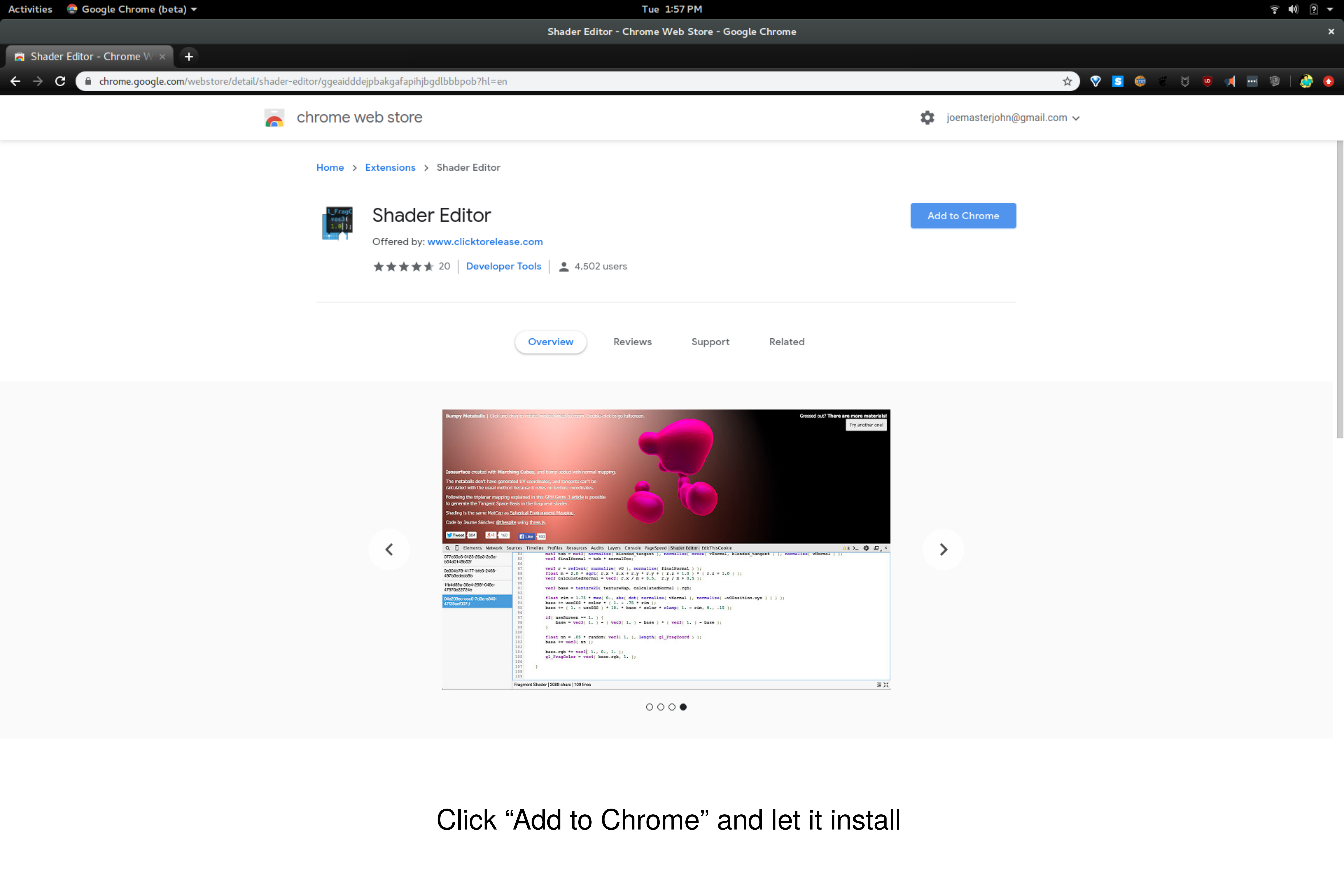
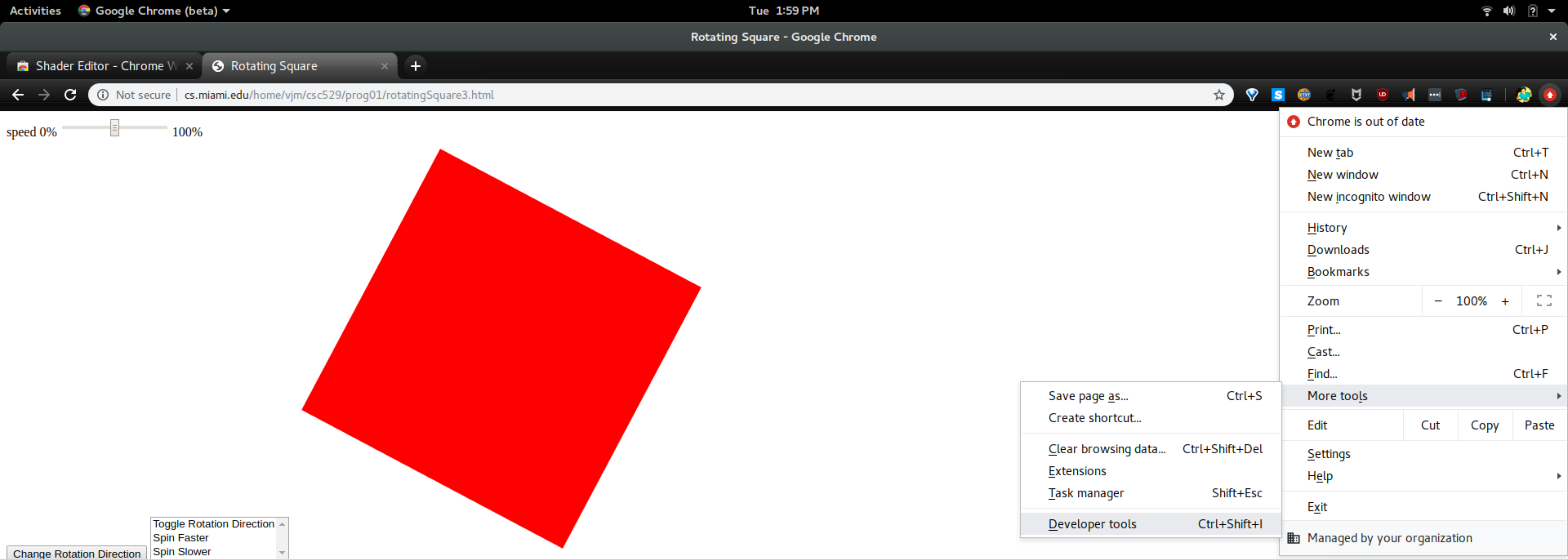


First of all, install the Shader Editor chrome extension. You can find it with a quick Google search.



Click “Add to Chrome” and let it install



Open up the rotatingSquare3.html file, either locally from your Downloaded files or view it online at cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html. Click the top right settings menu, and under “More tools” select “Developer tools”. The shortcut for this menu is Ctrl+Shift+I, at least on Ubuntu. It will list the shortcut for your OS next to the menu option.



Welcome!

WebGL GLSL shader editor extension
v1.0.17 (beta)

To start tracking the WebGL context,
the extension needs to reload the page.

[Reload](#)

Bugs, ideas and feedback: [GitHub page](#)
[@thespите](#) | [www.clicktorelease.com](#)

The Shader Editor extension will show up as an option in the developer tools menu bar. If you click it, it will bring you to the welcome page and inform you to reload the page to view and edit the shaders. Reload the page.


Activities Google Chrome (beta) Tue 2:00 PM

Rotating Square - Google Chrome

Shader Editor - Chrome W Rotating Square

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100%



Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Shaders Textures Settings

Program 1

```
1
2 attribute vec4 vPosition;
3 uniform float theta;
4
5 void
6 main()
7 {
8     float s = sin( theta );
9     float c = cos( theta );
10
11     gl_Position.x = -s * vPosition.x + c * vPosition.y;
12     gl_Position.y = s * vPosition.y + c * vPosition.x;
13     gl_Position.z = 0.0;
14     gl_Position.w = 1.0;
15 }
16
```

Vertex Shader | 284 chars | 16 lines

```
1
2
3 precision mediump float;
4
5 void
6 main()
7 {
8     gl_FragColor = vec4( 1.0, 0.0, 0.0, 1.0 );
9 }
10
```

Fragment Shader | 91 chars | 10 lines


On the left you will see a list of all shader programs loaded in this context. In our case there is only one program, “Program 1”. In the first editor you can see the vertex shader for this program.

Activities Google Chrome (beta) Tue 2:00 PM Rotating Square - Google Chrome

Shader Editor - Chrome W Rotating Square

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100%



Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Shaders Textures Settings

Program 1

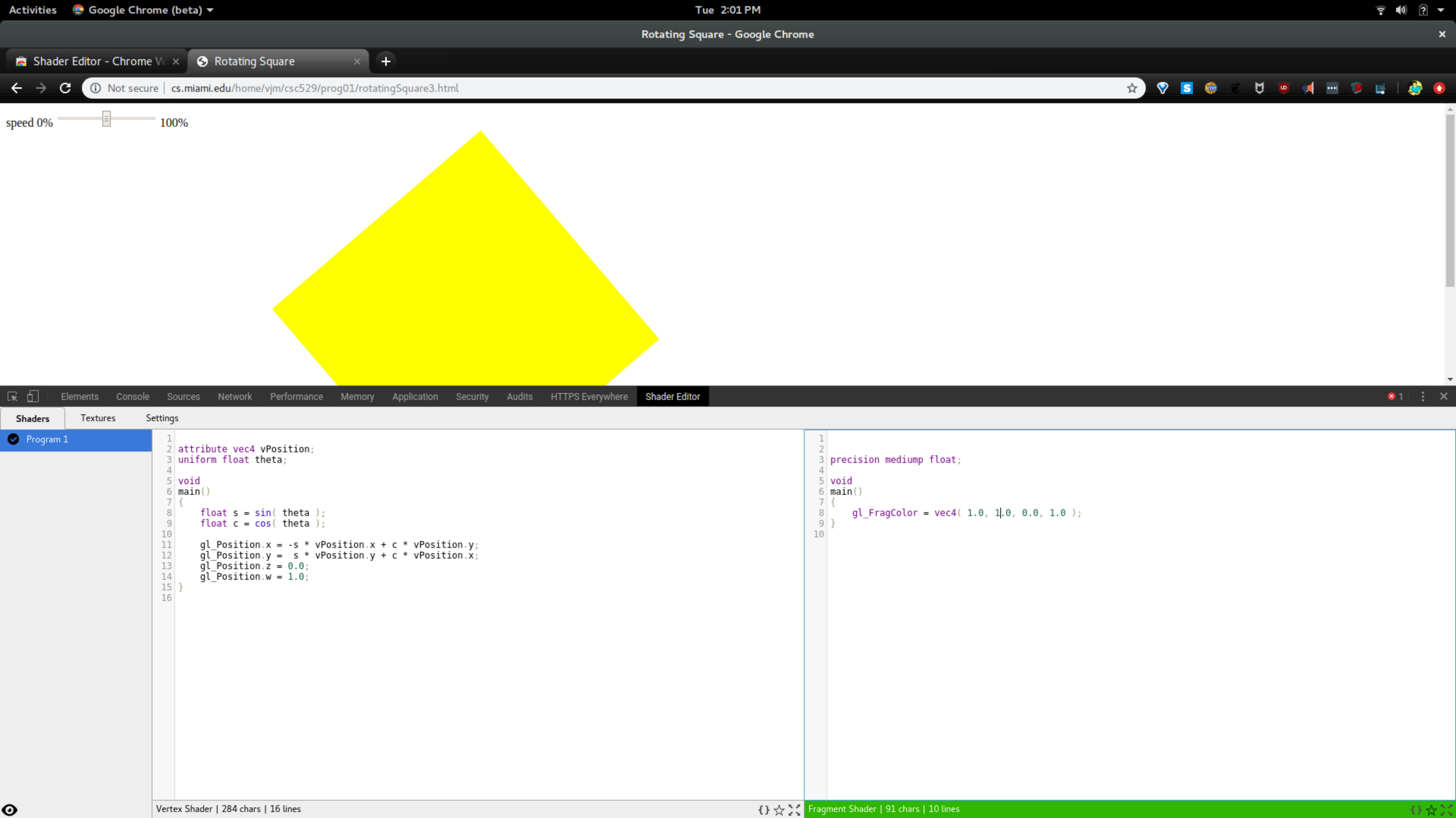
```
1
2 attribute vec4 vPosition;
3 uniform float theta;
4
5 void
6 main()
7 {
8     float s = sin( theta );
9     float c = cos( theta );
10
11     gl_Position.x = -s * vPosition.x + c * vPosition.y;
12     gl_Position.y = s * vPosition.y + c * vPosition.x;
13     gl_Position.z = 0.0;
14     gl_Position.w = 1.0;
15 }
16
```

Vertex Shader | 284 chars | 16 lines

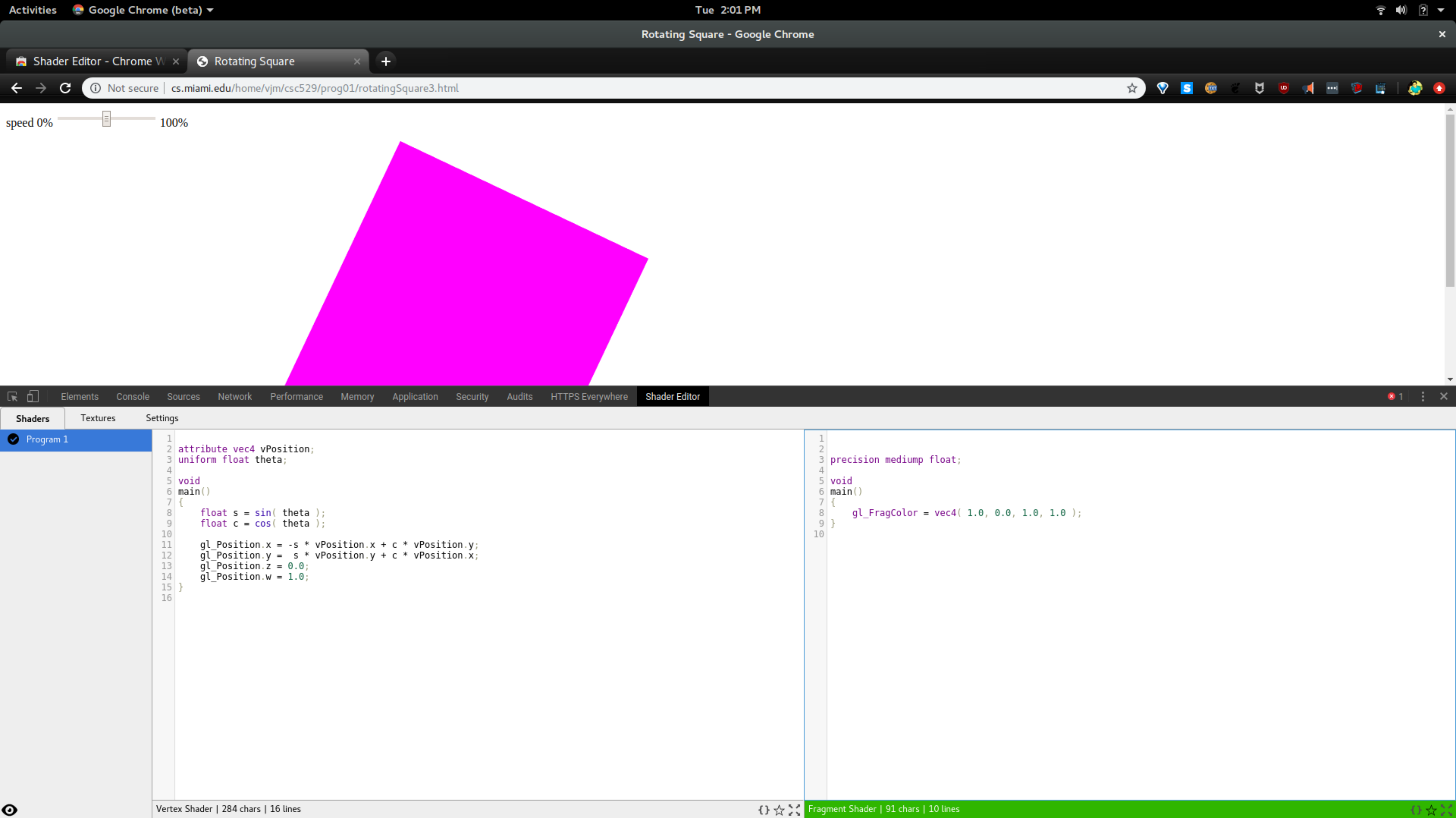
```
1
2
3 precision mediump float;
4
5 void
6 main()
7 {
8     gl_FragColor = vec4( 1.0, 0.0, 0.0, 1.0 );
9 }
10
```

Fragment Shader | 91 chars | 10 lines

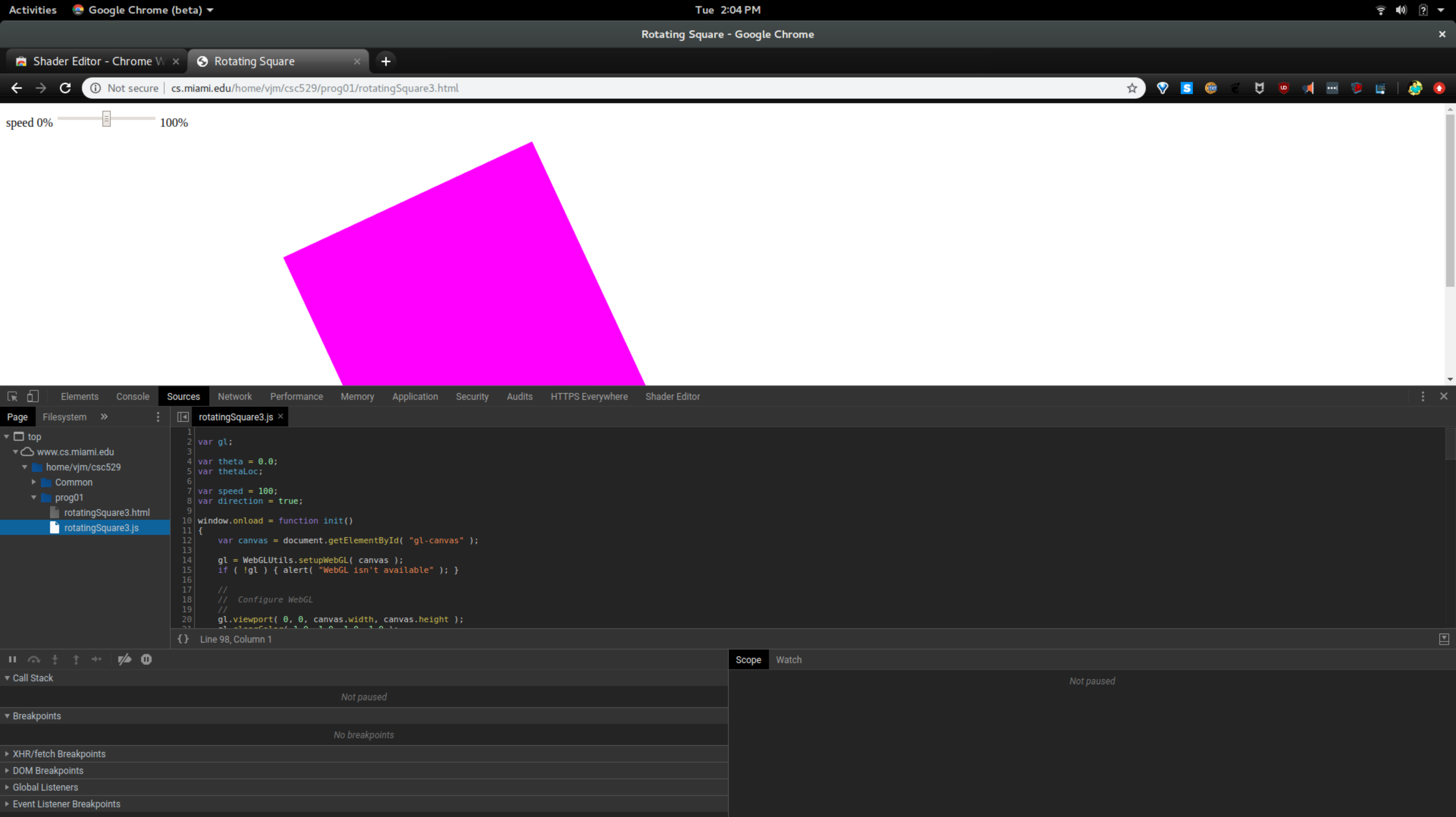
On the right is the fragment shader for this program.



Making changes to the shaders causes the program to recompile and run with your changes automatically after detecting a change in the editor. If you input code that does not compile, the editor will tell you which line is the first offending line and the program will not run. Here I've changed the value of `gl_FragColor`.



Here I've changed the value of `gl_FragColor` to make the square magenta.



The “sources” tab is where you can see all of the javascript and html that has been requested by the page. Here we see the main program that is rendering the square, “rotatingSquare3.js”.

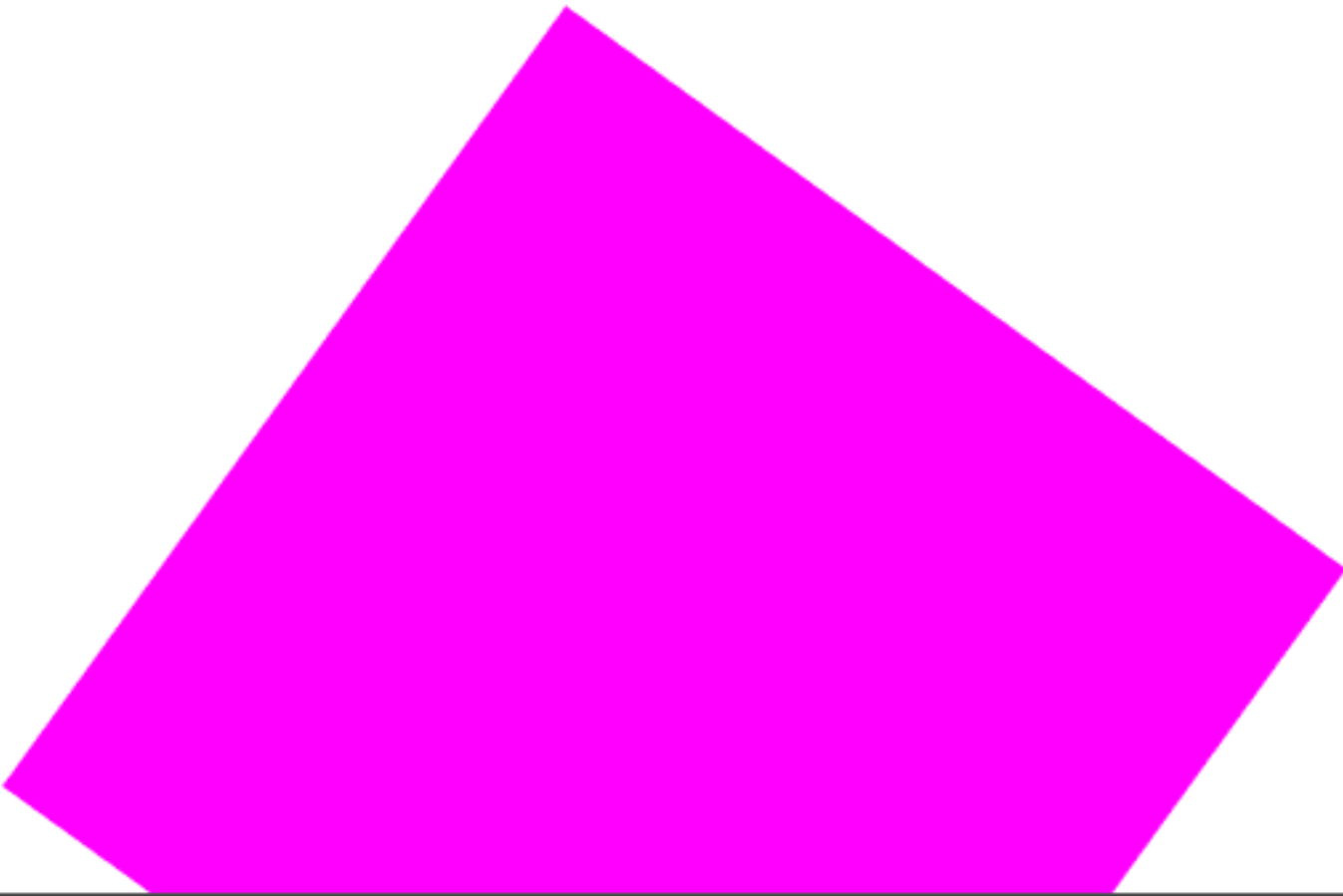
ActivitiesGoogle Chrome (beta)Tue 2:08 PM

Rotating Square - Google Chrome

Shader Editor - Chrome W xRotating Square x+

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0%100%



ElementsConsoleSourcesNetworkPerformanceMemoryApplicationSecurityAuditsHTTPS EverywhereShader Editor

PageFilesystem>>rotatingSquare3.js x

topwww.cs.miami.eduhome/vjm/csc529Commonprog01rotatingSquare3.htmlrotatingSquare3.js

```
1
2 var gl;
3
4 var theta = 0.0;
5 var thetaLoc;
6
7 var speed = 100;
8 var direction = true;
9
10 window.onload = function init()
11 {
12     var canvas = document.getElementById( "gl-canvas" );
13
14     gl = WebGLUtils.setupWebGL( canvas );
15     if ( !gl ) { alert( "WebGL isn't available" ); }
16
17     //
18     // Configure WebGL
19     //
20     gl.viewport( 0, 0, canvas.width, canvas.height );
```

{ } Line 16, Column 1

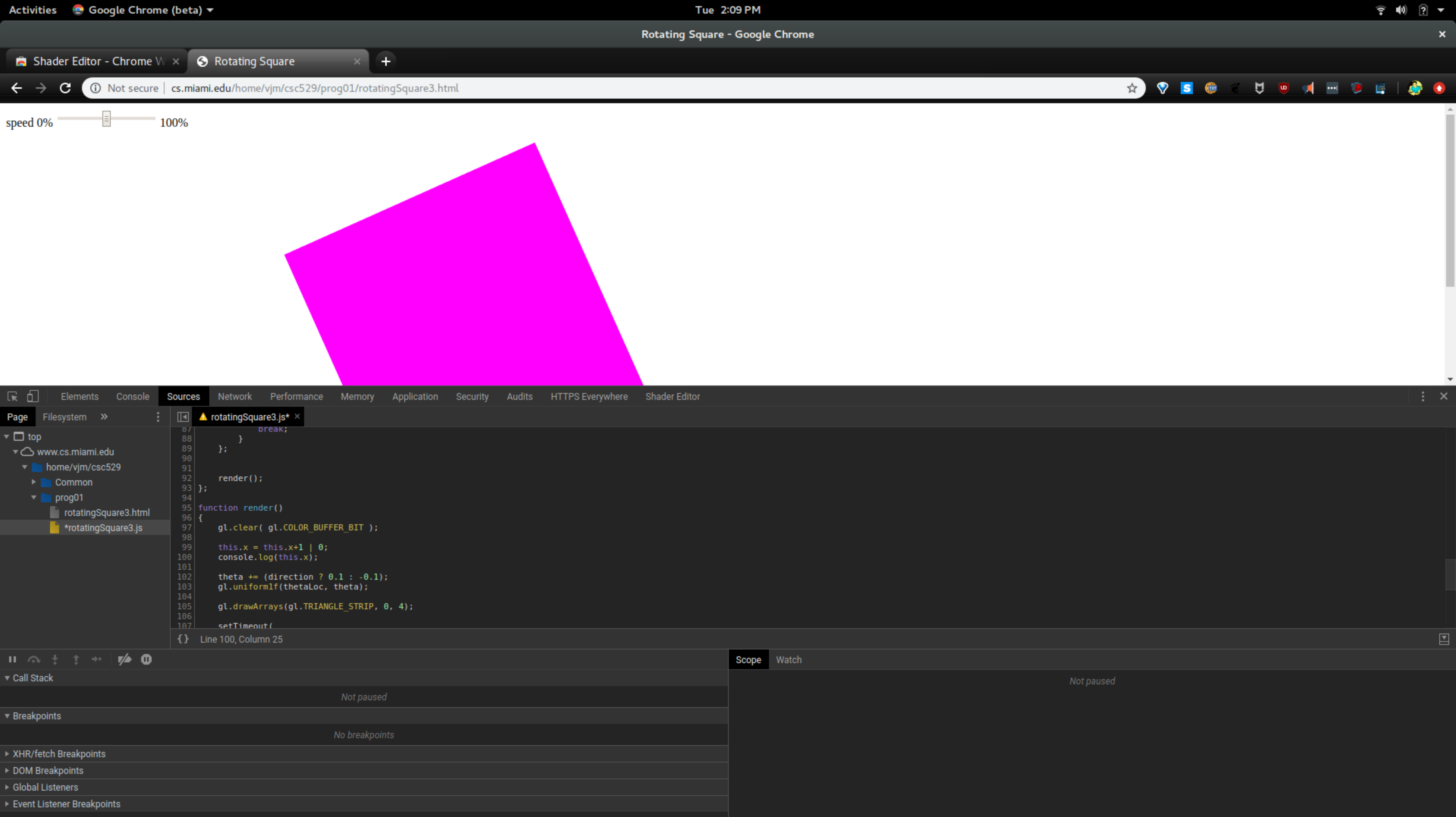
ScopeWatch

Not paused

No breakpoints

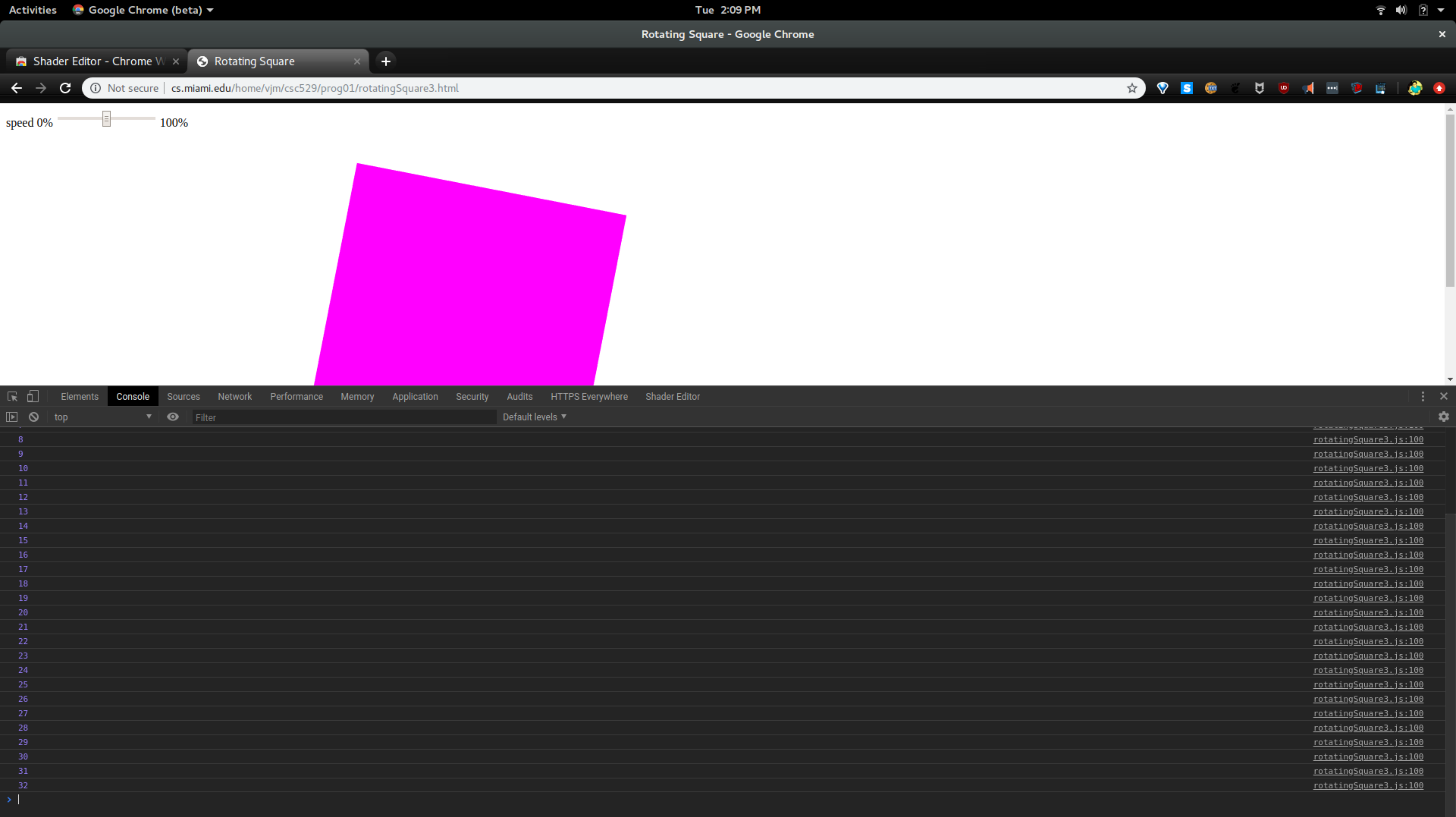
XHR/fetch BreakpointsDOM BreakpointsGlobal ListenersEvent Listener Breakpoints

This javascript editor is also editable, however you need to hit Ctrl+S to commit the changes you’ve made to the file. CAUTION: do not use this to make permanent edits to your program. Changes are not saved to file and will be lost from a number of different contexts, including reloading the page. The file tab will have a yellow caution sign after making changes.



Here I’ve edited the `render()` function to print a variable that increases every tick to the screen. `console.log()` will print output to the Console tab. “this” has a very different meaning in Javascript than its meaning in Java or C++. What is “this” referring to in the context of this call to `render`?

If you don’t know: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/this>



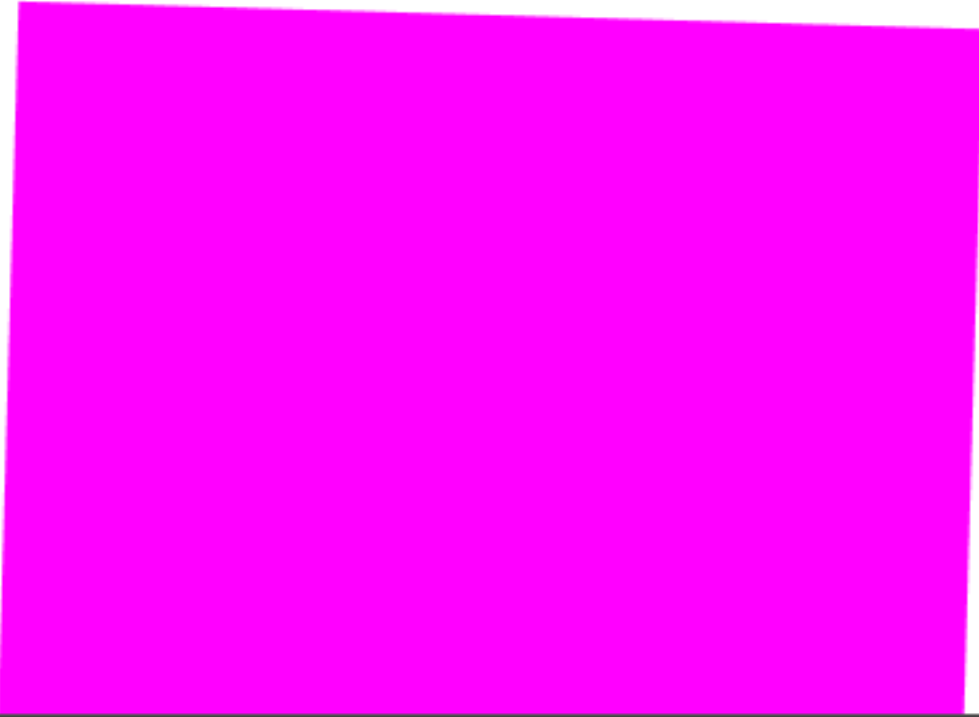
The console tab showing the output of our change.

Activities Google Chrome (beta) Tue 2:09 PM Rotating Square - Google Chrome

Shader Editor - Chrome W Rotating Square

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100%



Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Page Filesystem >> rotatingSquare3.js

- top
- www.cs.miami.edu
 - home/vjm/csc529
 - Common
 - prog01
 - rotatingSquare3.html
 - rotatingSquare3.js

```
1
2 var gl;
3
4 var theta = 0.0;
5 var thetaLoc;
6
7 var speed = 100;
8 var direction = true;
9
10 window.onload = function init()
11 {
12     var canvas = document.getElementById( "gl-canvas" );
13
14     this.x = this.x+1 | 0;
15     console.log(this.x);|
16
17     gl = WebGLUtils.setupWebGL( canvas );
18     if ( !gl ) { alert( "WebGL isn't available" ); }
19
20     //
```

{ } Line 15, Column 25

Scope Watch

Call Stack Not paused

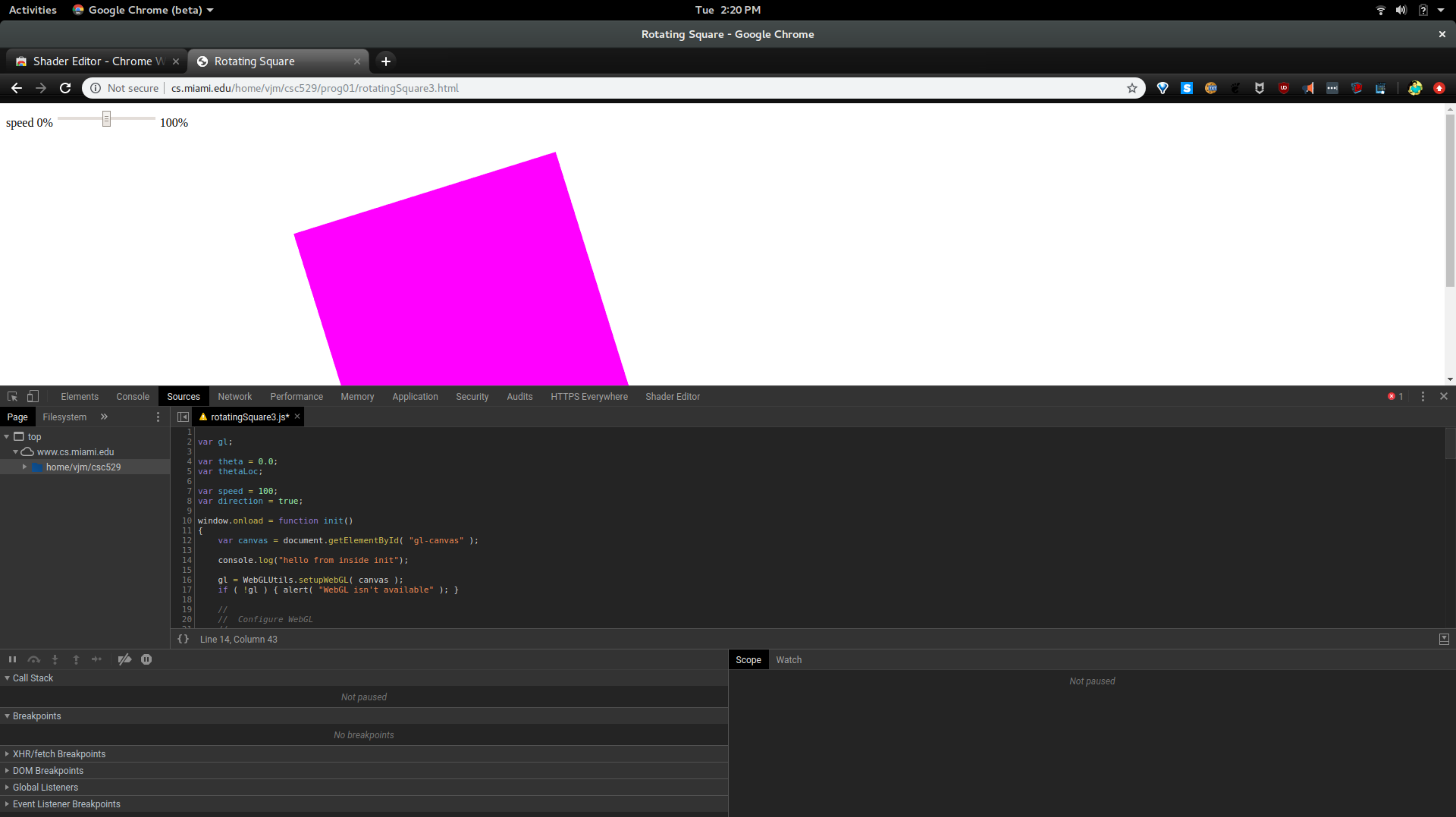
Breakpoints No breakpoints

- XHR/fetch Breakpoints
- DOM Breakpoints
- Global Listeners
- Event Listener Breakpoints

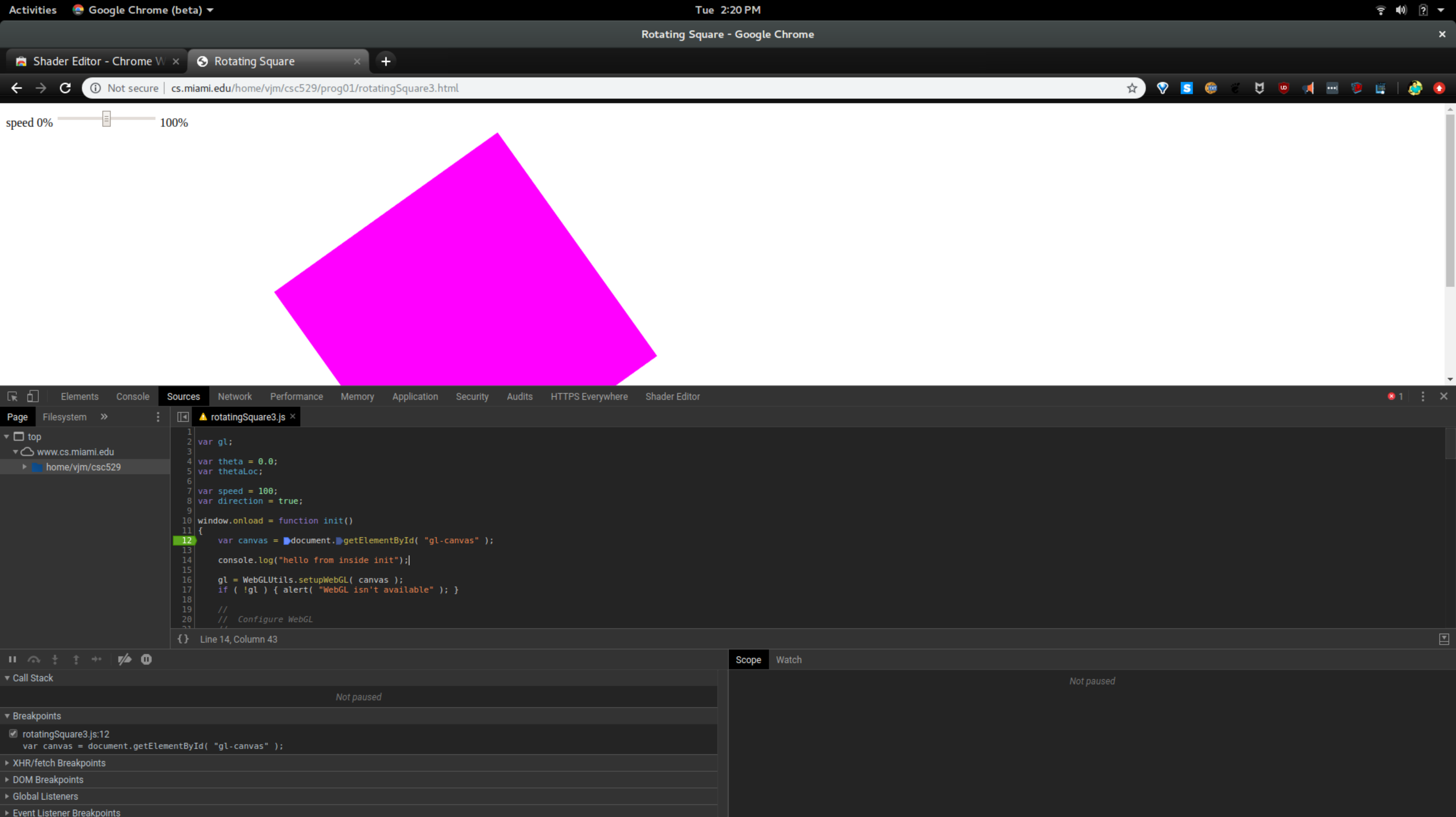
Here we'll make the same edit to the window.onload function.



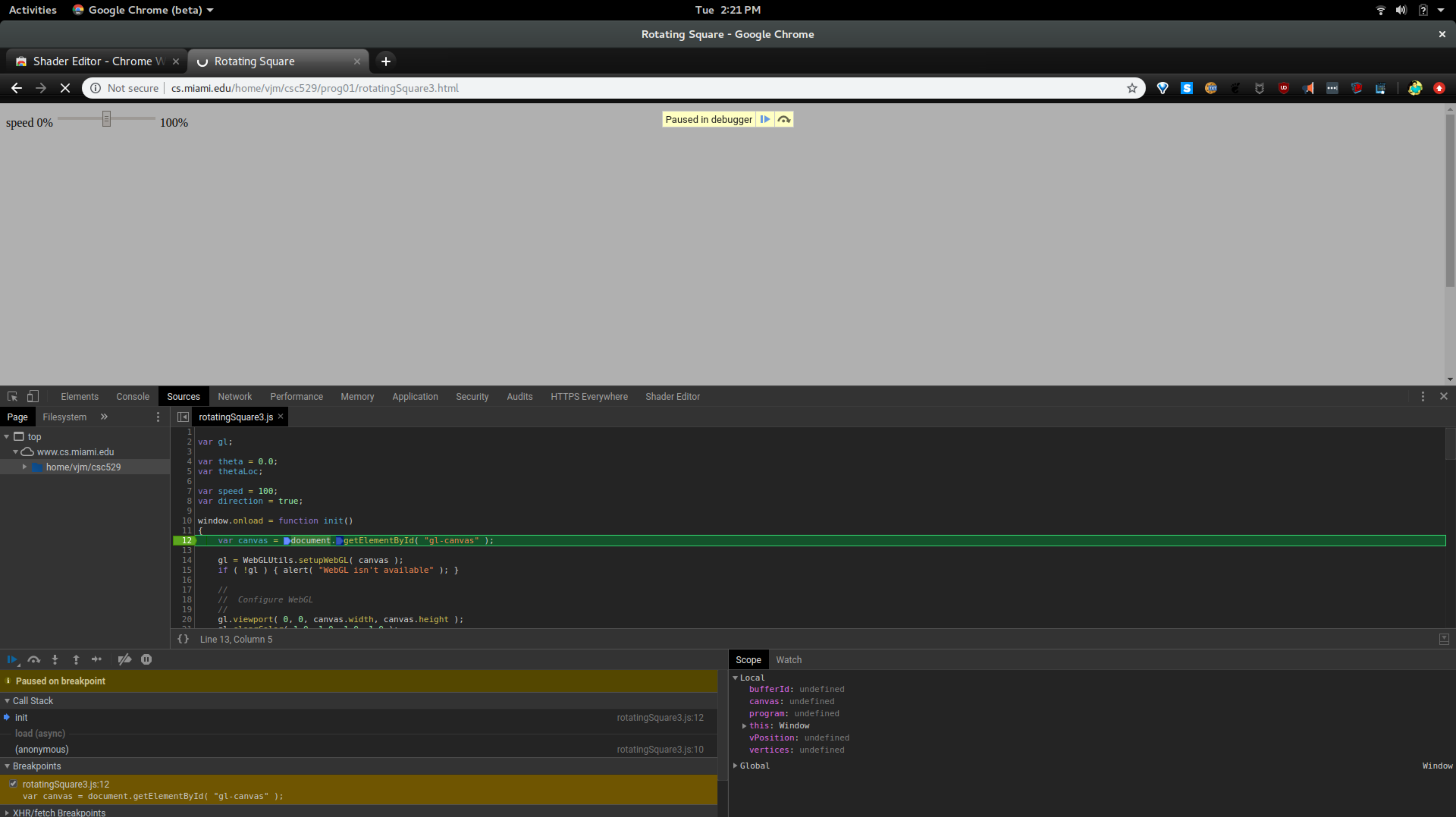
What's this?? No output to the console. That's because we've modified a function that has already been called, and will not be called again. `window.onload` gets called when every DOM element requested by the document has finished loading. You won't be able to get to the editor and make a change before that happens, so how can we live edit such a method?



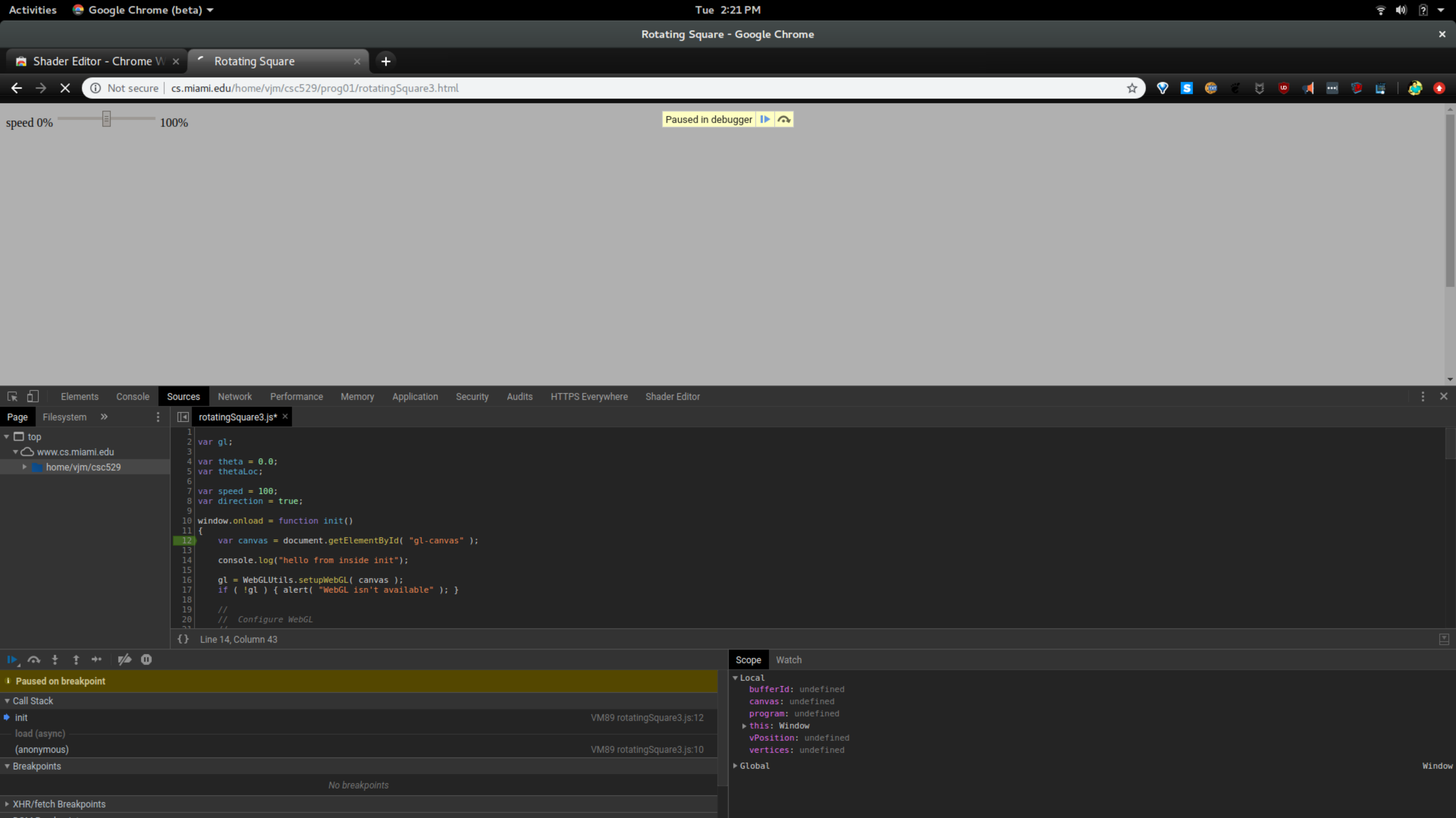
Here in our editor is where we want to make a change to the window.onload function. We've added the console.log.



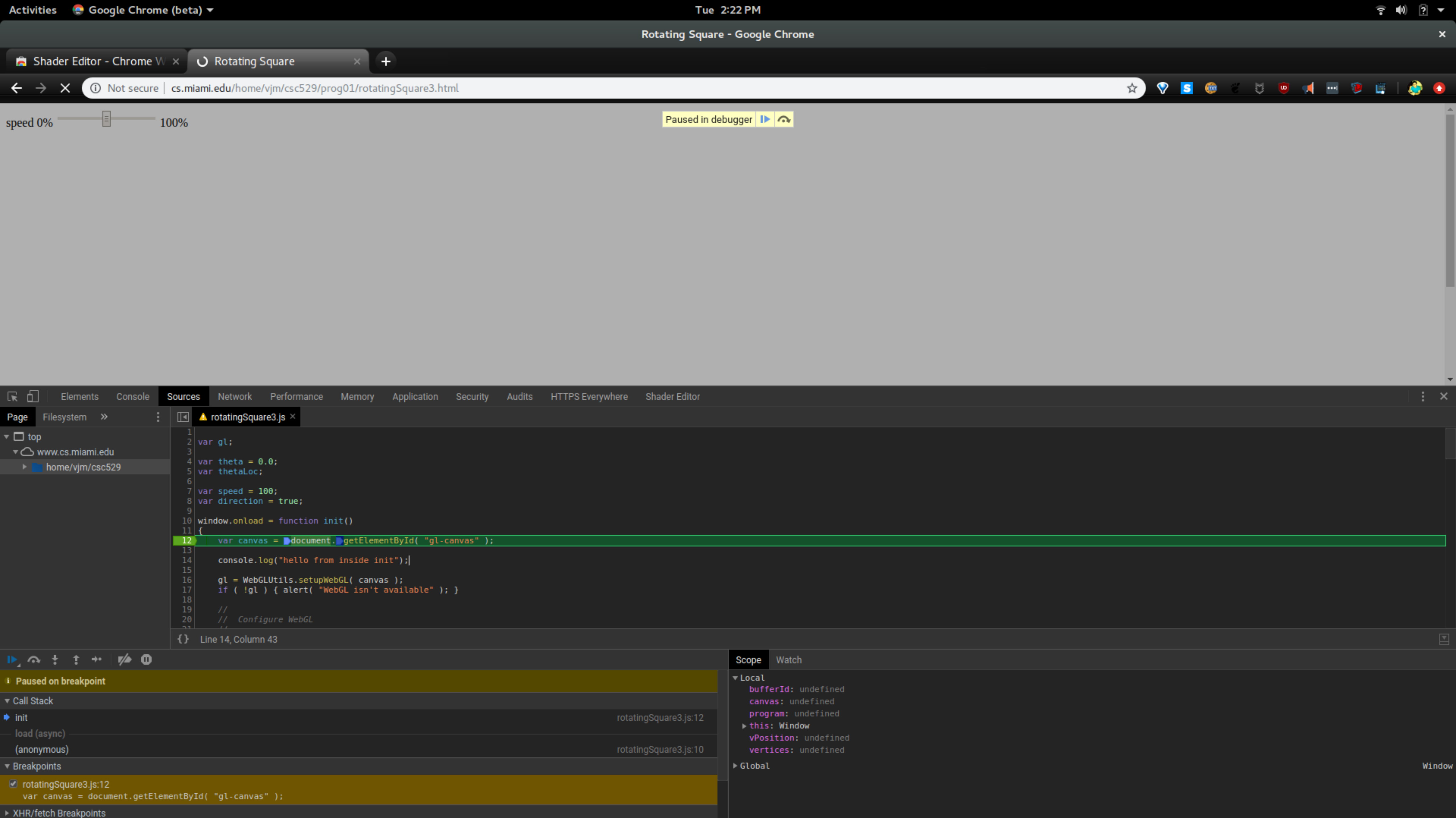
We can add a breakpoint to the code by clicking on the line number next to the line we want to break on. In order to break on this particular line in the window.onload function, we will have to reload the page, because this code has already executed. Hit refresh or F5, breakpoints are saved between refreshes.



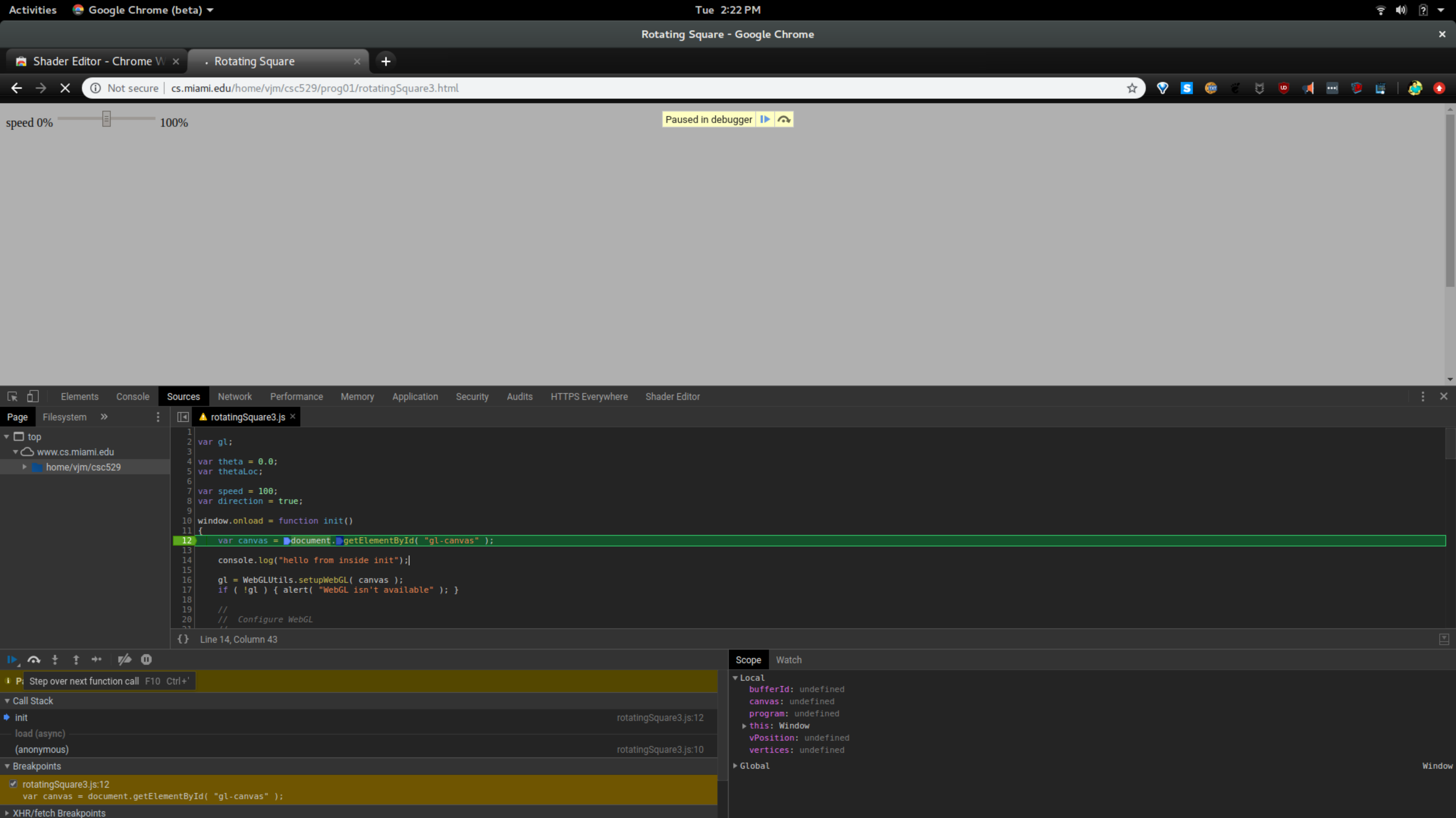
Here we can see the page has been paused by the debugger and the line we broke on is highlighted. BUT, the edit we made to the code is gone. Changes to the javascript code in the editor are not preserved between refreshes, be careful. We'll have to add it now while the program is paused.



Here we've added the console.log while the program is paused. Hit Ctrl+S to commit the change.



When we’ve committed a change with Ctrl+S, the page will reload and run up to the first breakpoint. This time though, changes to the code are preserved in the session. (AGAIN these changes are never saved to file. Only use this for small changes and things you can reproduce)



On the left are the set of debugger control buttons. The second button in the row is to step over the next function call. All of these commands also have keyboard shortcuts listed in their hover over text.

Activities Google Chrome (beta) Tue 2:22 PM Rotating Square - Google Chrome

Shader Editor - Chrome W x Rotating Square x +

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100% Paused in debugger

Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Page Filesystem >> rotatingSquare3.js x

```
1
2 var gl;
3
4 var theta = 0.0;
5 var thetaLoc;
6
7 var speed = 100;
8 var direction = true;
9
10 window.onload = function init()
11 {
12   var canvas = document.getElementById( "gl-canvas" ); canvas = canvas#gl-canvas {width: 512, height: 512, title: "", lang: "", translate: true, ...}
13
14   console.log("hello from inside init");
15
16   gl = WebGLUtils.setupWebGL( canvas );
17   if ( !gl ) { alert( "WebGL isn't available" ); }
18
19   //
20   // Configure WebGL
21   //
```

{ } Line 14, Column 5

Debugger paused

Call Stack

- init rotatingSquare3.js:14
- load (async)
- (anonymous) rotatingSquare3.js:10

Breakpoints

- rotatingSquare3.js:12 var canvas = document.getElementById("gl-canvas");
- XHR/fetch Breakpoints

Scope Watch

Local

- bufferId: undefined
- ▶ canvas: canvas#gl-canvas
- program: undefined
- ▶ this: Window
- vPosition: undefined
- vertices: undefined

Global

Window

After stepping over line twelve we are stopped on line 14.

Activities Google Chrome (beta) Tue 2:22 PM Rotating Square - Google Chrome

Shader Editor - Chrome W Rotating Square

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100% Paused in debugger

Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Page Filesystem >> rotatingSquare3.js

```
1
2 var gl;
3
4 var theta = 0.0;
5 var thetaLoc;
6
7 var speed = 100;
8 var direction = true;
9
10 window.onload = function init()
11 {
12   var canvas = document.getElementById( "gl-canvas" );
13   canvas = canvas#gl-canvas {width: 512, height: 512, title: "", lang: "", translate: true, ...}
14   console.log("hello from inside init");
15
16   gl = WebGLUtils.setupWebGL( canvas );
17   if ( !gl ) { alert( "WebGL isn't available" ); }
18
19   //
20   // Configure WebGL
21   //
```

{ } Line 16, Column 5

Debugger paused

Call Stack

- init rotatingSquare3.js:16
- load (async)
- (anonymous) rotatingSquare3.js:10

Breakpoints

- rotatingSquare3.js:12
var canvas = document.getElementById("gl-canvas");
- XHR/fetch Breakpoints

Scope Watch

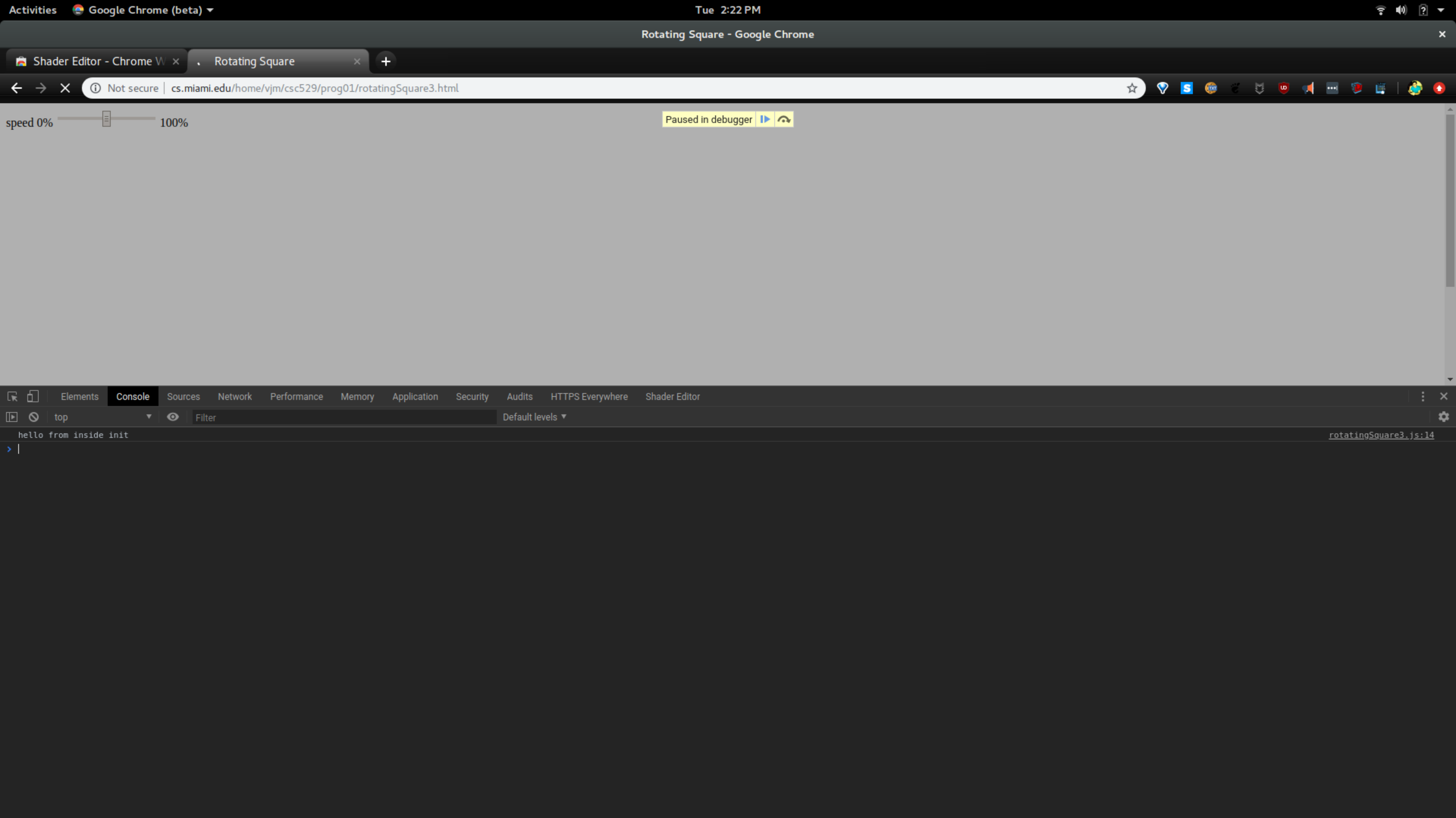
Local

- bufferId: undefined
- ▶ canvas: canvas#gl-canvas
- program: undefined
- ▶ this: Window
- vPosition: undefined
- vertices: undefined

Global

Window

After stepping over line 14 we've stopped on line 16. Let's go to the console to check if our console.log printed anything out.



Our console.log statement successfully printed to the console tab.

Activities Google Chrome (beta) Tue 2:22 PM Rotating Square - Google Chrome

Shader Editor - Chrome W Rotating Square

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100% Paused in debugger

Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Page Filesystem >> rotatingSquare3.js

```
1
2 var gl;
3
4 var theta = 0.0;
5 var thetaLoc;
6
7 var speed = 100;
8 var direction = true;
9
10 window.onload = function init()
11 {
12   var canvas = document.getElementById( "gl-canvas" );
13   console.log("hello from inside init");
14
15   gl = WebGLUtils.setupWebGL( canvas );
16   if ( !gl ) { alert( "WebGL isn't available" ); }
17
18   //
19   // Configure WebGL
20   //
```

Line 16, Column 5

Resume script execution F8 Ctrl+\

Call Stack

- init rotatingSquare3.js:16
- load (async)
- (anonymous) rotatingSquare3.js:10

Breakpoints

- rotatingSquare3.js:12
var canvas = document.getElementById("gl-canvas");

XHR/fetch Breakpoints

Scope Watch

Local

- bufferId: undefined
- ▶ canvas: canvas#gl-canvas
- program: undefined
- ▶ this: Window
- vPosition: undefined
- vertices: undefined

Global

Window

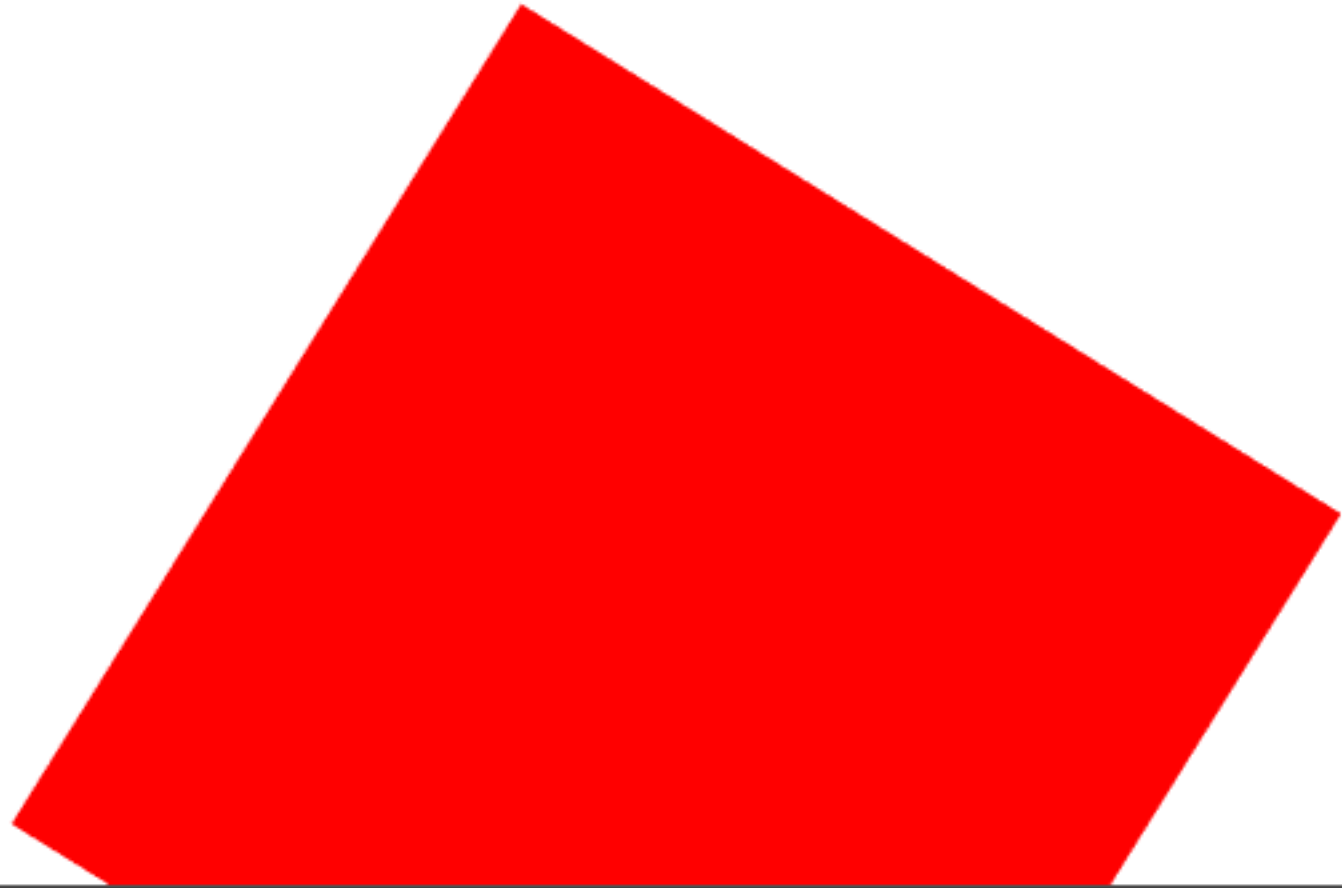
Let's let the script resume execution from here. The first debugger command button in the row switches between pause and play. Hit it to resume the execution of the scripts on the page (up to the next breakpoint if any exist).

Activities Google Chrome (beta) Tue 2:22 PM Rotating Square - Google Chrome

Shader Editor - Chrome W Rotating Square

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100%



Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Page Filesystem >> rotatingSquare3.js

```
1
2 var gl;
3
4 var theta = 0.0;
5 var thetaLoc;
6
7 var speed = 100;
8 var direction = true;
9
10 window.onload = function init()
11 {
12   var canvas = document.getElementById( "gl-canvas" );
13
14   console.log("hello from inside init");
15
16   gl = WebGLUtils.setupWebGL( canvas );
17   if ( !gl ) { alert( "WebGL isn't available" ); }
18
19   //
20   // Configure WebGL
21 }
```

Line 16, Column 5

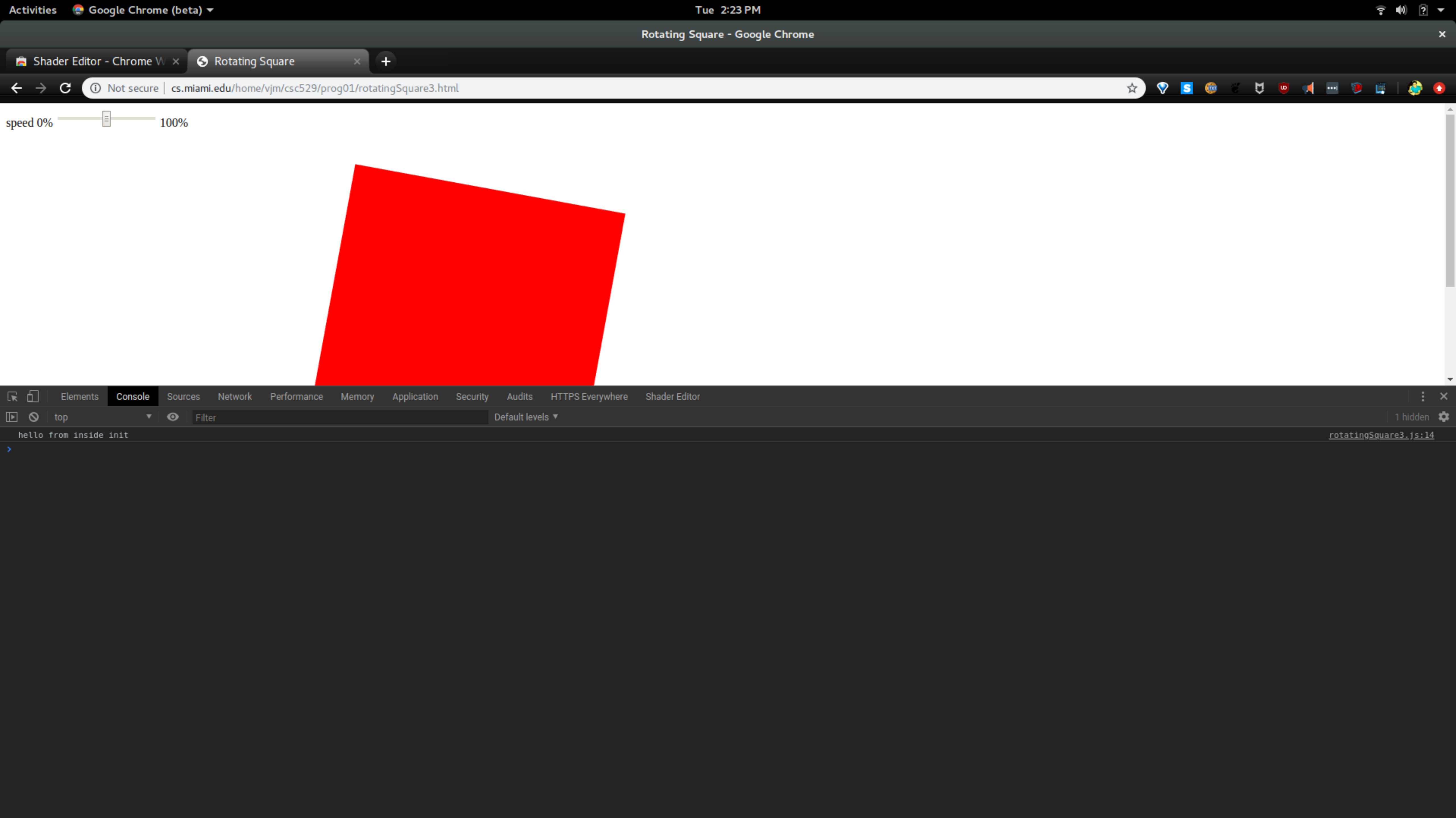
Scope Watch

Call Stack Not paused

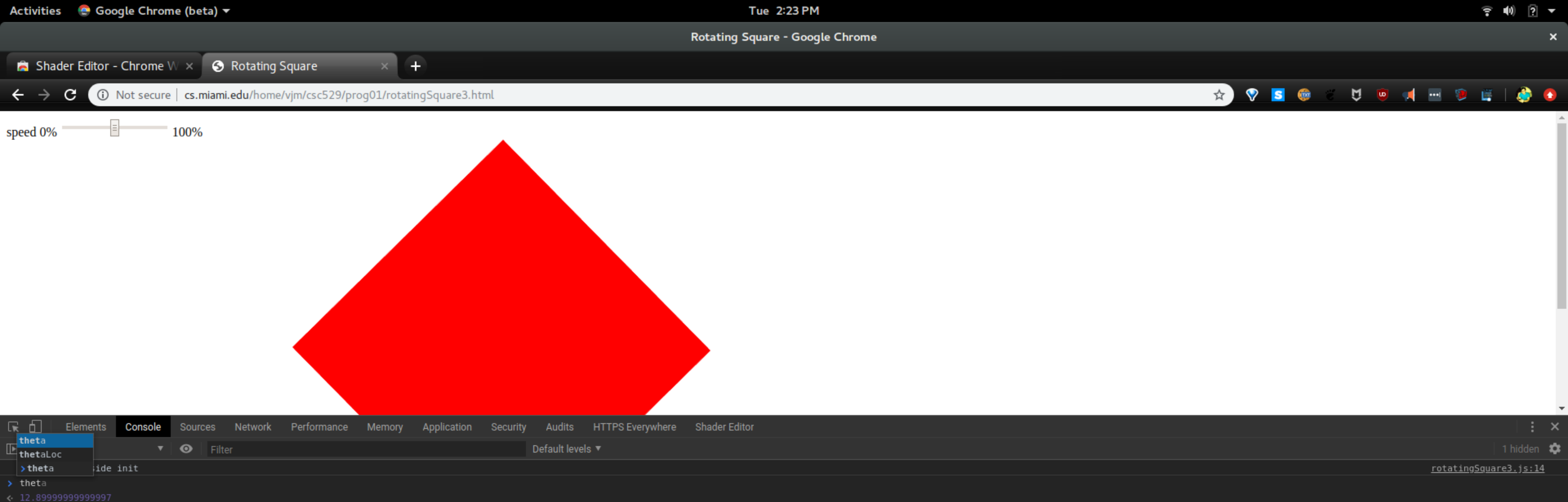
Breakpoints

- rotatingSquare3.js:12
var canvas = document.getElementById("gl-canvas");
- XHR/fetch Breakpoints
- DOM Breakpoints
- Global Listeners
- Event Listener Breakpoints

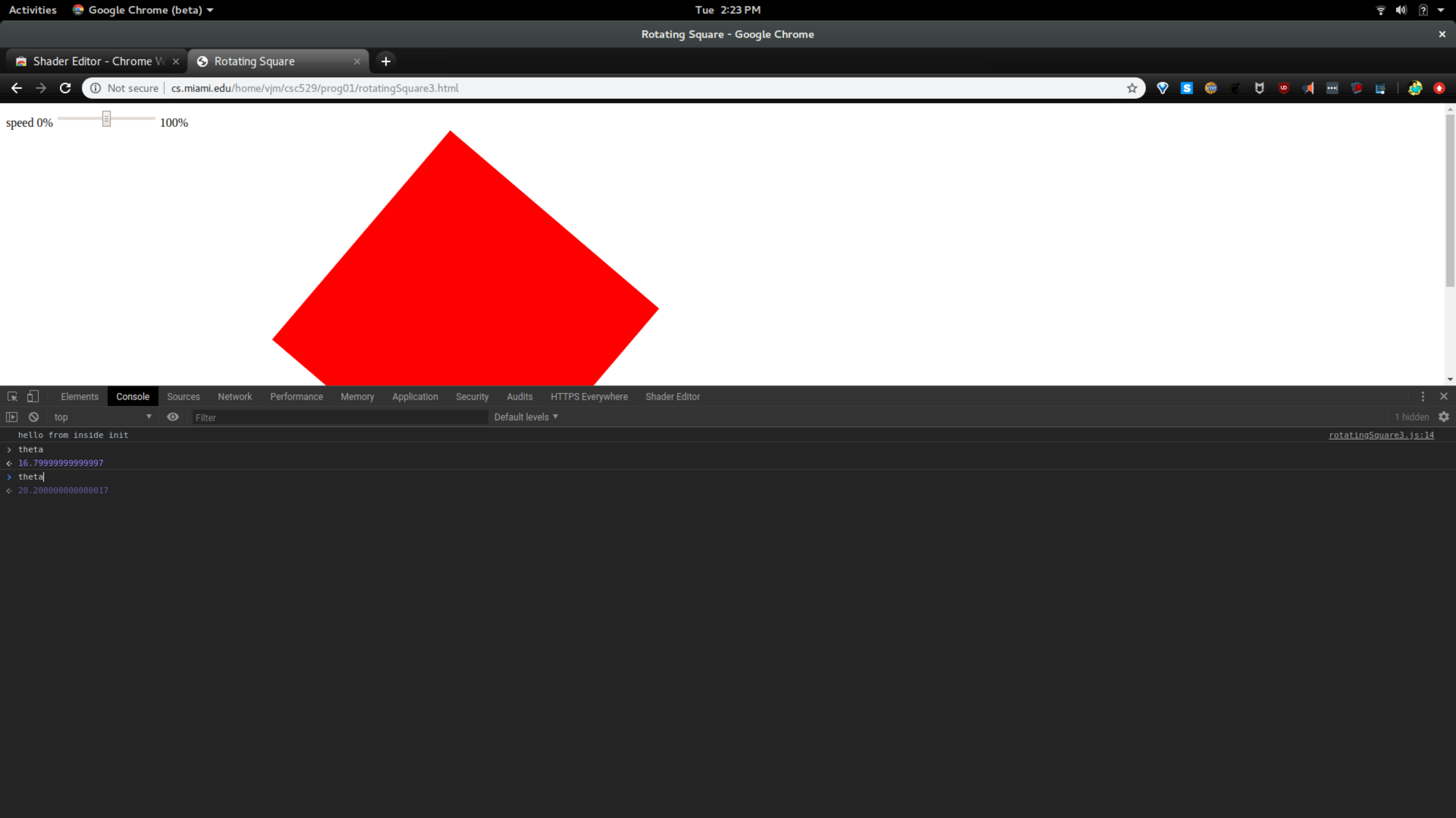
The square should now appear and start to rotate as before.



As well as being the place where print statements are logged, the console also acts as a REPL for the global javascript context of the page.



Here I am accessing a variable “theta” that is declared in the global scope. The console will try to autocomplete things for you.



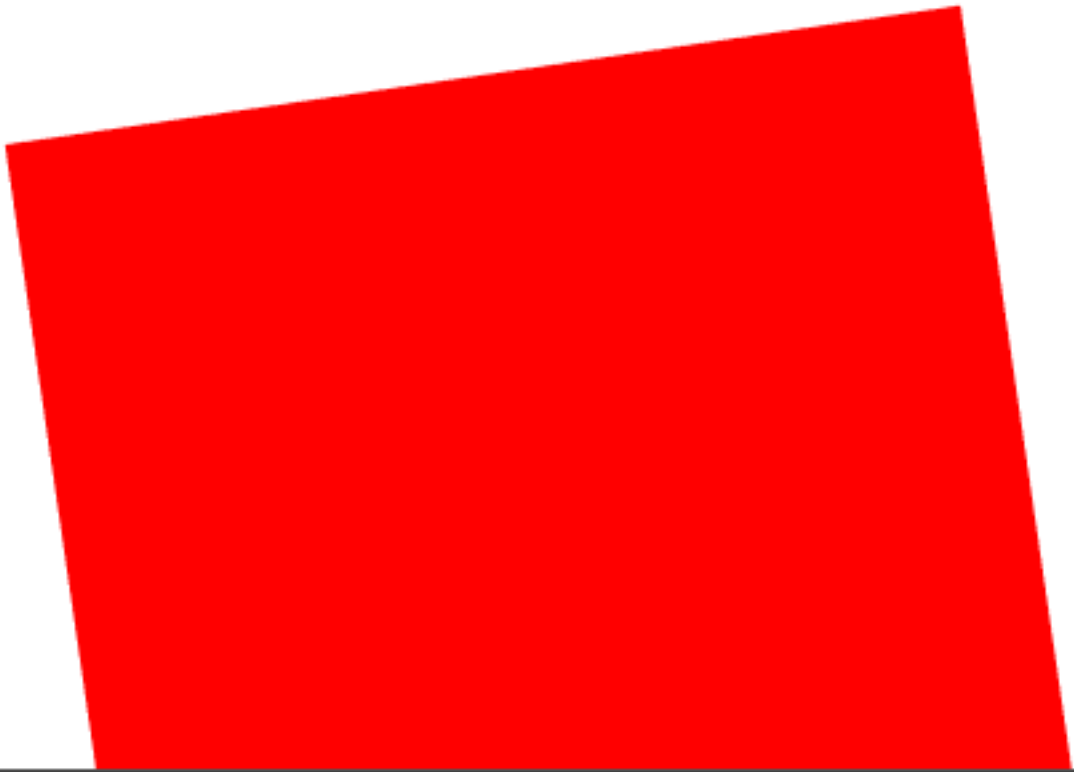
The console prints out the value of “theta” at the time I hit enter. Here I’m typing “theta” again to confirm that its value is changing between frames.

Activities Google Chrome (beta) Tue 2:23 PM Rotating Square - Google Chrome

Shader Editor - Chrome W Rotating Square

Not secure | cs.miami.edu/home/vjm/csc529/prog01/rotatingSquare3.html

speed 0% 100%



Elements Console Sources Network Performance Memory Application Security Audits HTTPS Everywhere Shader Editor

Page Filesystem >> rotatingSquare3.js

```
1
2 var gl;
3
4 var theta = 0.0;
5 var thetaLoc;
6
7 var speed = 100;
8 var direction = true;
9
10 window.onload = function init()
11 {
12     var canvas = document.getElementById( "gl-canvas" );
13
14     console.log("hello from inside init");
15
16     gl = WebGLUtils.setupWebGL( canvas );
17     if ( !gl ) { alert( "WebGL isn't available" ); }
18
19     //
20     // Configure WebGL
21 }
```

2 lines, 22 characters selected

Call Stack Not paused

Breakpoints No breakpoints

XHR/fetch Breakpoints

DOM Breakpoints

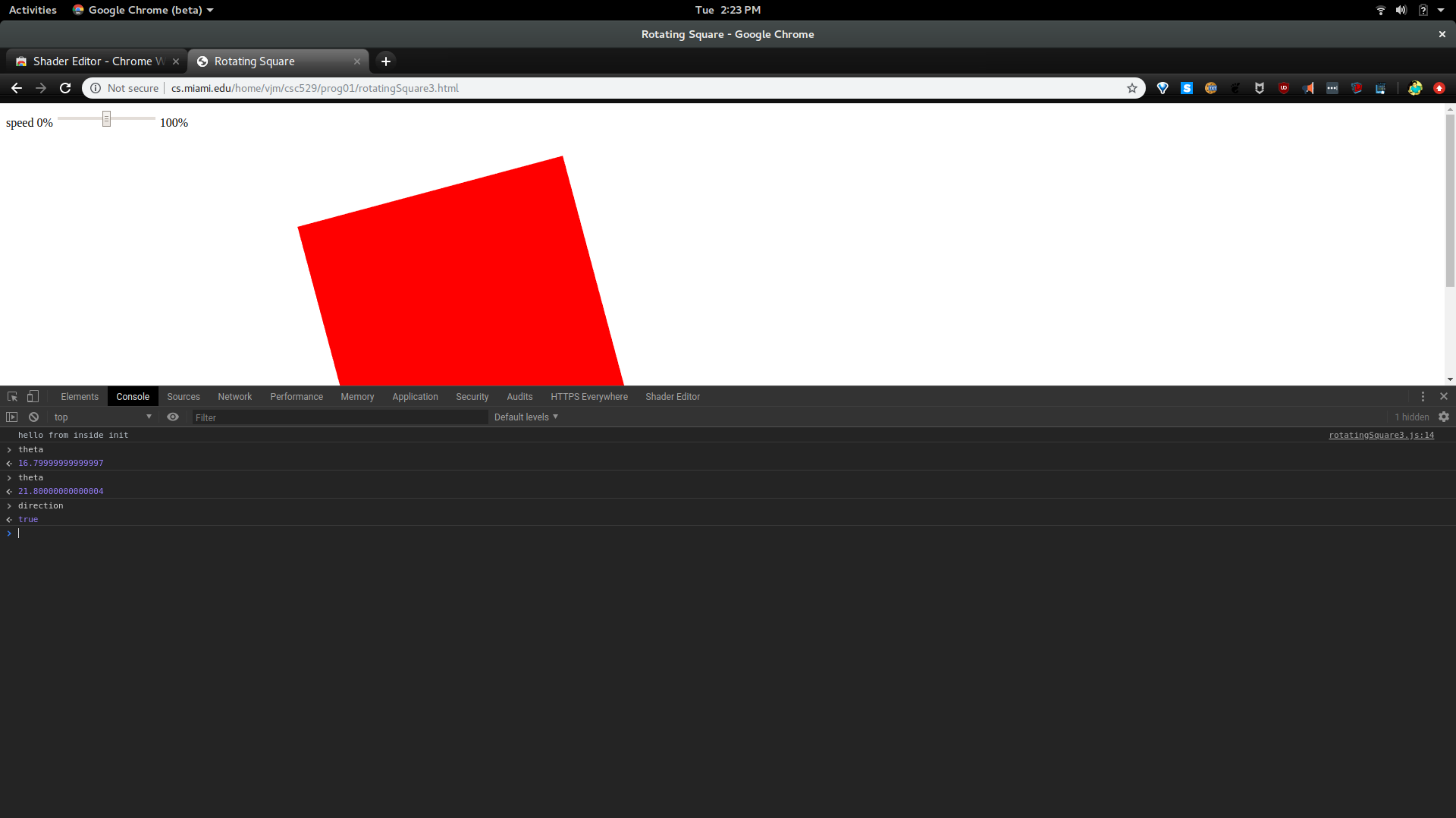
Global Listeners

Event Listener Breakpoints

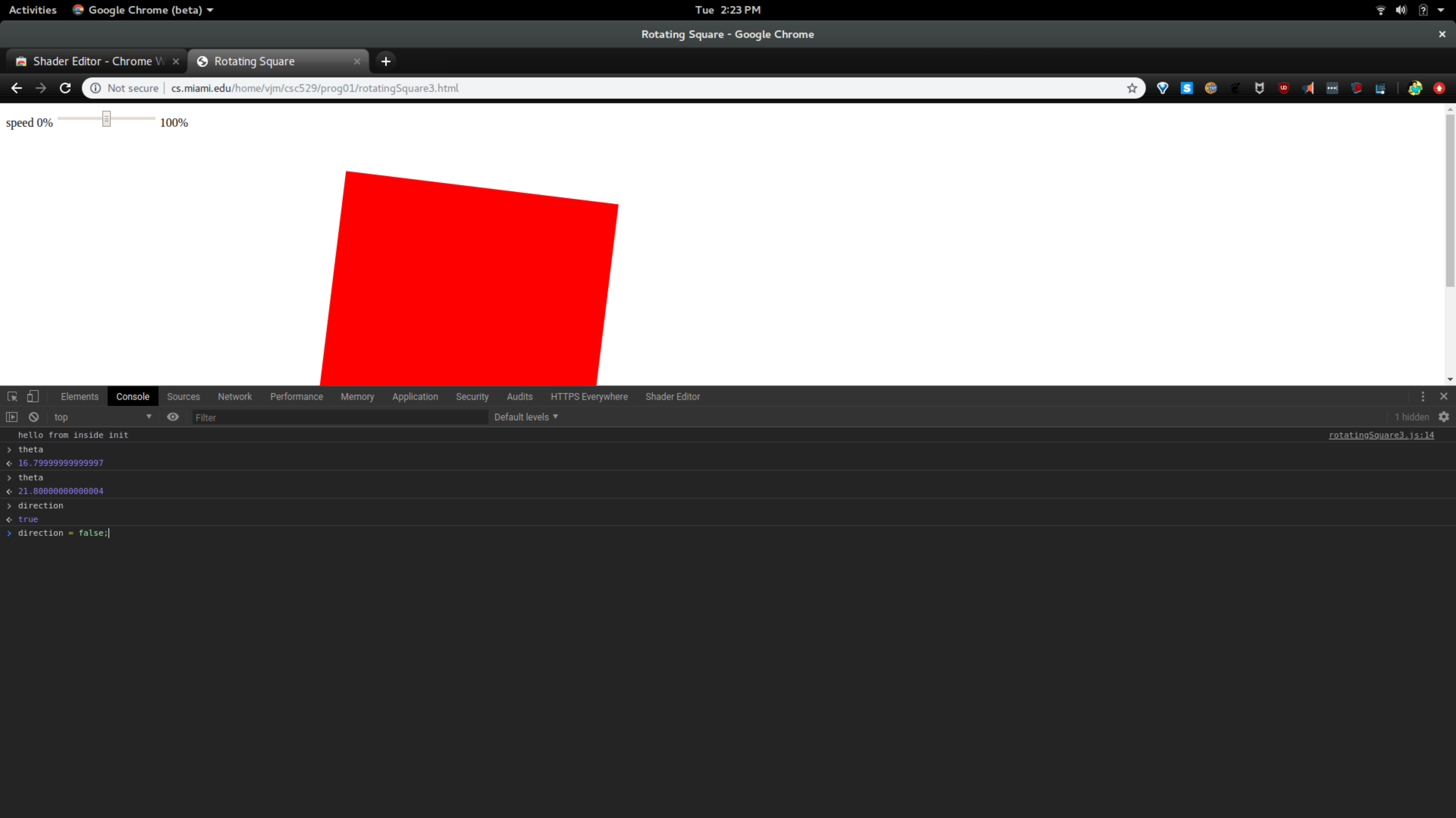
Scope Watch

Not paused

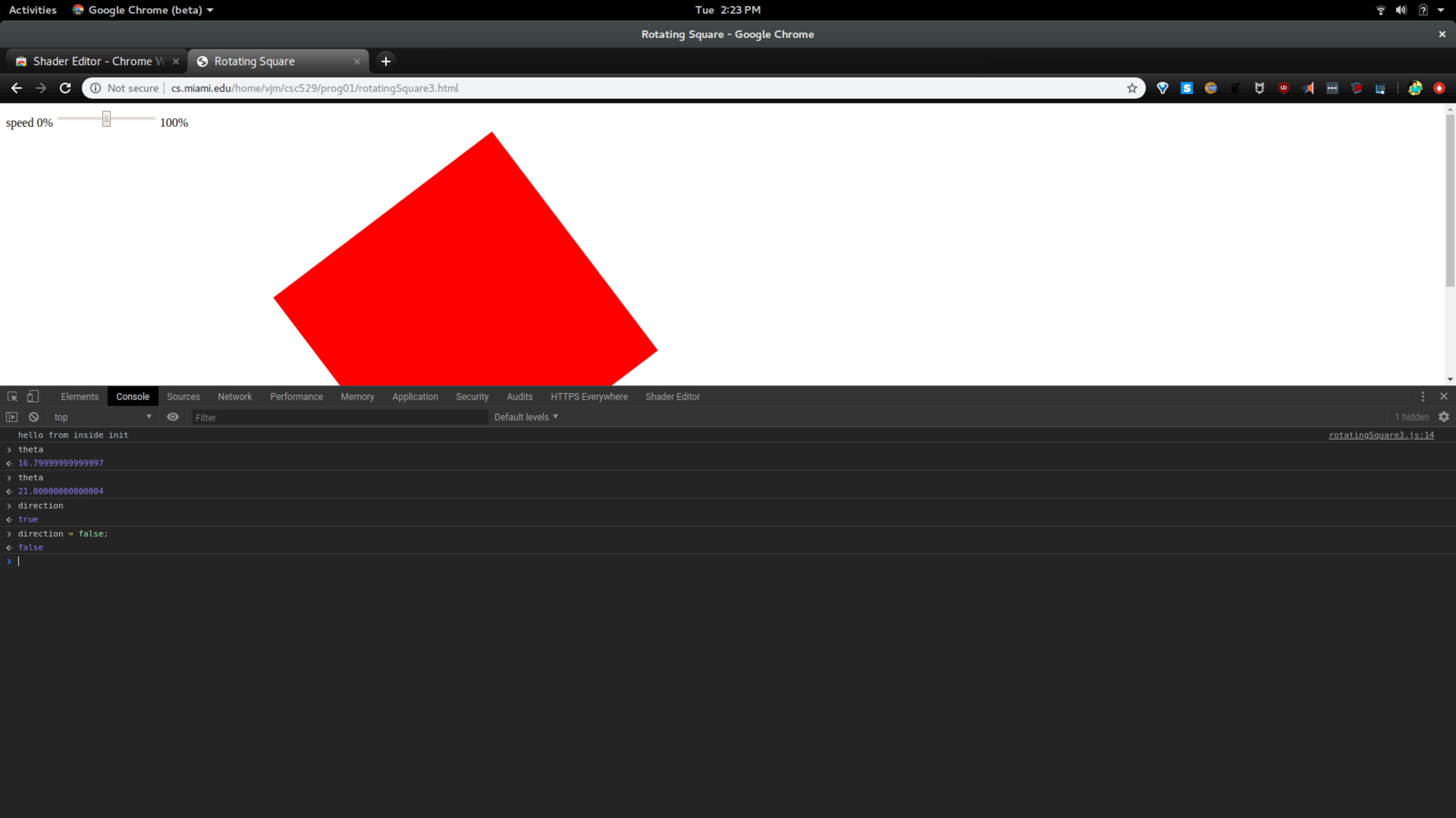
In our script we have a variable declared in the global scope called “direction”. It is currently set to “true”, meaning the square will rotate counterclockwise.



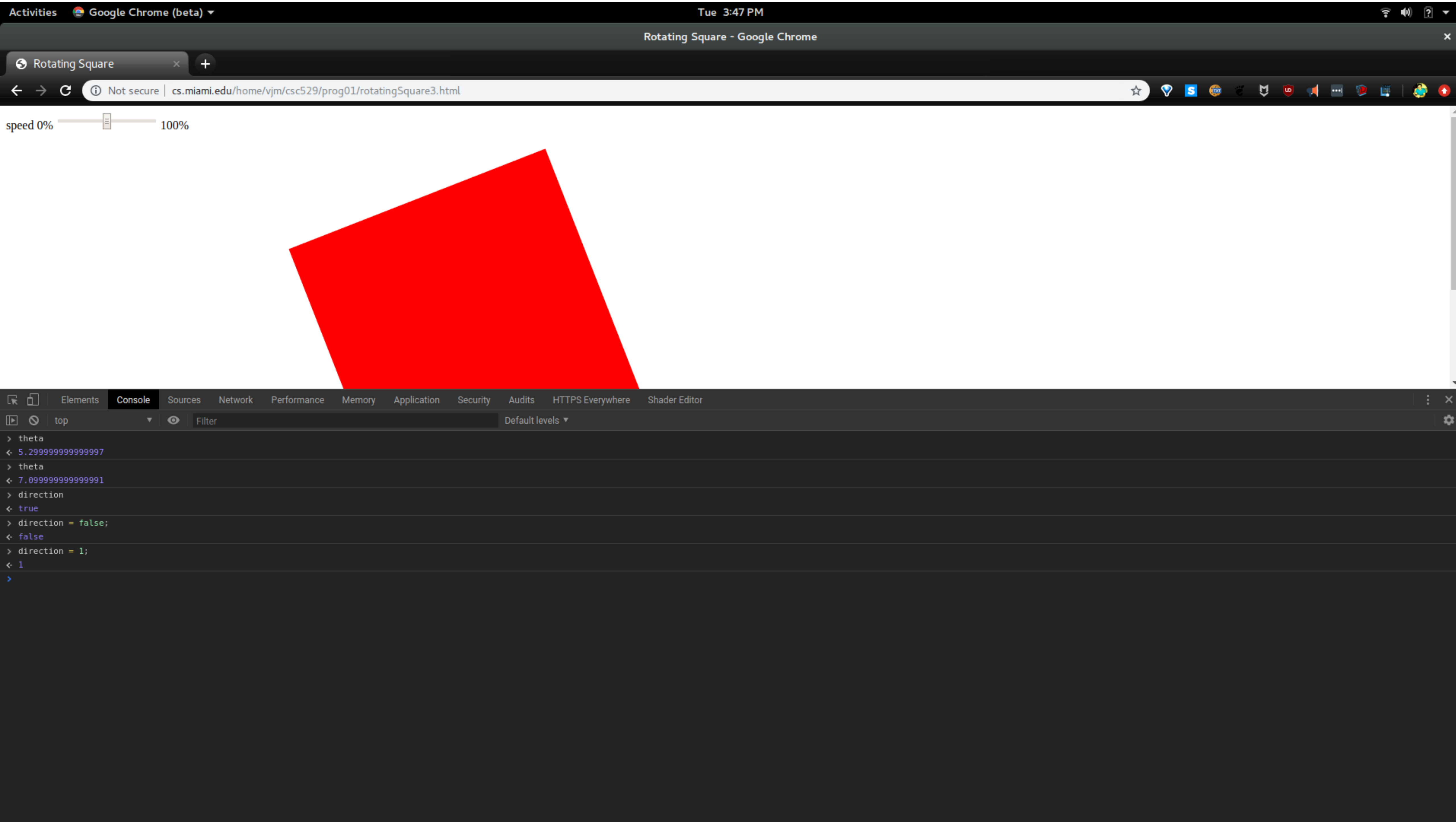
We can check that the current value of “direction” is true, and confirm it rotates counterclockwise.



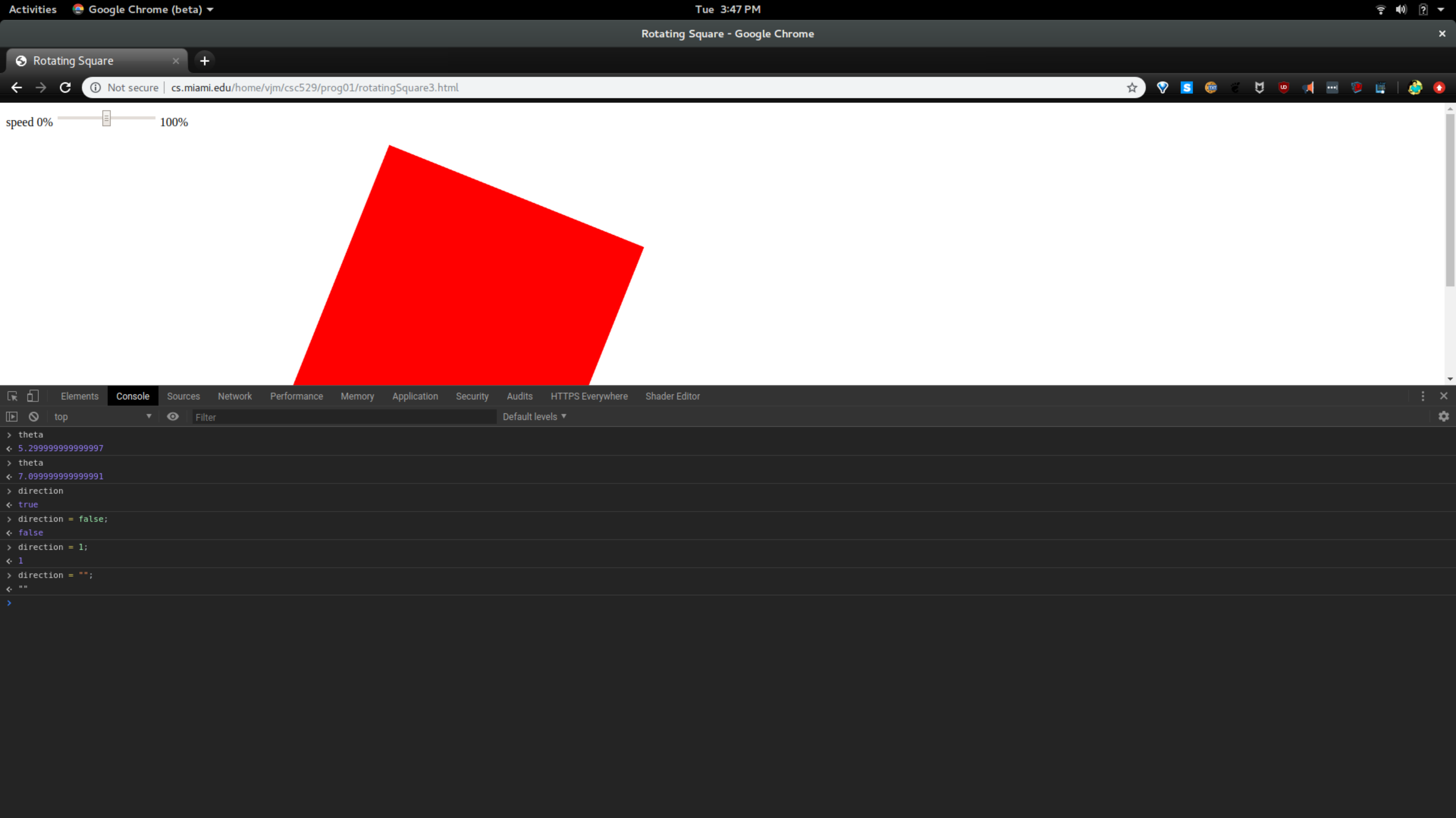
The REPL allows you to modify variables as well. Let's set it to false.



Verify that the square is now rotating clockwise.



Let's try setting the direction to 1. Wait, but isn't "direction" a boolean? Well in C, the number 0 means false and anything else (including 1) means true. Maybe javascript behaves this way? Verify that the square is rotating counterclockwise now.



Now set direction equal to the empty string. What happens now? Surely it will crash... or does it rotate clockwise again? What's going on? Try to find in the code where "direction" is being used, then take a look at: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Comparison_Operators