

Porting Shadertoy effects to Qt Quick Effect Maker

You can use effects created in Shadertoy in Qt Quick Effect Maker. When you use Shadertoy

Contents

↓ Using a Shadertoy effect in Qt Quick Effect Maker

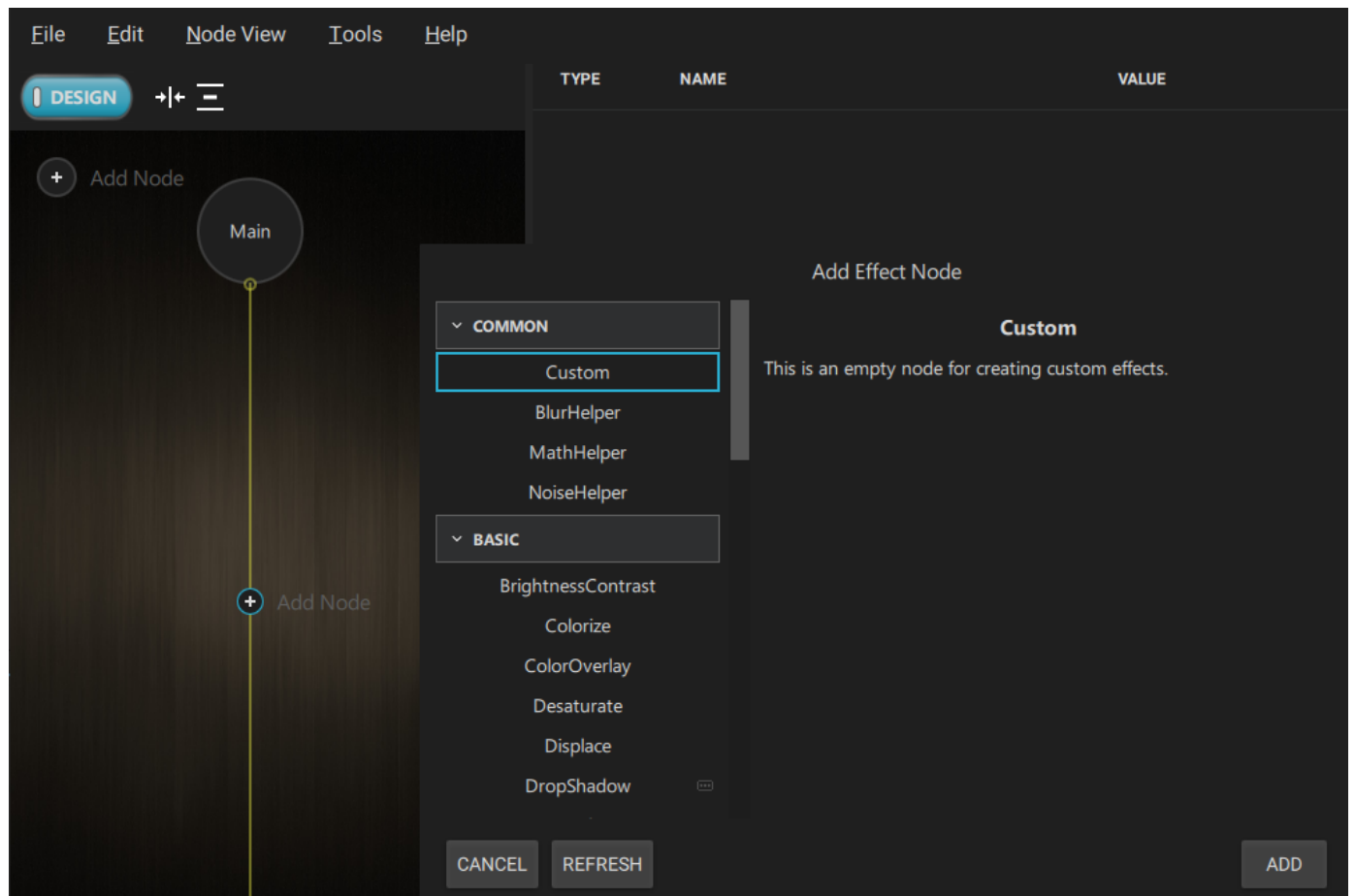
effects, consider the following:

- Qt Quick Effect Maker doesn't support the following Shadertoy features:
 - Multipass effects (Buffer tabs)
 - Audio
 - Cube maps
 - 3D textures
- Shadertoy supports only fragment shaders and built-in textures. To improve the effect performance in Qt Quick Effect Maker, move some calculations to the vertex shader and use custom images to simplify the shader code.
- The coordinate system differs between Shadertoy and Qt Quick Effect Maker. In Shadertoy, the origin (0,0) is located in the lower-left corner while Qt Quick Effect Maker has the origin in the upper-left corner.

Using a Shadertoy effect in Qt Quick Effect Maker

To use a Shadertoy effect in Qt Quick Effect Maker:

1. In Qt Quick Effect Maker, create a new effect.
2. In the node editor, select **Add node** and then, under **Common**, select **Custom**. This creates an empty node.



3. In Shadertoy, copy all the code from the **Image** tab.
4. In Qt Quick Effect Maker, double-click the **Custom** node in the node editor. This opens the code editor.
5. Paste the Shadertoy code to the **Frag** tab.
6. Find the Main function in the code, it looks something like:

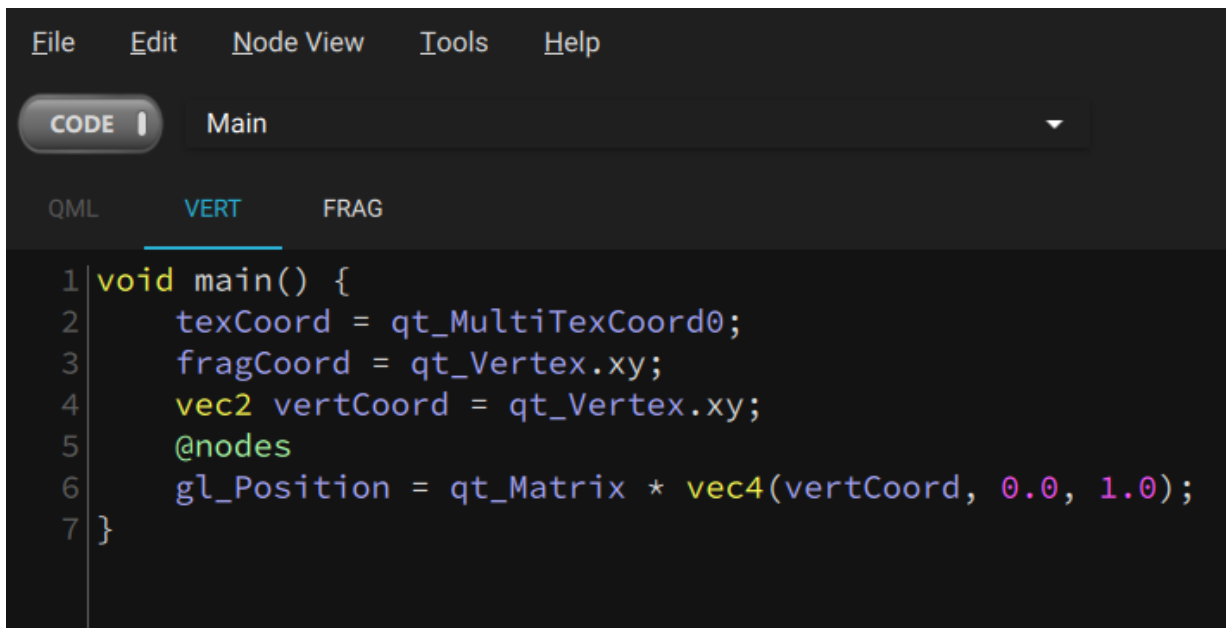
```
void mainImage( out vec4 fragColor, in vec2 fragCoord )
```

7. Replace this line with:

```
@main
```

Note: You can't have @main and the following { on the same line.

8. Optional. If the effect depends on the coordinate system, it appears flipped upside down. To solve this, you need to flip the y-coordinate:
 1. Go to the **Vert** tab.



The screenshot shows the Qt Quick Effect Maker interface. At the top, there is a menu bar with 'File', 'Edit', 'Node View', 'Tools', and 'Help'. Below the menu bar, there is a 'CODE' button and a dropdown menu currently showing 'Main'. Underneath, there are three tabs: 'QML', 'VERT', and 'FRAG'. The 'VERT' tab is selected and highlighted with a blue underline. The code editor displays the following GLSL code:

```
1 void main() {  
2     texCoord = qt_MultiTexCoord0;  
3     fragCoord = qt_Vertex.xy;  
4     vec2 vertCoord = qt_Vertex.xy;  
5     @nodes  
6     gl_Position = qt_Matrix * vec4(vertCoord, 0.0, 1.0);  
7 }
```

2. In the drop-down menu, select **Main**.
3. Find the fragCoord line, it should look something like:

```
fragCoord = qt_Vertex.xy;
```

4. Replace this line with:

```
fragCoord = vec2(qt_Vertex.x, iResolution.y - qt_Vertex.y);
```

5. Similarly, you might need to adjust texCoord and iMouse.

Now, the effect runs and looks the same as the Shadertoy effect.

[Creating a blur effect](#) [Wiggly](#)

© 2022 The Qt Company Ltd. Documentation contributions included herein are the copyrights of their respective owners.

The documentation provided herein is licensed under the terms of the [GNU Free Documentation License version 1.3](#) as published by the Free Software Foundation.

Qt and respective logos are [trademarks](#) of The Qt Company Ltd. in Finland and/or other countries worldwide. All other trademarks are property of their respective owners.