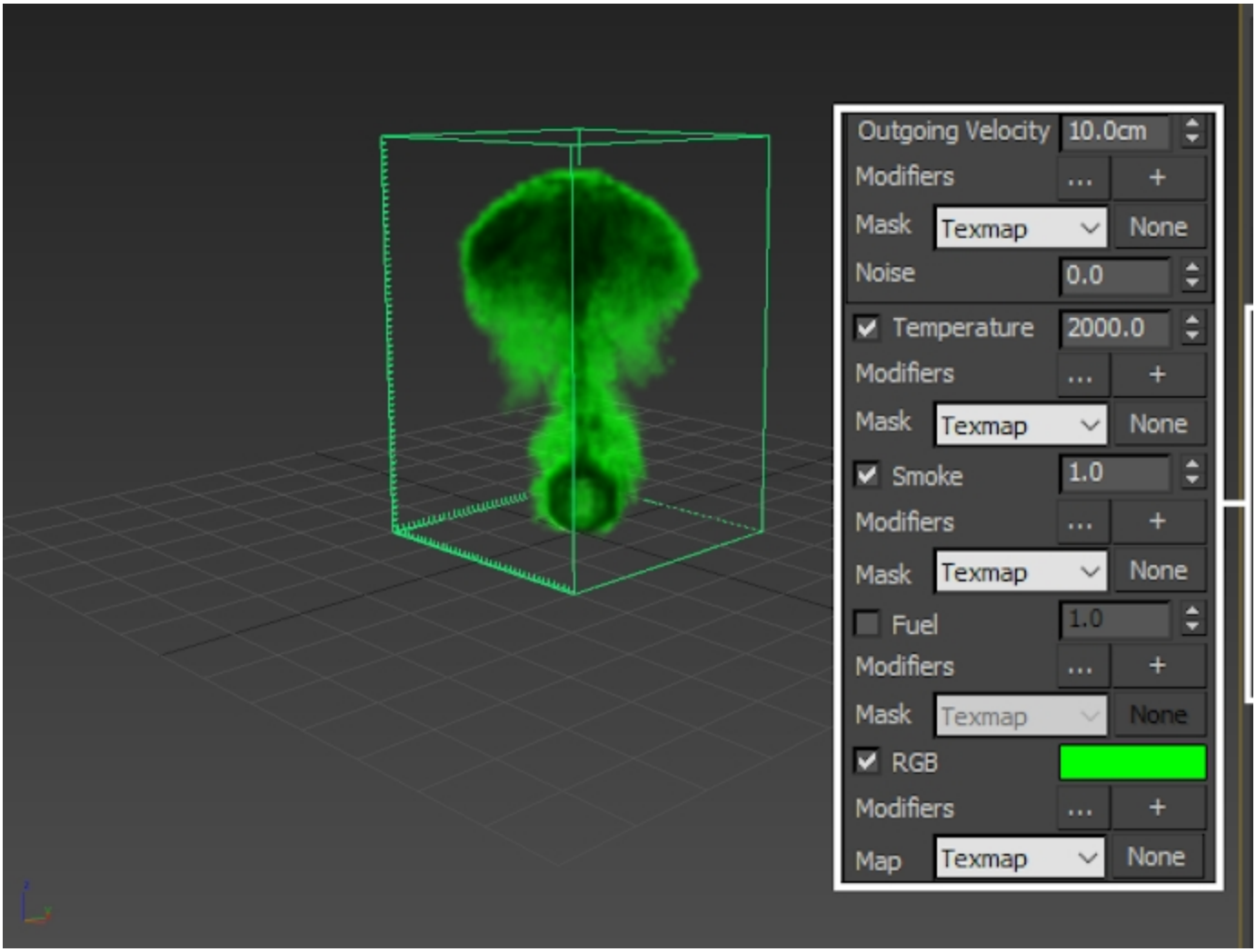


Finally, select the **Phoenix Source** and **enable RGB**. The color swatch to the right (green by default) is used to specify what color is emitted into the RGB channel from the source geometry (the Sphere).

If you run the simulation, the Smoke should now be green.

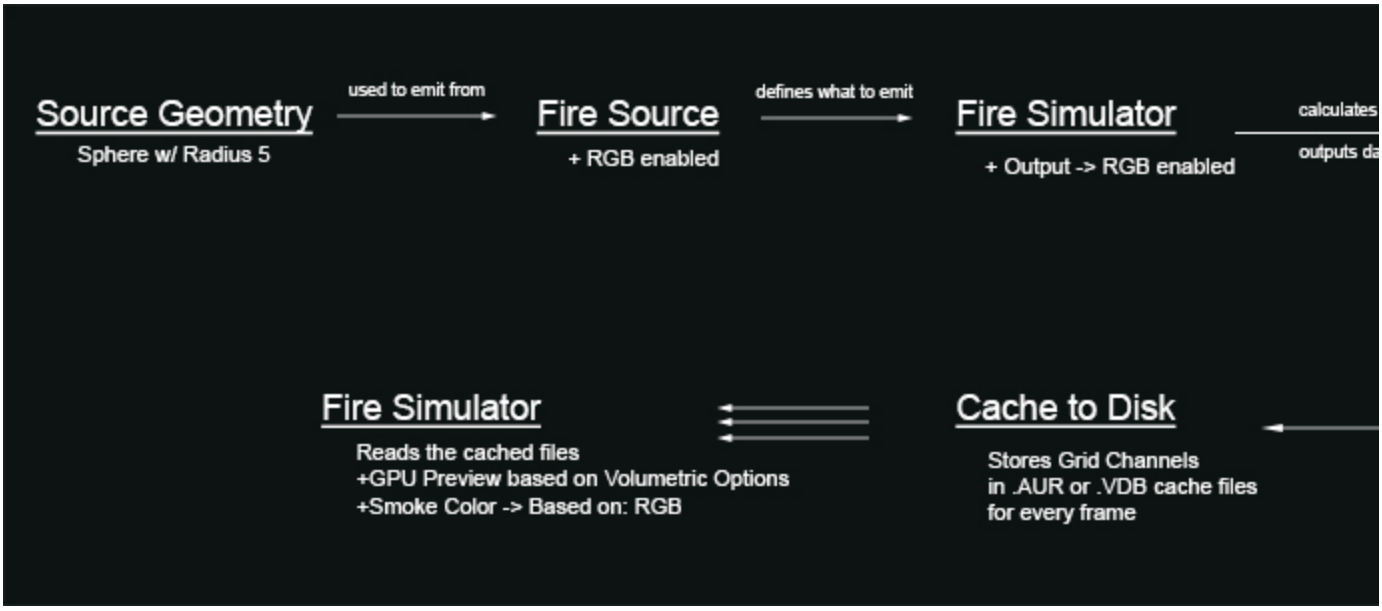
Note that the emission with this setup is uniform. You can have 3 different spheres with 3 separate Phoenix Sources whose RGB emission is set to a different color. In the next chapter we see how to use a single Phoenix Source to emit different colors all from the same geometry.

Hit **Phoenix Simulator Simulation Start**. The color of the smoke in the viewport should now be green.



To recap:

1. Create a source geometry, a Phoenix Simulator and a Phoenix Source.
2. Add your source geometry to the Emitter Nodes list of the Phoenix Source.
3. Enable GPU Preview on the Phoenix Simulator.
4. Open the Volumetric Option from the Rendering rollout of the Simulator.
5. Disable Fire and set the Smoke Color Based on to RGB so the Smoke color is based on the RGB Grid Channel.
6. Enable the export of RGB channel from the Output rollout and disable Temperature to save some disk space.
7. Set the desired emission color on the Phoenix Source. Don't forget to enable RGB from the tickbox.



Chapter 2: Emitting RGB from a texture

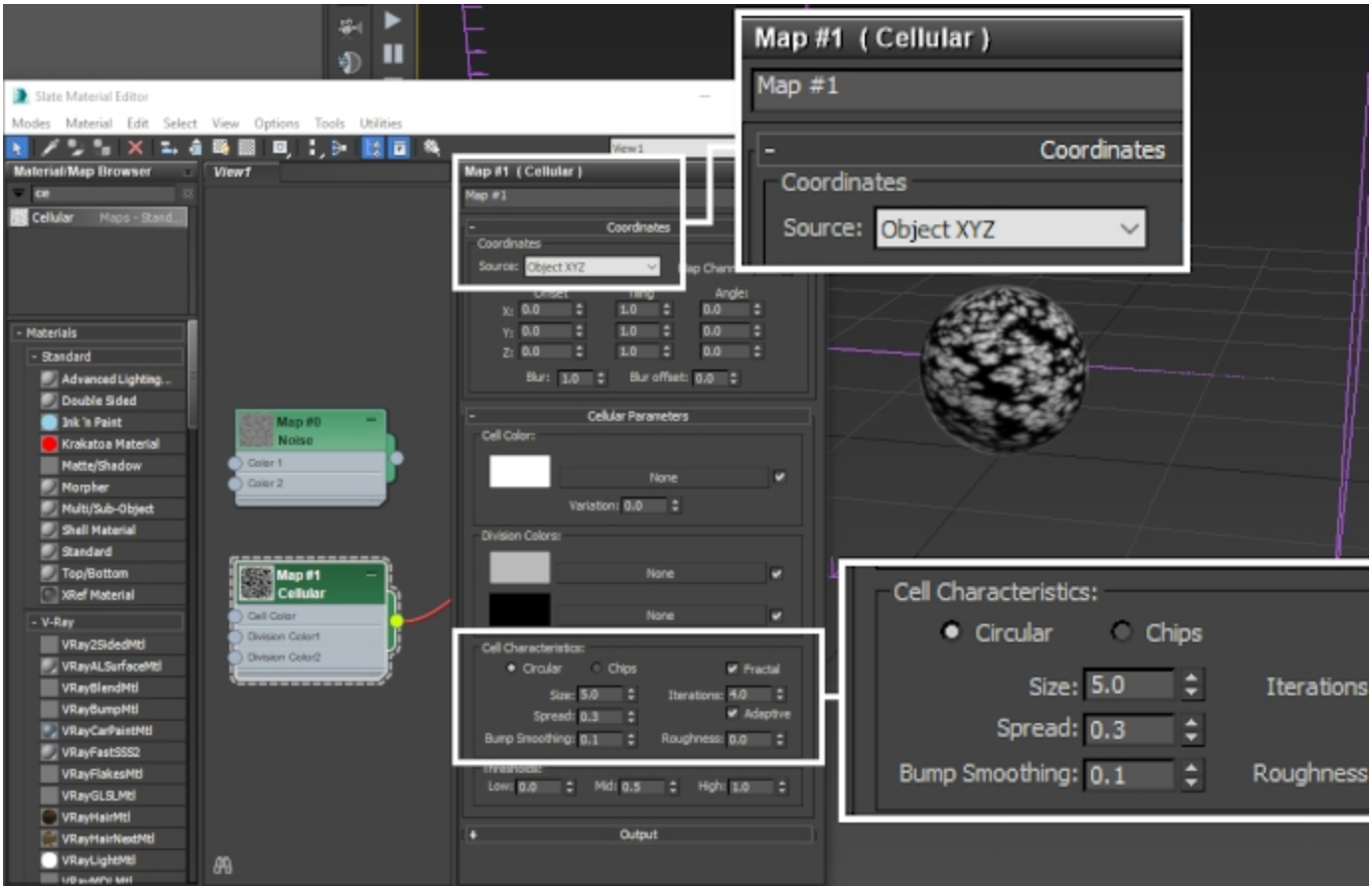
We will continue with the scene from Chapter 1. You can use the Chapter01_Uniform_RGB.max file if you didn't follow along.

Assuming that you already have a setup with a working Smoke Color by RGB, we will simply **assign a new material to the Sphere and plug a Cellular Noise texture to the diffuse slot**.

You may want to use the Trash icon on the Phoenix Toolbar to delete the cache files if the Viewport preview is getting in the way.

If you can't see the colors of the noise texture in your Viewport, Right-Mouse-Button click on the material and select "Show Shaded Material in Viewport".

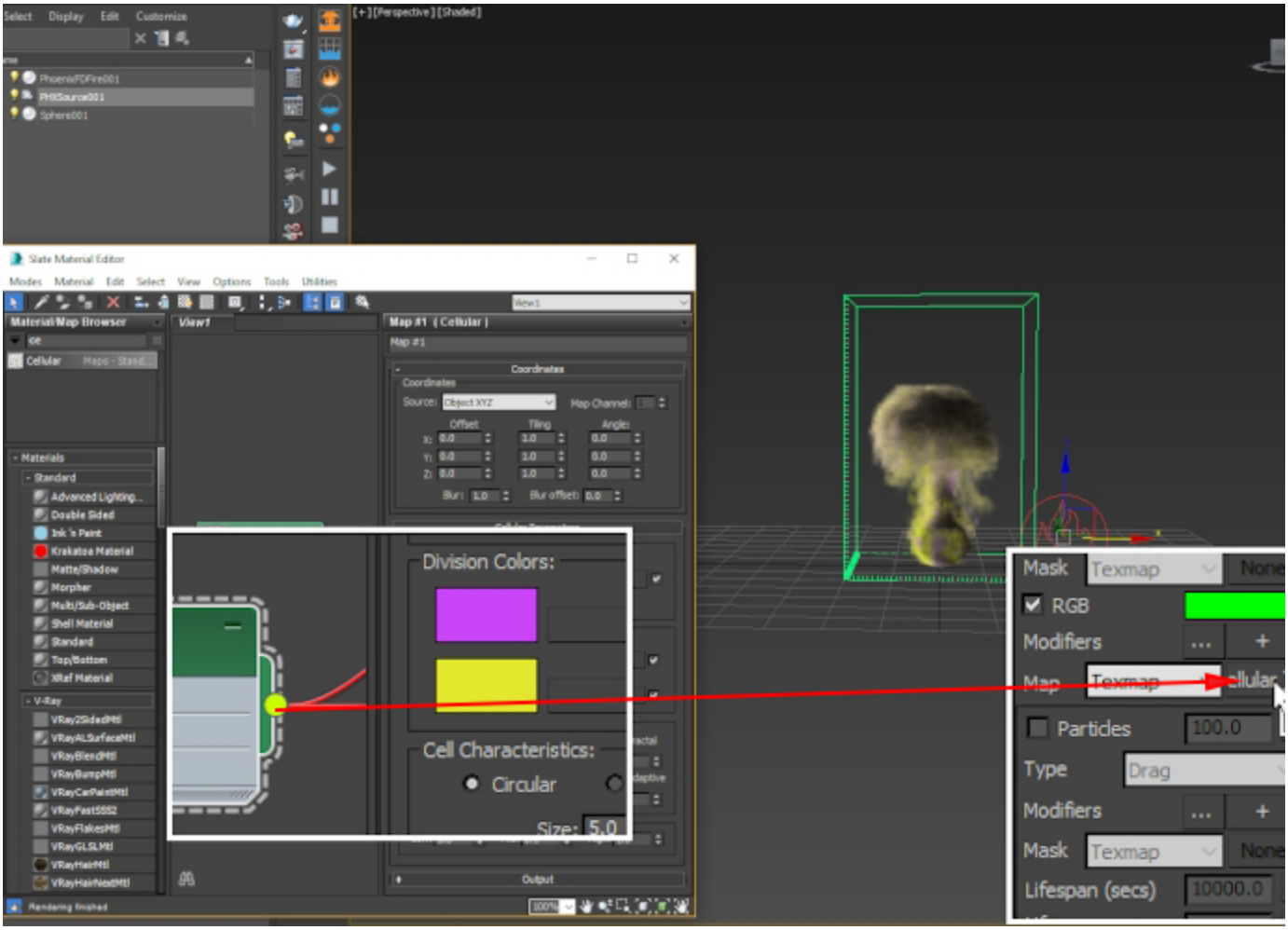
Assigning a material to the source geometry is not a requirement for the texture emission to work. Phoenix does not pay attention to the material assignments on your geometry. We do this so we can easily preview the texture in the Viewport.



Once you're happy with the Noise texture's appearance, **Left-Mouse-Button click and drag from the texture's Output swatch to the RGB Map Input swatch on the Phoenix Source**.

The Source is now aware of the texture and will use it for the emission of RGB.

When using a Texture map with a Phoenix Source in Volume Brush or Volume Inject modes, you need to change the Mapping from Explicit (which uses the UVs of the object) to one of the Planar modes. When Phoenix turns your object into a volume to use for emitting contents, the texture has to be sampled not only on the surface of the object (which is what the UVs are for) but inside its volume as well. Therefore, if the Mapping is set to Planar, the volume of your object can be properly filled based on the values of the texture.



Hit **Phoenix Simulator Simulation Start** to run the sim.

The color of the Smoke should now be based on the texture you applied to the Source. This holds true for all textures, not just the Noise. For example, you could manually paint a texture in Photoshop and use that in the source.

Chapter 3: Emitting RGB based on the Vertex Color of the Source Geometry

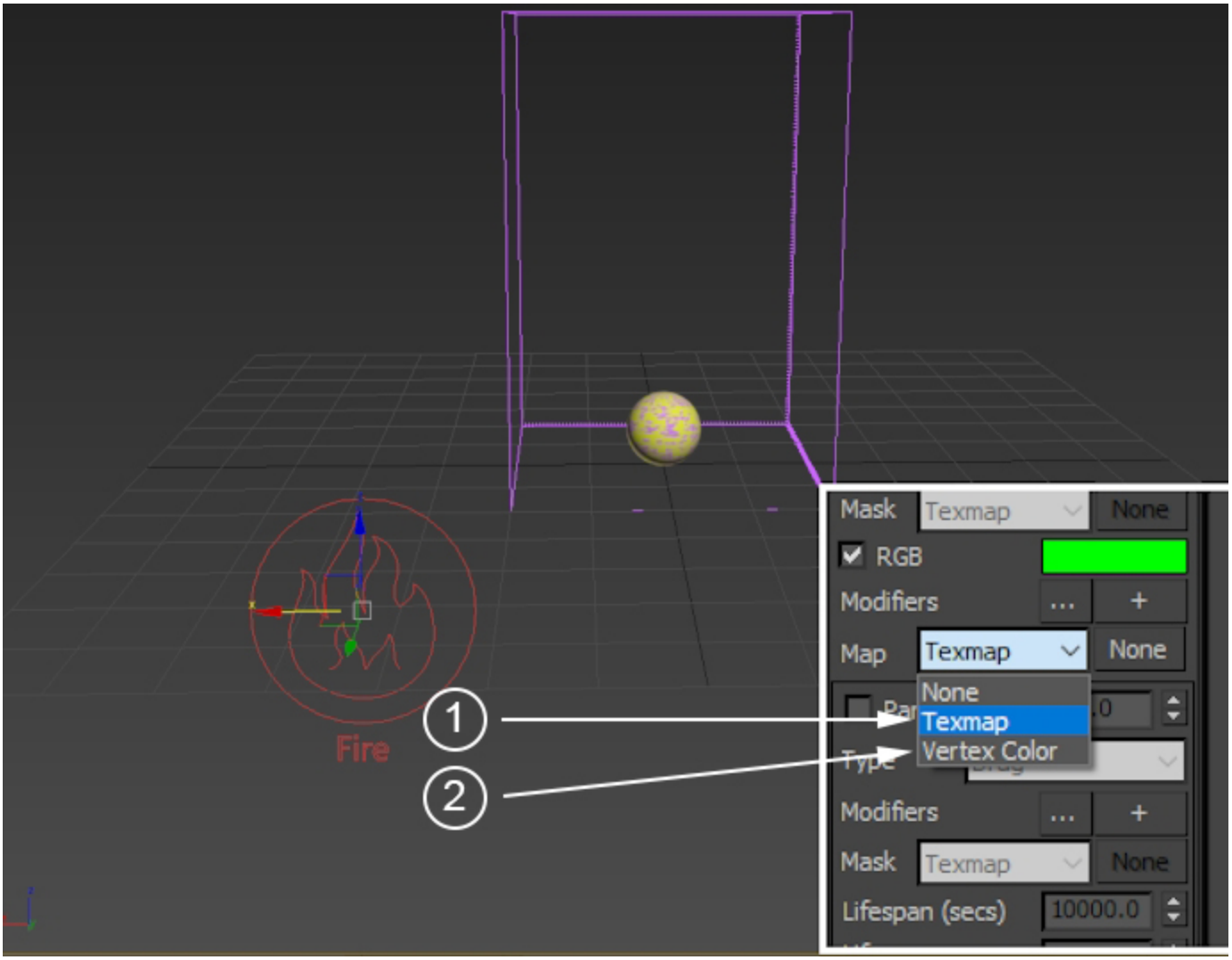
Continuing with the scene from Chapter 2, delete the cache files or hide the Phoenix Simulator so it doesn't get in the way.

We will quickly go over painting vertex color on our source geometry with the VertexPaint modifier.

We will then use that in the Phoenix Source for RGB emission. There are 2 ways to do this:

1. Use a **V-Ray Vertex Color texture** in the same workflow as described in Chapter 2 or
2. Set the **RGB Map parameter of the Phoenix Source to Vertex Color** (it is set to Texmap by default).

Both options should produce the same result.



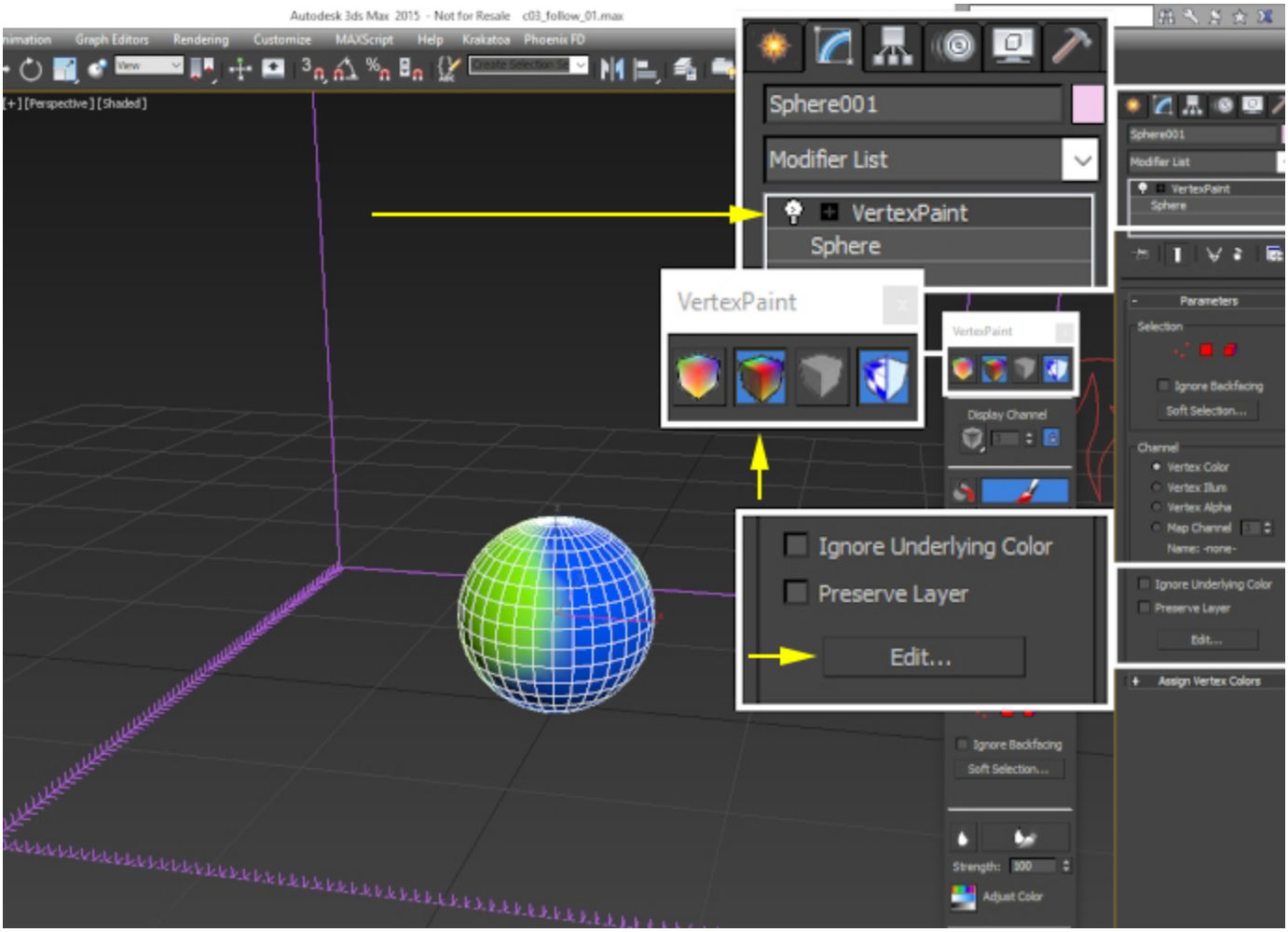
Unhook the Cellular Noise texture from the material applied to the Sphere.

Assign a VertexPaint modifier to the Sphere - if the window with the VertexPaint settings does not appear for you or you've accidentally closed it, hit the Edit... button. Feel free to paint whatever colors you see fit.

Under the VertexPaint window, make sure to enable the display of vertex color by choosing either the first or the second icons on the top at the top of the toolbar.

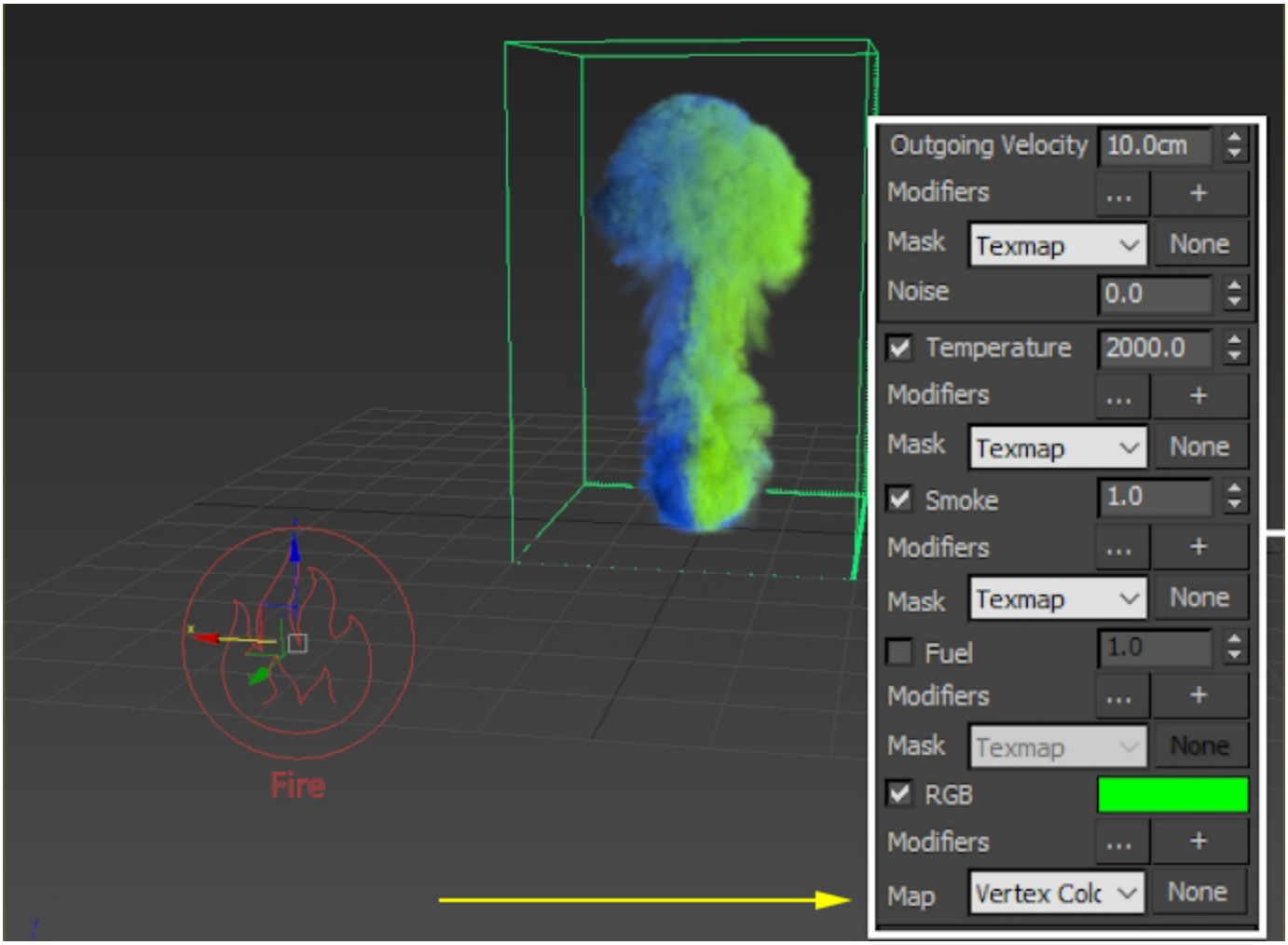
The color swatch next to the Color Picker icon is used to select the paint color.

You can use the Brush icon to invoke the Brush tool, or the Bucket icon next to its left to flood the object with the selected color.



Select the **Phoenix Source** and **set the RGB Map option to Vertex Color**.

Hit the **Simulator Start** button to run the simulation. The RGB emission should now be based on the colors you painted on the Sphere.



Chapter 4: Changing the RGB color of simulations after emission with a second Phoenix Source in Brush mode

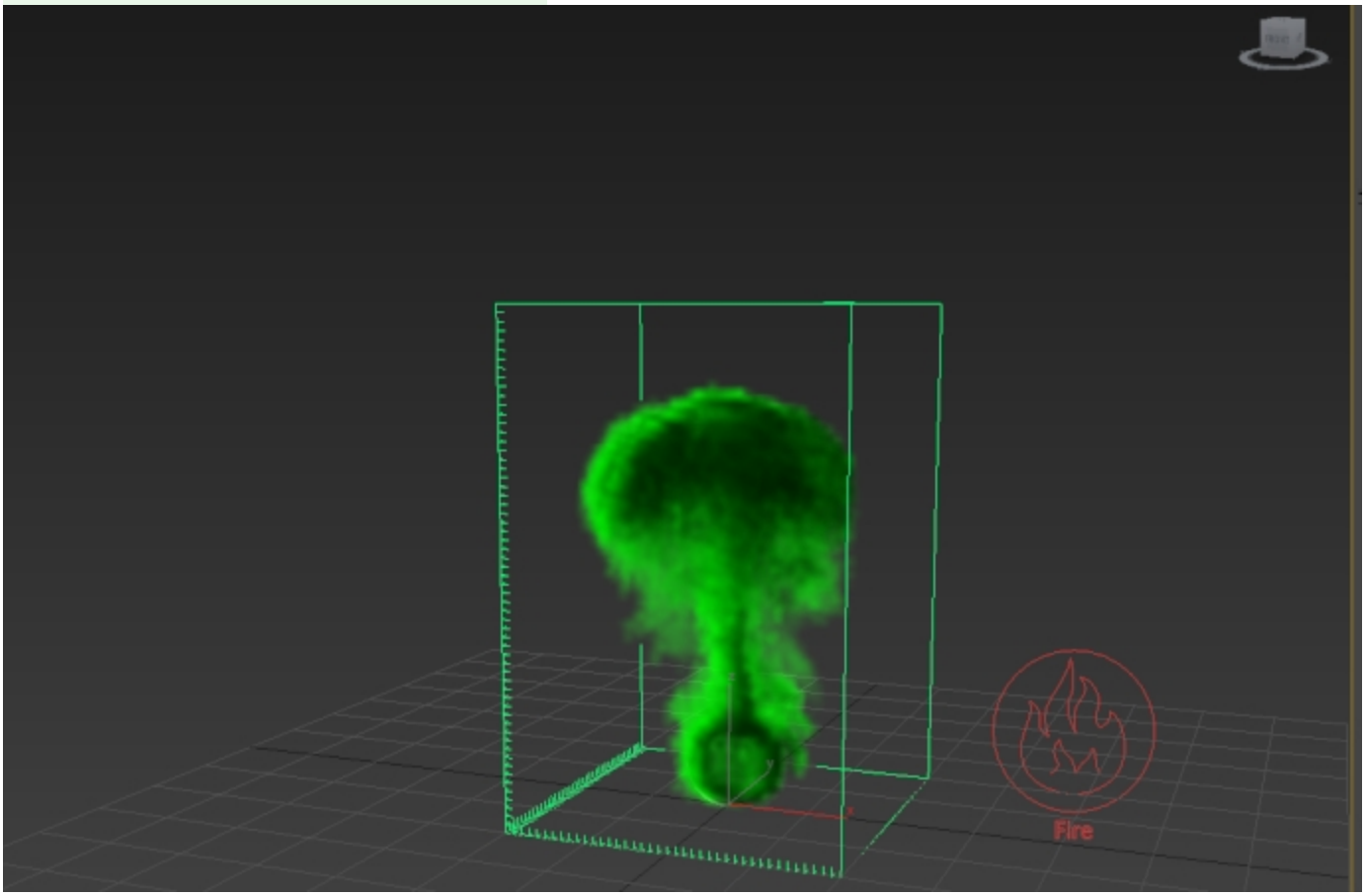
Chapters 1 through 3 covered the basics - how to set up the export, preview and rendering of the RGB channel, and how to emit into the RGB channel from a piece of geometry using the Phoenix Fire/Smoke Source.

In this chapter and the next, we go over 2 separate workflows which allow you to **modify the RGB channel after emission**. We will first take a look at the simpler of the two - using a **Phoenix Source in Brush Mode**.

To start off, re-create the setup from Chapter 1, or open the provided project file called Chapter04_Brush_RGB_Only.max.

1. Create a source geometry, a Phoenix Simulator and a Phoenix Source.
2. Add your source geometry to the Emitter Nodes list of the Phoenix Source.
3. Enable GPU Preview on the Phoenix Simulator.
4. Open the Volumetric Option from the Rendering rollout of the Simulator.
5. Disable Fire and set the Smoke Color Based on to RGB so the Smoke color is based on the RGB Grid Channel.
6. Enable the export of RGB channel from the Output rollout and disable Temperature to save some disk space.

7. Set the desired emission color on the Phoenix Source. Don't forget to enable RGB from the checkbox.

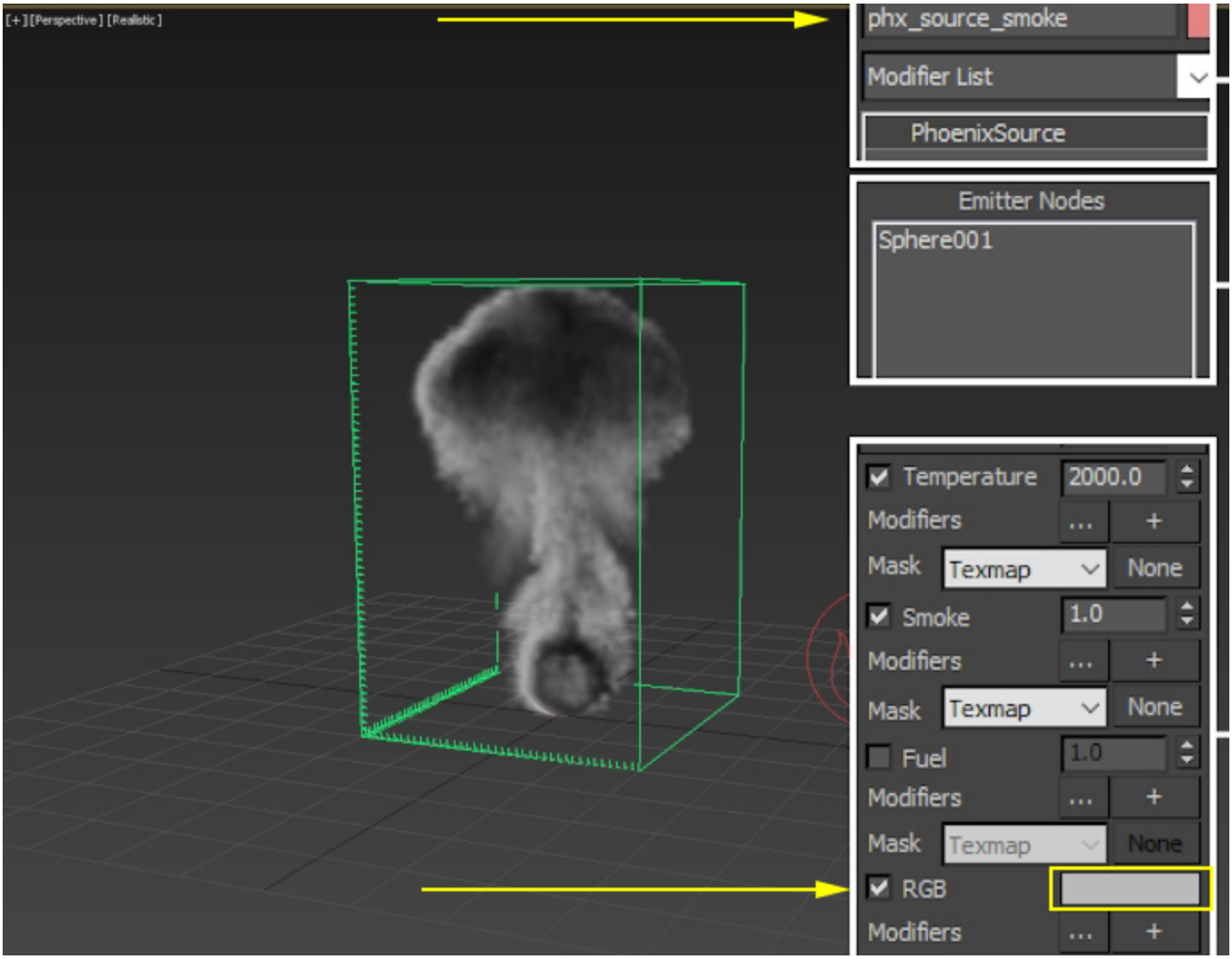


Set the **RGB** on the **Source** to **mid-gray** (RGB on the Fire Source is green by default).

Also, rename the Source to **phx_source_smoke** or something along those lines. In the next step we add a second one so keeping organized will be quite helpful.

As the Smoke comes out of the Sphere, its color should now be gray because we set the RGB on the source as such. What if we could somehow change the color once the smoke reaches a certain area of the simulator, or touches a certain object? For that to happen, **you need to find a way to modify the RGB channel**.

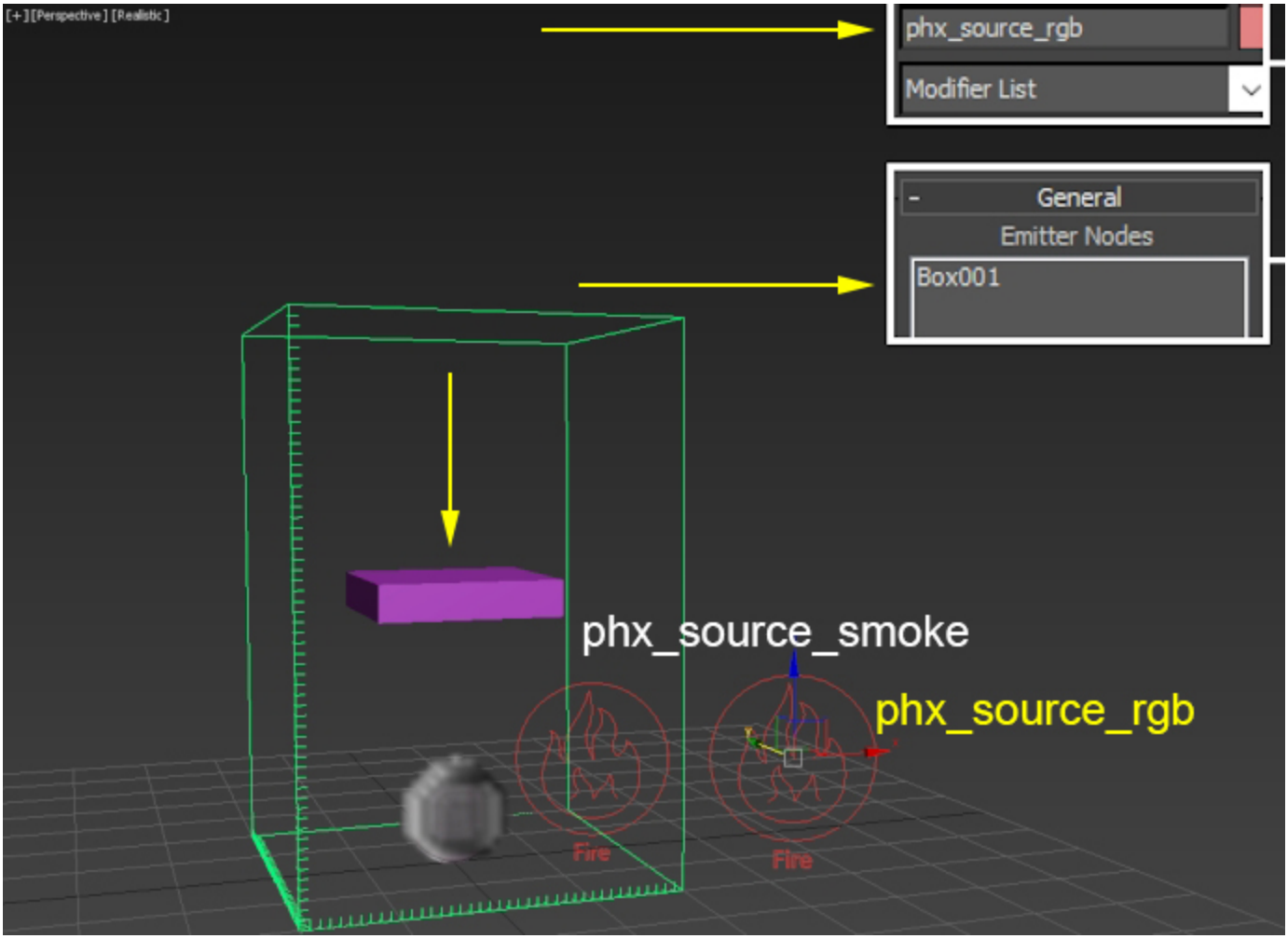
Remember - the Smoke does **not** magically carry the color by itself - **there is an entirely separate RGB Grid Channel being simulated** and the color of the smoke is being based on it. If you set the RGB Channel for the entire bounding box of the simulator to 3 separate colors, the Smoke will happily travel through and inherit them.



Create another **Phoenix Fire/Smoke Source** and rename it to **phx_source_rgb** to keep your scene tidy.

Then, **create a box and position it somewhere along the path of the smoke**. We will set the box as the Emitter object for the phx_source_rgb Phoenix Source and **only emit RGB from it**.

As a result, **when the smoke moves through the volume of the box, it will inherit whatever color we've specified on the RGB swatch** of phx_source_rgb.



Add the box to the **Emitter Nodes** list of **phx_source_rgb** and **set** the **Emit Mode** to **Volume Brush**. You will be prompted with a **message** stating that only **Non-Solid** emitters can be used in Brush Mode and Phoenix can covert the object to Non-Solid for you - choose Yes.

Disable Temperature and Smoke and enable RGB.

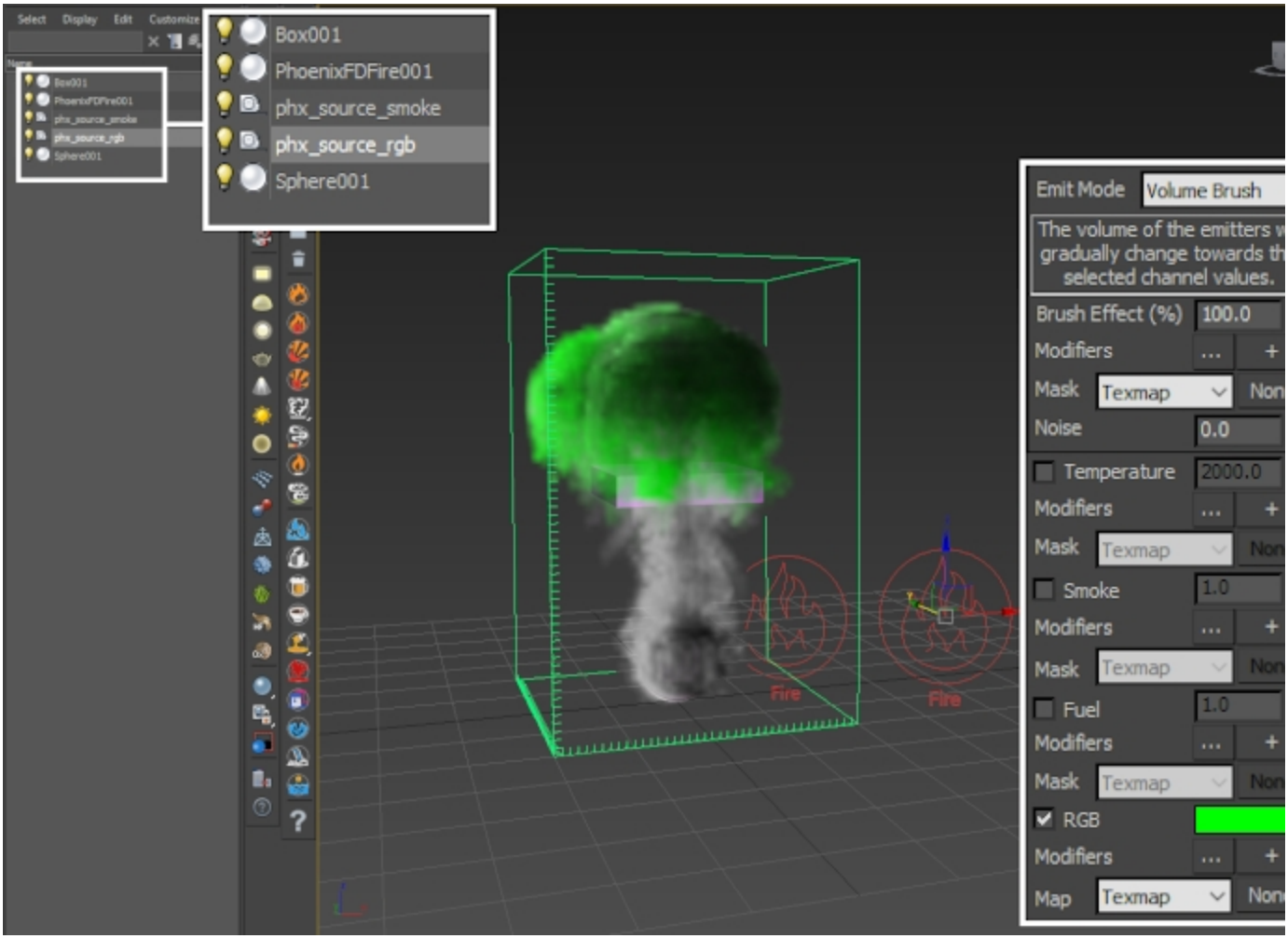
If you now **Start** the simulation, the smoke should be gray and only turn green once it passes through the box.

You can control how fast the transition happens from the **Brush Effect (%)** parameter on the source. At 100, the color will change immediately while a lower value will produce a slower transition over time.

The Volume Brush and Volume Inject modes use the entire volume of the specified emitter objects, as opposed to the Surface Force Mode which emits only from the outer surface (basically, the shell) of the object.

Furthermore, if the box is Solid the smoke will collide with it instead of passing right through which is not what we want.

As a side note - you can bring up the Phoenix Per-Node Properties for any object in your scene with the Right Mouse Button.



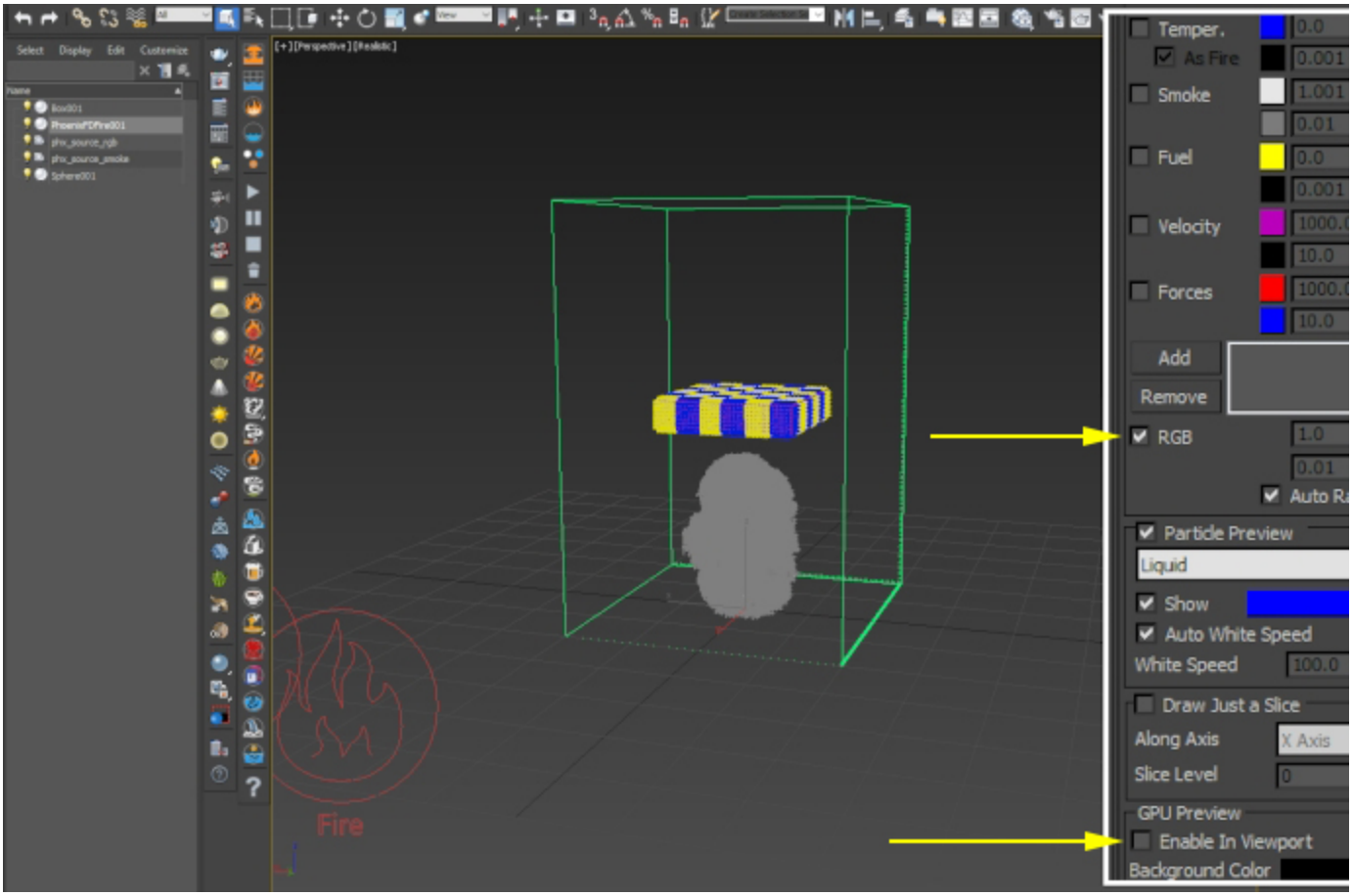
The methods discussed in Chapters 1-3 can be applied here as well - you can modify the RGB emission of phx_source_rgb with a regular texture or a Vertex Color map. Try it out!

Instead of using the Smoke channel as the middle-man, **you can preview the RGB Channel directly**. To do so, go to **Sim ulator Preview** rollout and **disable GPU Preview**.

Then, in the **Voxel Preview** section, **enable RGB** at the bottom and **disable all the other channels**.

You should now be able to see the values in the RGB channel even before the Smoke comes into contact with it.

The voxel preview can be quite useful for debugging your setup.



Chapter 5: Changing the RGB color of simulations after emission with a Mapper and a V-Ray Distance Texture

In this final chapter, we will re-create the setup from Chapter 4 using a **Phoenix Mapper** and a **V-Ray Distance Texture**.

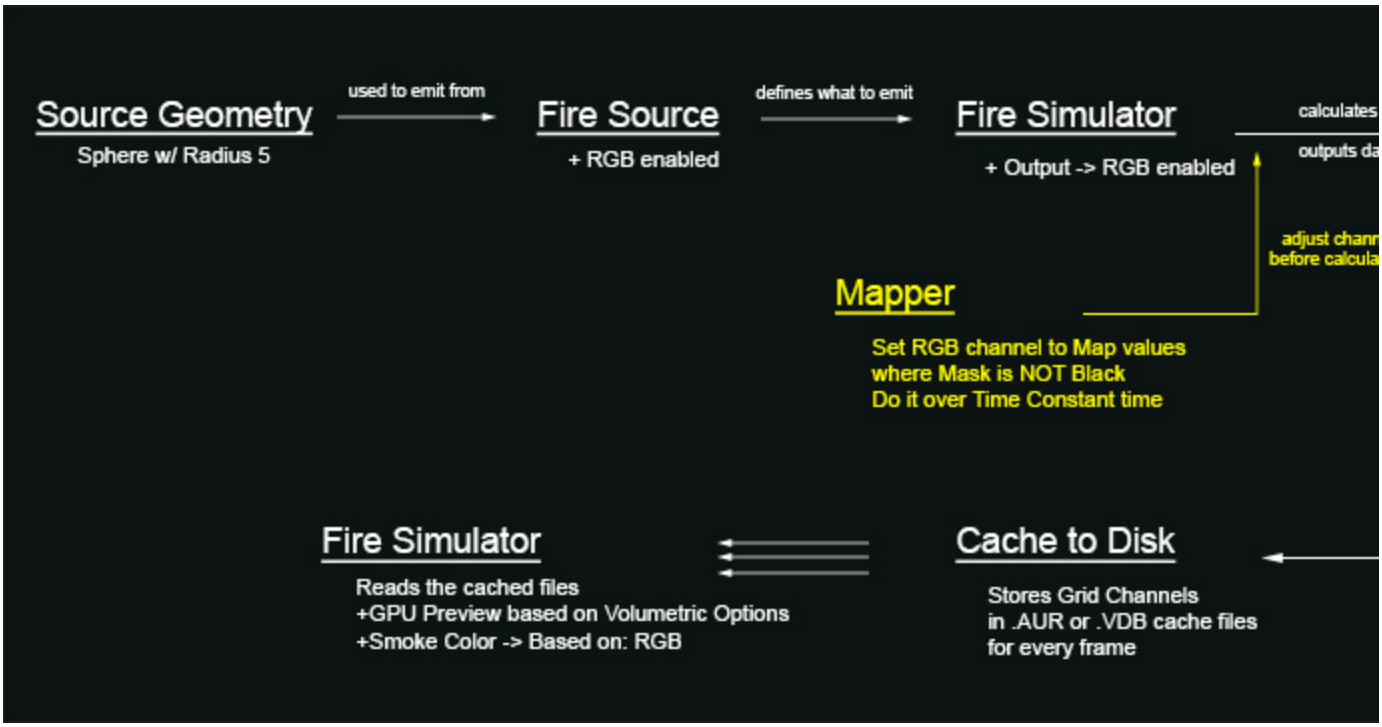
The **Distance Texture** will be used as a **Mask** for the **Mapper** which will **constrain the Mapper** to setting the values of the RGB Channel only in proximity of the object specified in the **D istance Texture Objects** list.

As this setup can be a bit harder to wrap your head around, here's a detailed explanation:

1. The V-Ray Distance Texture has 4 parameters to pay close attention to: the Objects list, Far Color, Near Color and Distance. When you apply the texture to object A, it returns a color value between Near Color and Far Color based on the distance between object A and the objects in the Objects list. This is done for each shading point of the object A - the texture will calculate the distance between the shading point and the objects in the list and return a certain value.
2. The Phoenix Mapper is used to set the values for **ea ch individual cell/voxel** of the Simulator. This is an important distinction - it doesn't simply flood the entire container - you can specify where (Mask) the value (Map) is applied.
3. You can think of the Phoenix Mapper as an object that you're applying the Distance Texture to. Instead of shading points, we now work with the cells/voxels

of the Simulator. The texture will return a value based on the distance between each individual voxel and the geometry objects in the list.

4. The result of this can be used as a Mask to feed to the Mapper. You can then give it a Map and its color will only be applied to those voxels where the Distance Texture evaluates to a value greater than 0 (black).



Starting with the final scene from Chapter 4, delete the **phx_source_rgb** but keep the box geometry.

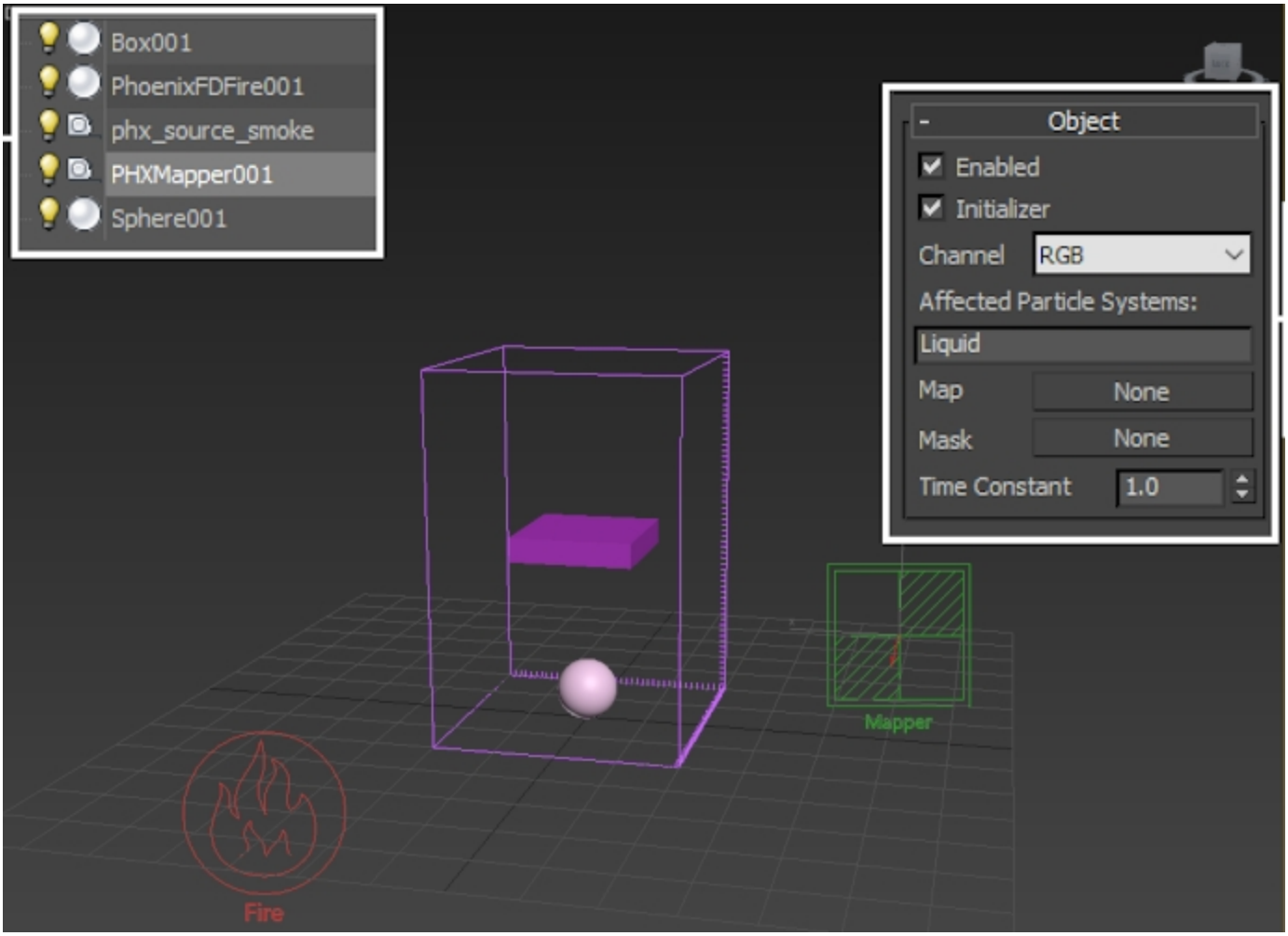
Go to **Create Helpers PhoenixFD PHXMapper** and create one in your scene. **The position of the Mapper in your scene does not matter** - you can place it anywhere you see fit.

Set the **Channel** parameter to **RGB** so the Mapper knows to **affect** the **RGB Channel** of the **Simulator**.

The **Affected Particle Systems** parameter defaults to **Liquid** and can be used to specify what Particle Systems will be affected by the Mapper in the case that you are using a **Phoenix Liquid Simulator**.

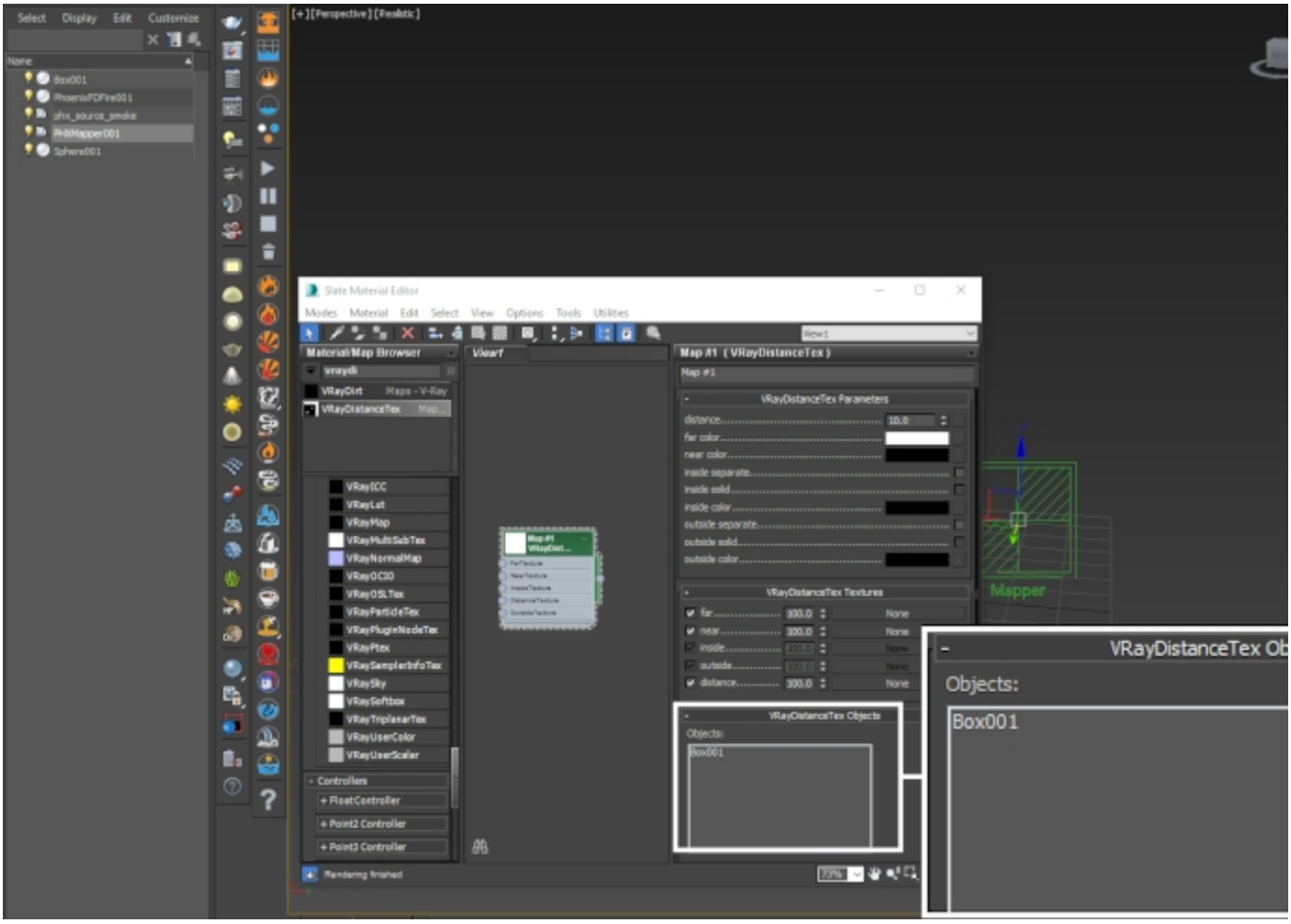
As we are using a **Fire/Smoke Simulator**, this parameter will have **no effect** on the simulation.

If you're running a Liquid simulation and you want to affect the color of the particles, add the respective particle system name to the **Affected Particle Systems** list (i.e. Liquid, Foam, Splash etc.).



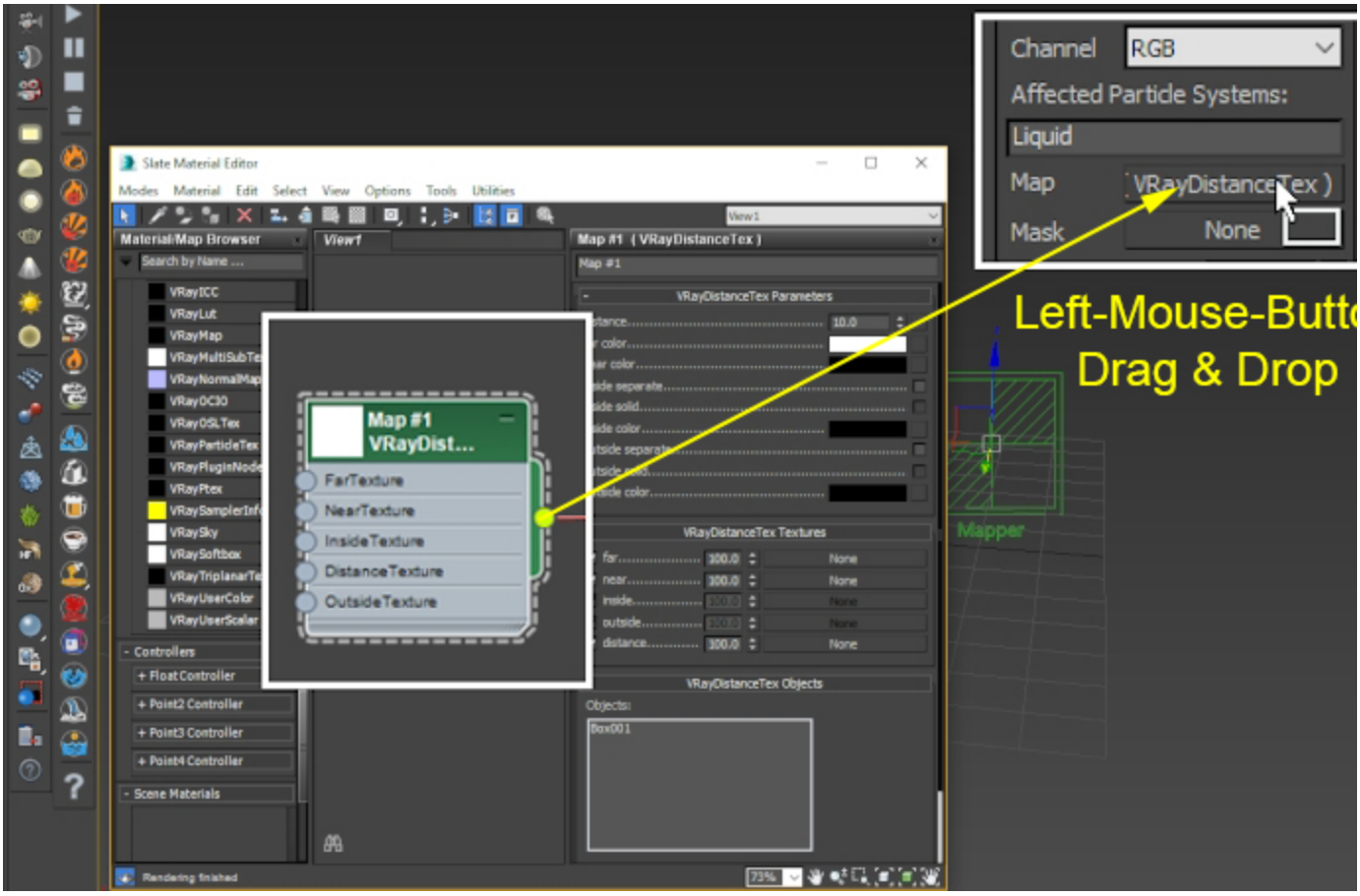
Open the **Material Editor** and create a **V-Ray Distance Texture**.

Add the **Box001** object to the **Objects** list and leave everything else at default.



Select the **Phoenix Mapper** node and **Left-Mouse-Button click and drag** from the **V-Ray Distance Texture's output** swatch to the **Map parameter on the Mapper**.

Earlier we mentioned that we will use the **Distance Texture** as a **Mask** so you may be wondering why are we putting it into the **Map** slot instead. It's just **for preview** - the Map parameter is what is actually set so it's easier to plug the texture here, set it up correctly and then move it to the Mask slot.



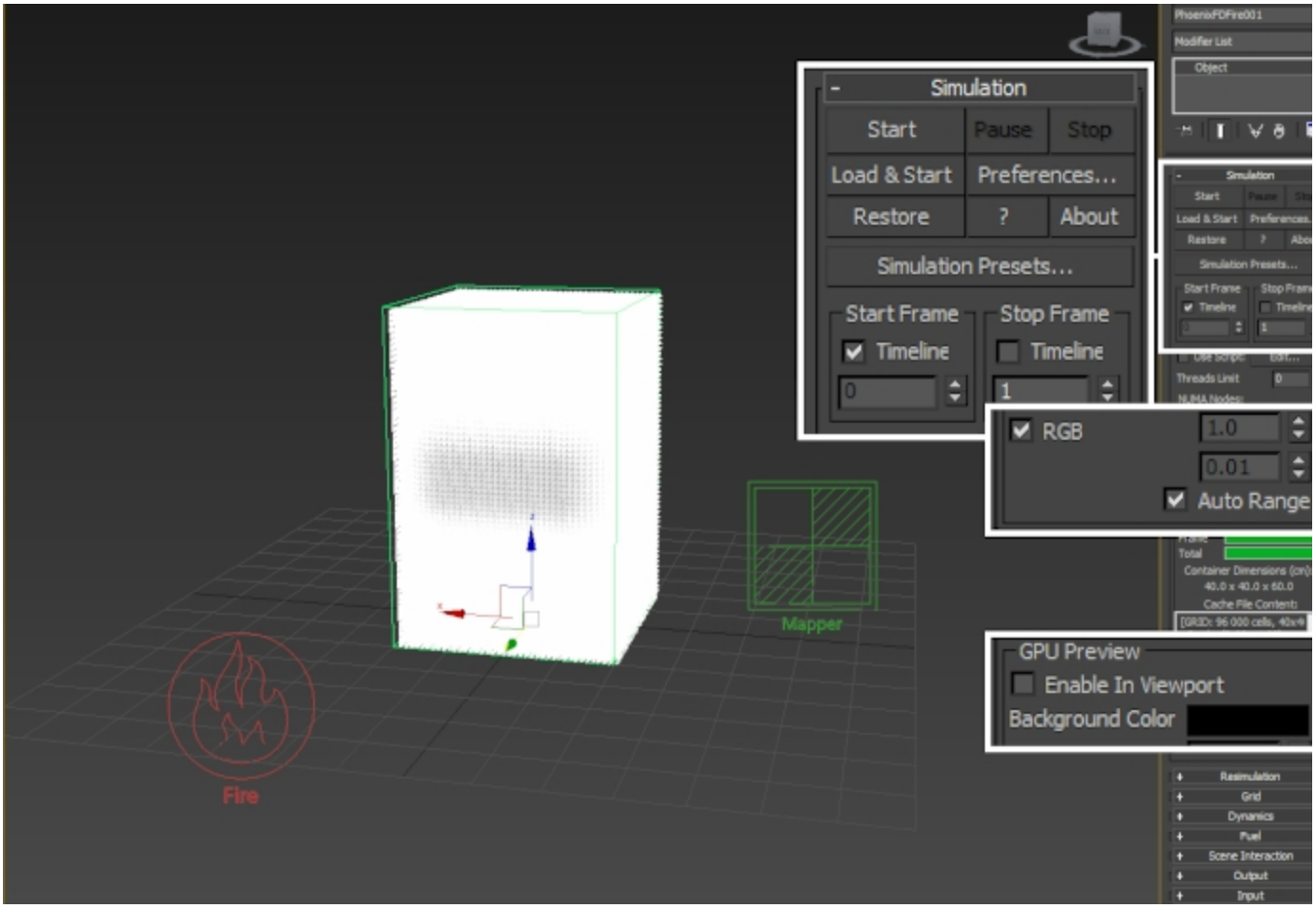
Open the **Preview** rollout of the Phoenix Simulator and **disabl e GPU Preview**.

Enable Voxel Preview RGB and disable Smoke, Temperature, etc. if those are enabled.

The voxel preview will allow you to easily check how the Mapper is affecting the RGB Channel.

Go to the Simulation rollout and **uncheck Timeline** under **Sto p Frame**. **Set the value to 1** so the simulation will only run for one frame.

Hit **Start**.



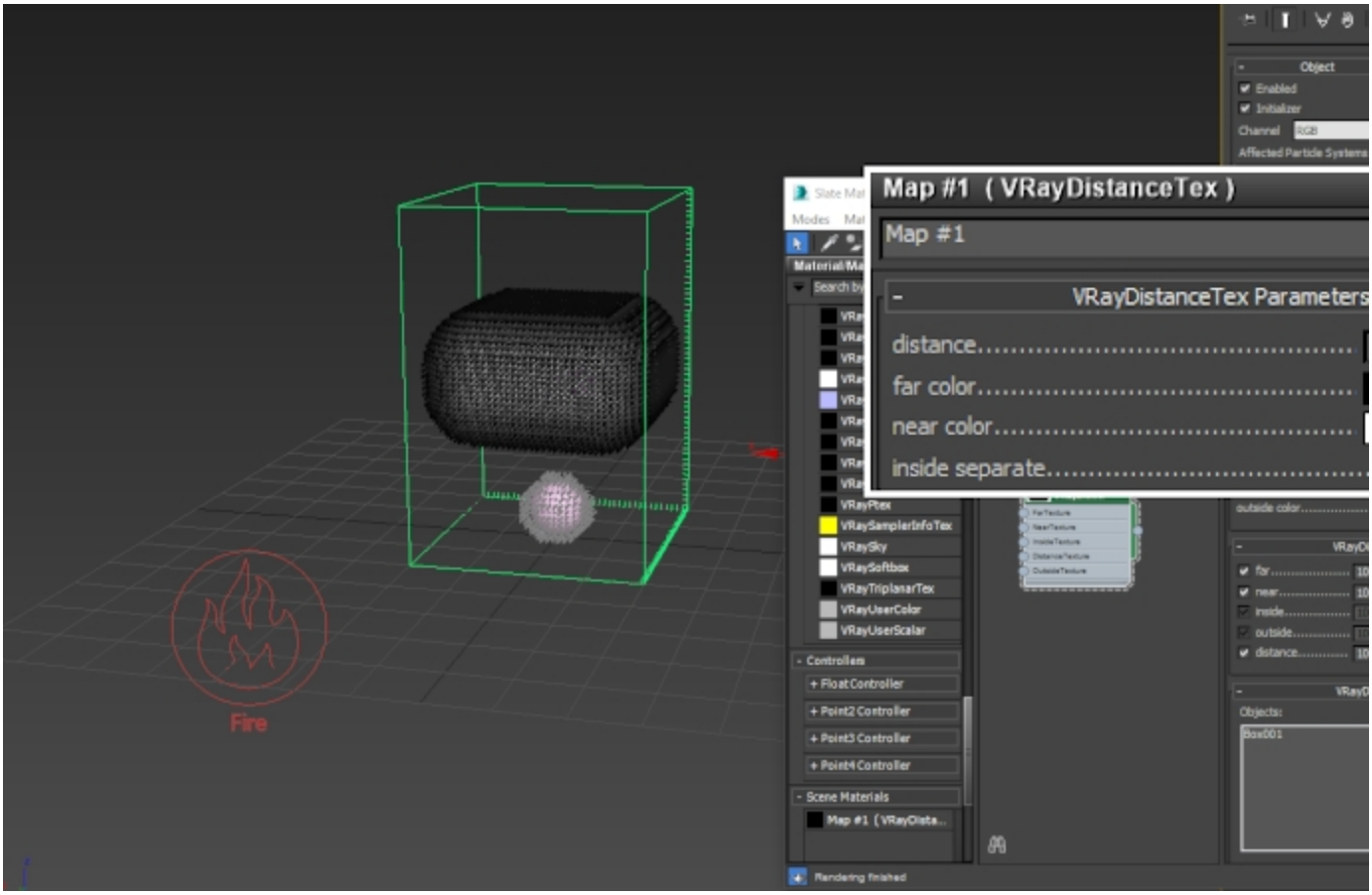
The entire RGB Channel should now be white.

Instead, **we want only the voxels near the object to be white** and the rest of them - black. That way, when we put the Distance Texture in the Mask slot, it will only affect the voxels close to the object.

The reason for this is the settings of the V-Ray Distance Texture. By default, the Far Color is set to white (therefore the RGB of the voxels far away from the object is set to white), and the Near Color is black (so the RGB of the voxels close to the object is set to black).

Reverse the Far and Near Color - set the Far Color to black and the Near Color to white.

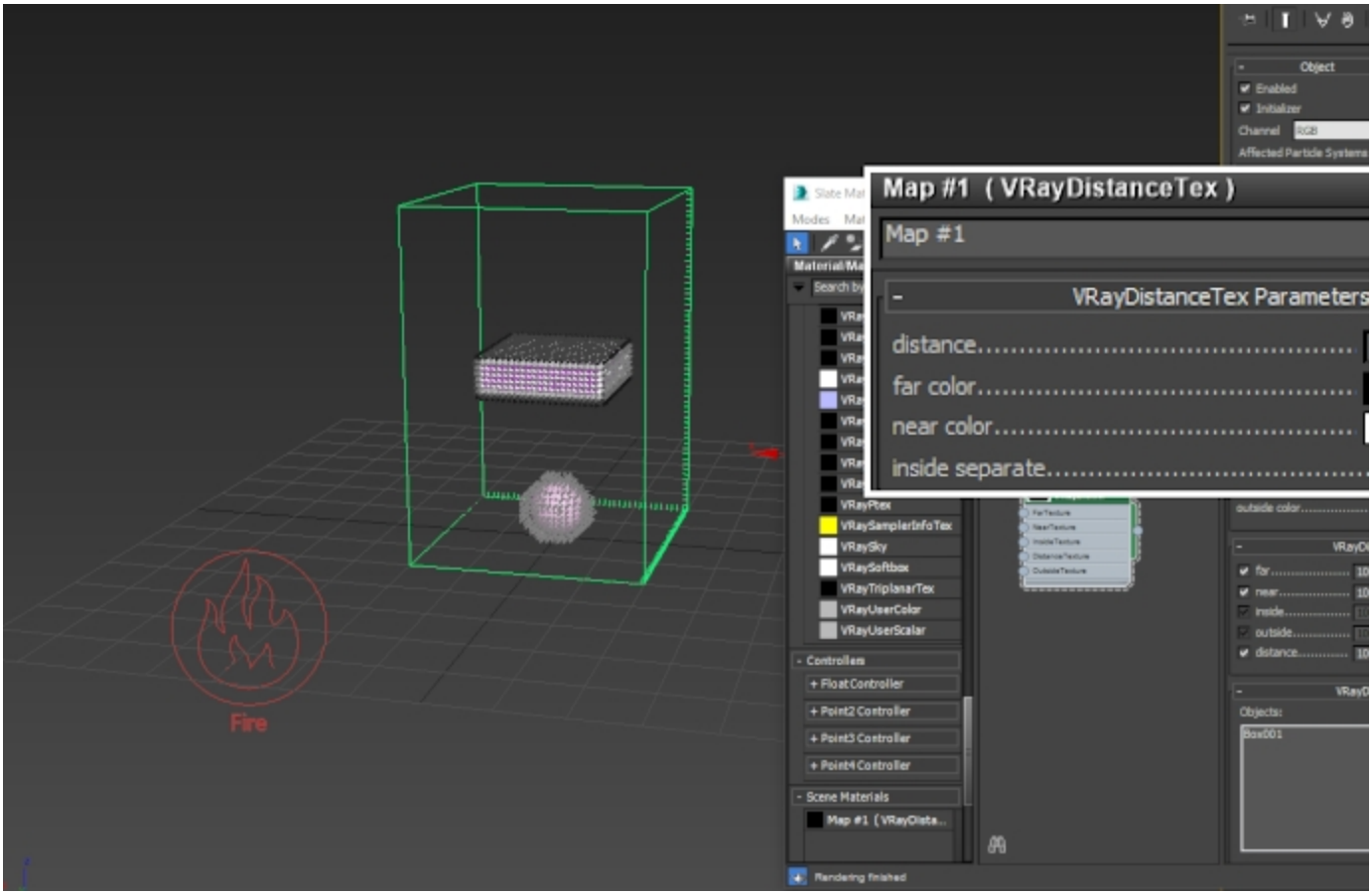
Hit **Start**.



You should now see an area around the box transitioning from white to black. The thickness of this 'wrapper' around your object is controlled by the Distance parameter on the Texture.

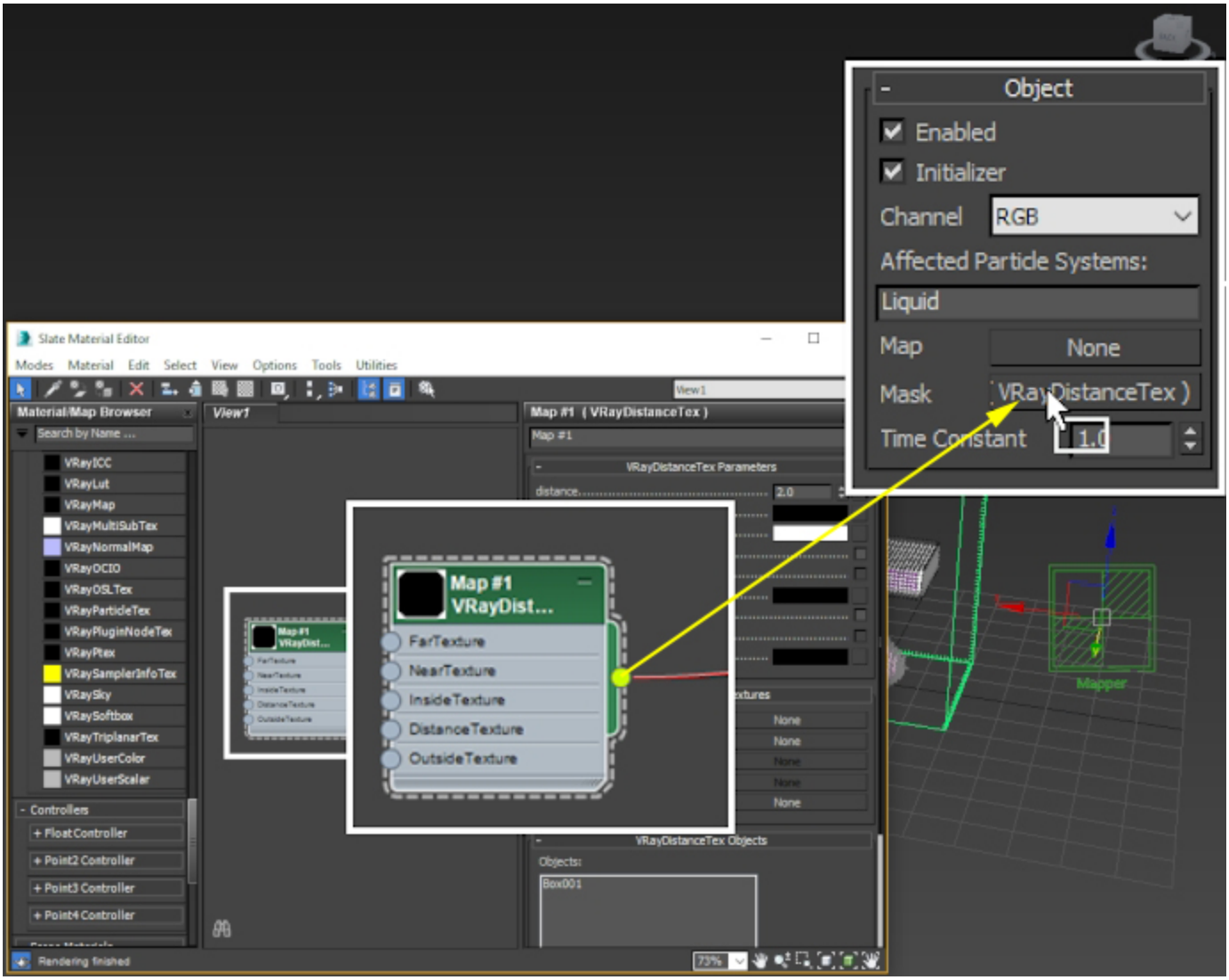
Reduce the **Distance** parameter on the **V-Ray Distance** Texture to **2**.

Hit **Start**.



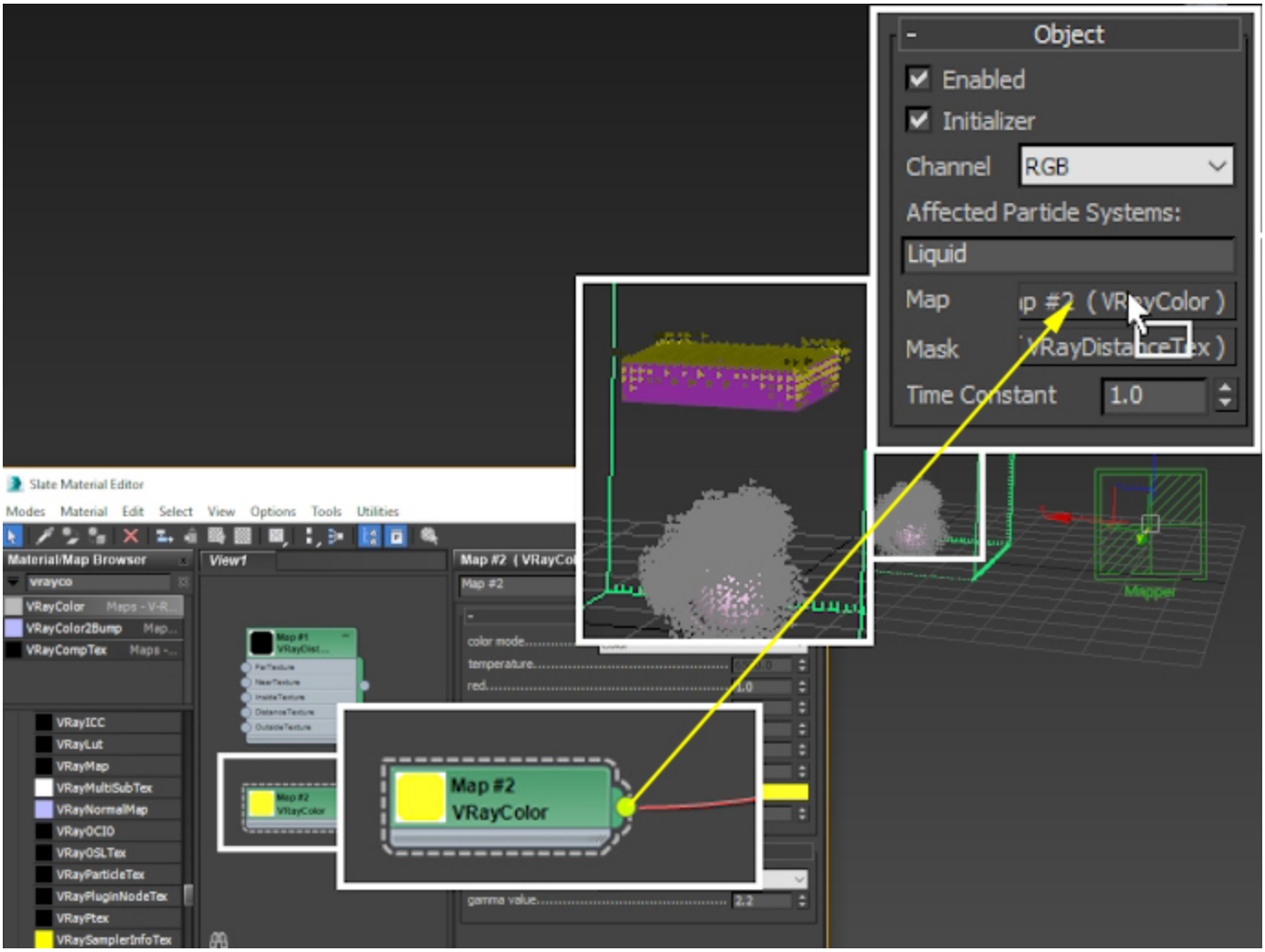
The 'wrapper' should now fit tightly around your object. That's all you need to do for the V-Ray Distance Texture.

Plug the **Distance Texture** into the **Mask** slot of the Phoenix Mapper and **remove it from the Map slot** with Right-Mouse-Button Clear.



Add a **VRayColor** to the **Map** slot (or any other texture - that's up to you). Don't forget to set the Coordinates Mapping to Planar from Object if using a texture map.

Increase the Simulation Stop Frame value to 10 or so and hit **Start** to run the simulation.



If you're looking at the voxel preview, you will notice that the RGB is correctly set up at the start of the simulation but once the smoke reaches it, it gets diluted and disappears.

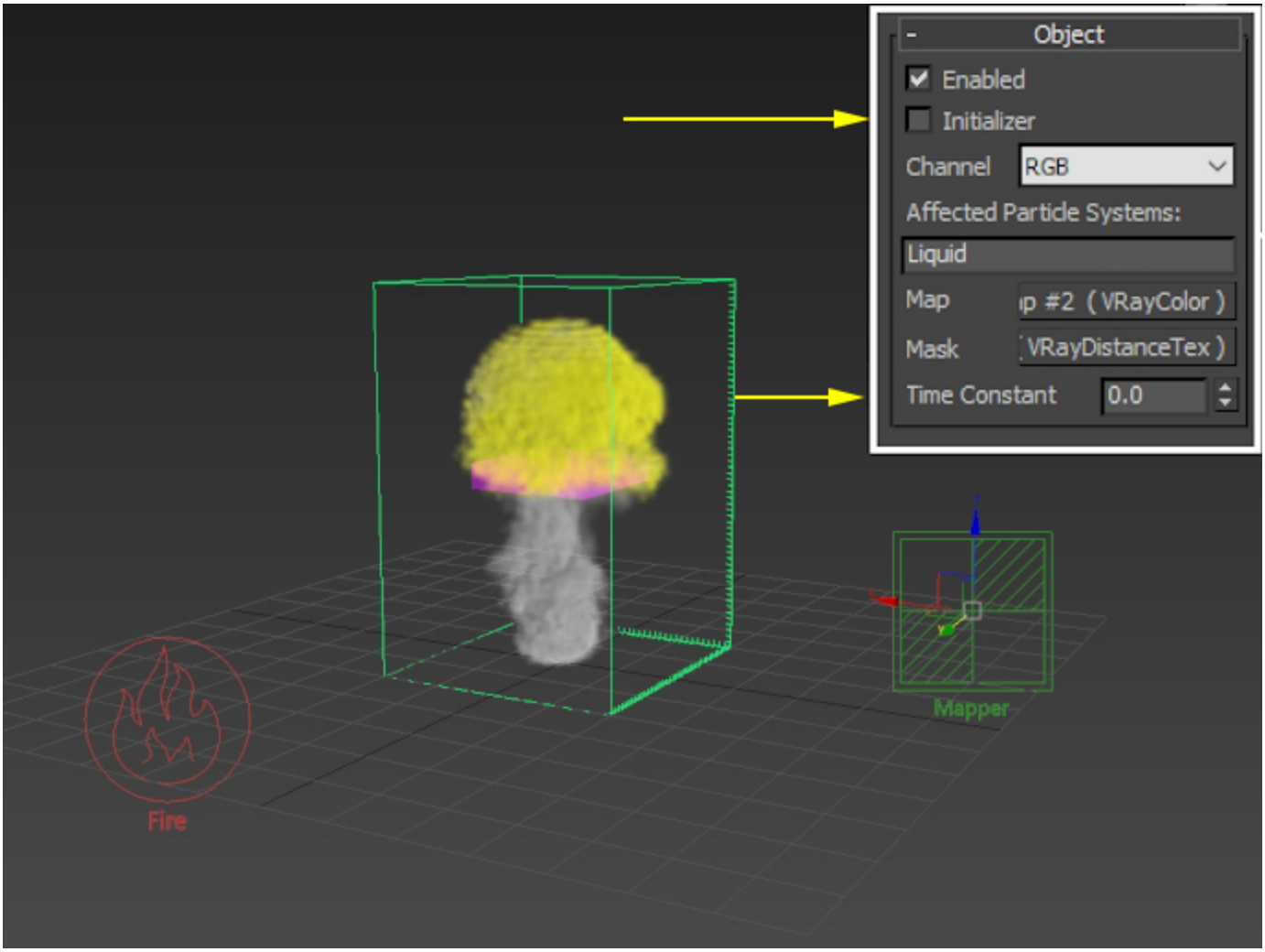
The reason is the Mapper settings. By default, the **Initializer** option is enabled. This means that the Mapper will set the specified channel (RGB in our case) to whatever value is provided by the Map **on the first frame of the simulation** and then **it will become inactive**.

Disable the Initializer option, so the Mapper would affect the entire duration of the simulation.

Lastly, the **Time Constant** parameter at the bottom controls **h ow quickly** the Mapper will set the affected cells of the simulator to the specified Map value. **Time Constant is in seconds** - the default value of 1 means that the Mapper will gradually set the RGB channel for the affected voxels over 1 second.

Set Time Constant to 0 so the effect is instantaneous. If you want the colors to mix on contact instead of being outright overwritten, setting this to 0.1 or so should be a good starting point.

That is all. You can hit **Start** to run the simulation.


Object
Enabled
Initializer
Channel
RGB
Affected Particle Systems:
Liquid
Map
ıp #2 (VRayColor)
Mask
VRayDistanceTex)
Time Constant
0.0
Mapper
Fire