Library SUICA

Lesson №8

Graphics in a browser

WebGL

What is WebGL?

- Technology for 3D graphics in a browser

Why WebGL?

- Platform independent – from smartphones to computers
- Runs in a browser without specialized environment and plugins
- Uses the graphical hardware

SUICA

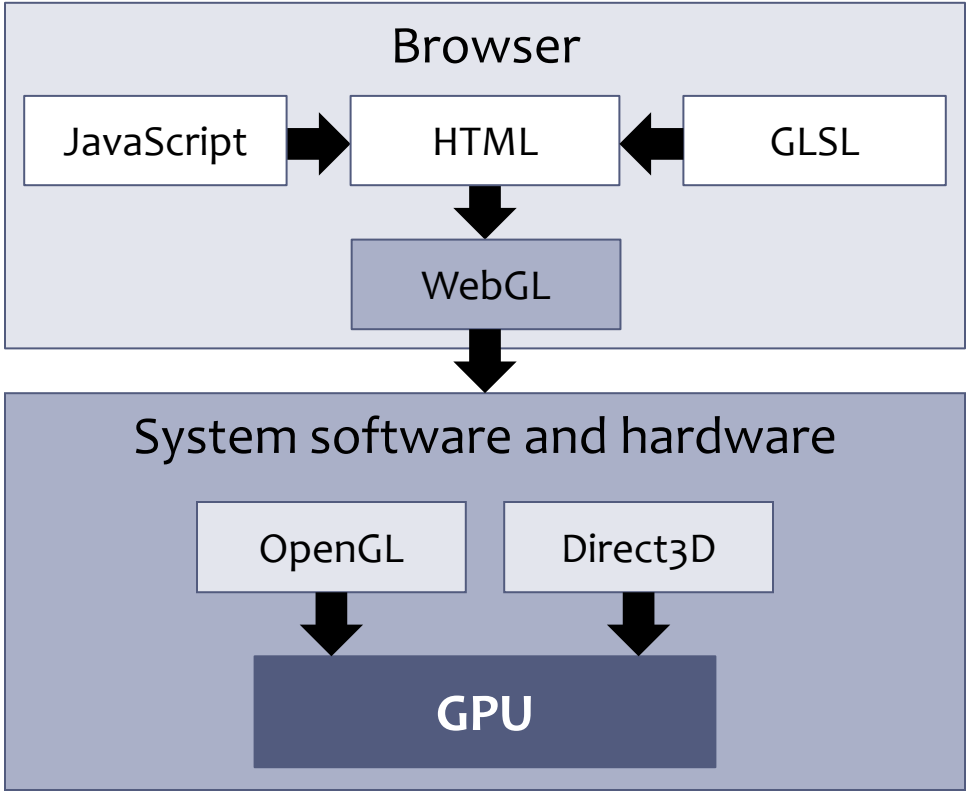
What is Suica?

- A library for 3D graphics, based on WebGL

Why Suica?

- Simple and easy to use objects
- Hide the complexity of WebGL
- Implements some of the elements in a typical interactive graphical mobile application

Architecture of a WebGL application



Application of Suica

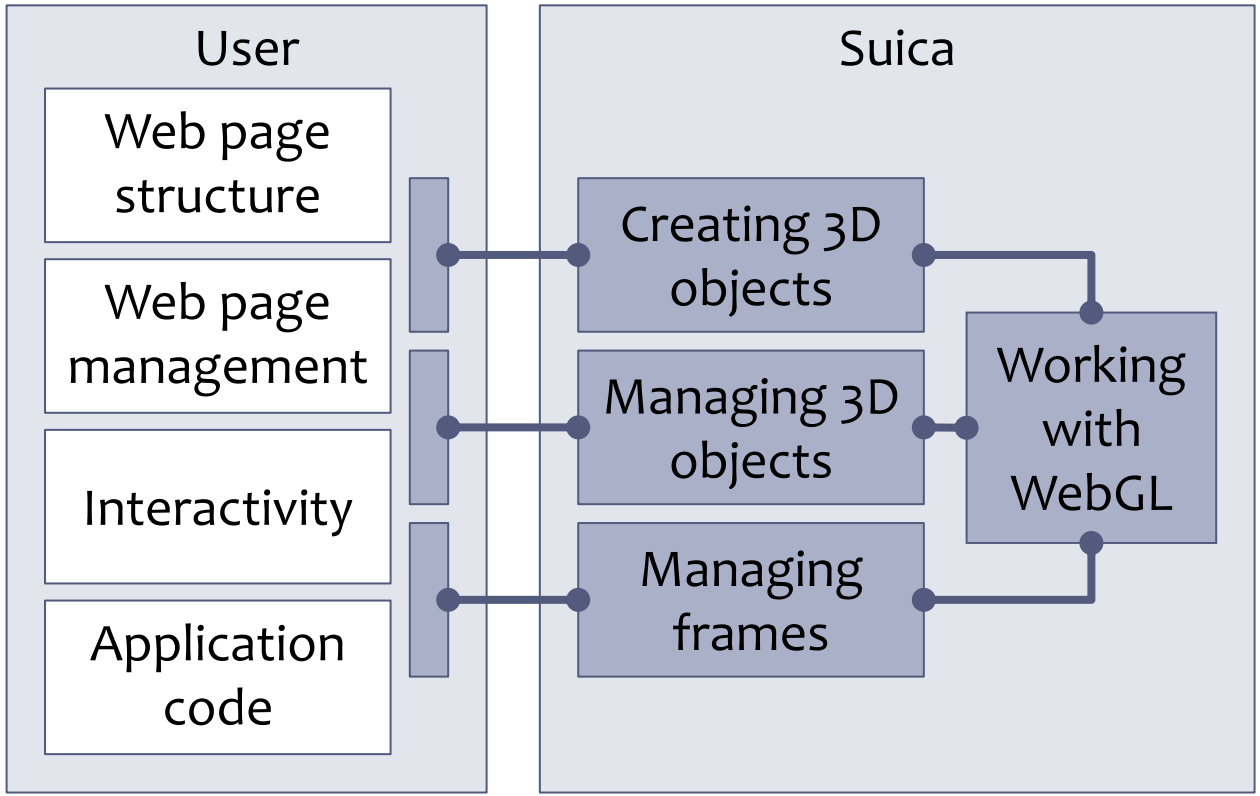
Architecture

Role of Suica

- Creating basic 3D objects
- Controlling their properties
- Generating and updating frames

Role of the user

- Structure and management of the web page
- Interactivity and event processing
- The actual logic of the application

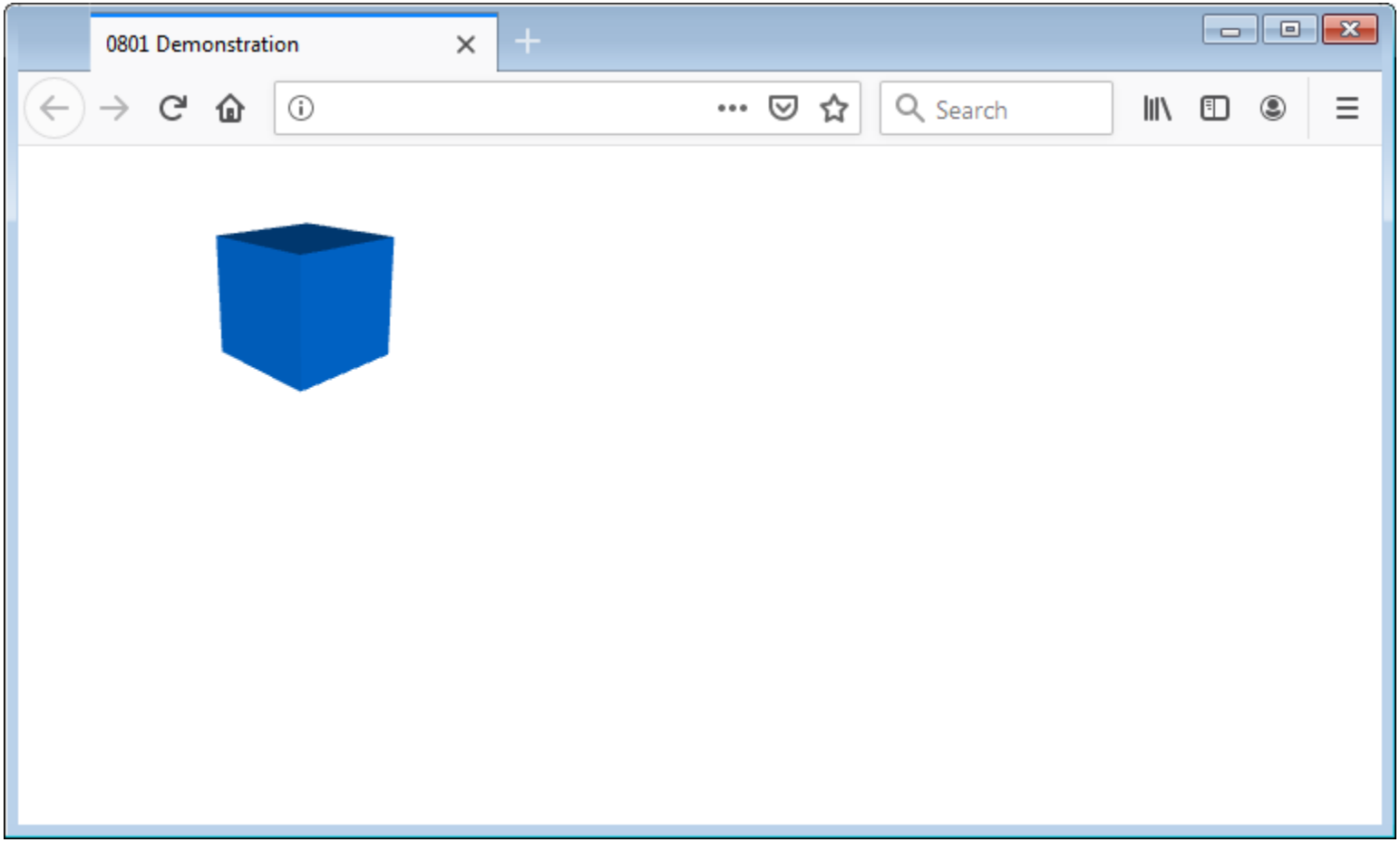

User
Web page structure
Web page management
Interactivity
Application code
Suica
Creating 3D objects
Managing 3D objects
Managing frames
Working with WebGL

Just for demonstration

Minimal program

- Creating a rotating blue cube

```
function main()
{
   new Suica();
   cube([0,0,0],1);
   demo(4);
}
```



Just for illustration

TRY IT

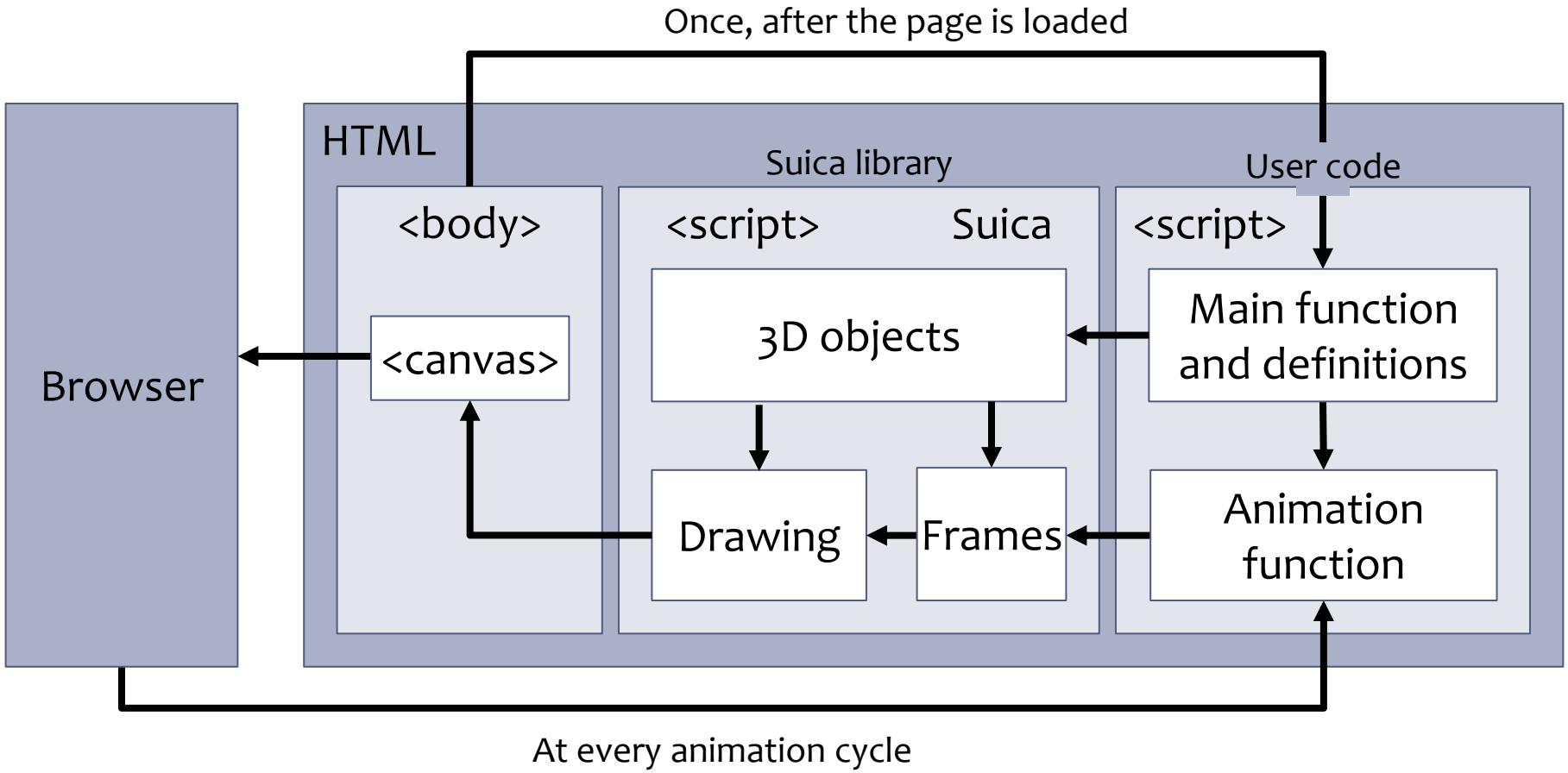
Suica application code

Major elements

- The library itself
- Main function
- Animation function (optional)
- Main function activation
- <canvas> element to contain the graphics

Details

- The library is activated with element <script>
- The main and the other functions are in another <script>
- The <body> contains a <canvas>, where graphics is shown
- After loading <body> the event onload is activated and executes the main function
- The main function creates the graphical objects and sets their main properties
- If there is animation function, it is working continuously


Once, after the page is loaded
HTML
Suica library
User code
<body>
<script>
Suica
<script>
Browser
<canvas>
3D objects
Main function and definitions
Drawing
Frames
Animation function
At every animation cycle

Initialization

Scripts

Script elements in <head>

- Includes the library
- Minimized version suica.min.js
- Full version (with comments) suica.js

```
<head>
   <script src="suica.min.js"></script>
<head>
```

Program

- In a separate scrip in the <head>
- There must be a main function
- The name is defined by the user, traditionally it is main

```
<head>
   <script src="suica.min.js"></script>
   <script>
      function main()
      {
         ...
      }
   </script>
</head>
```

In <body>

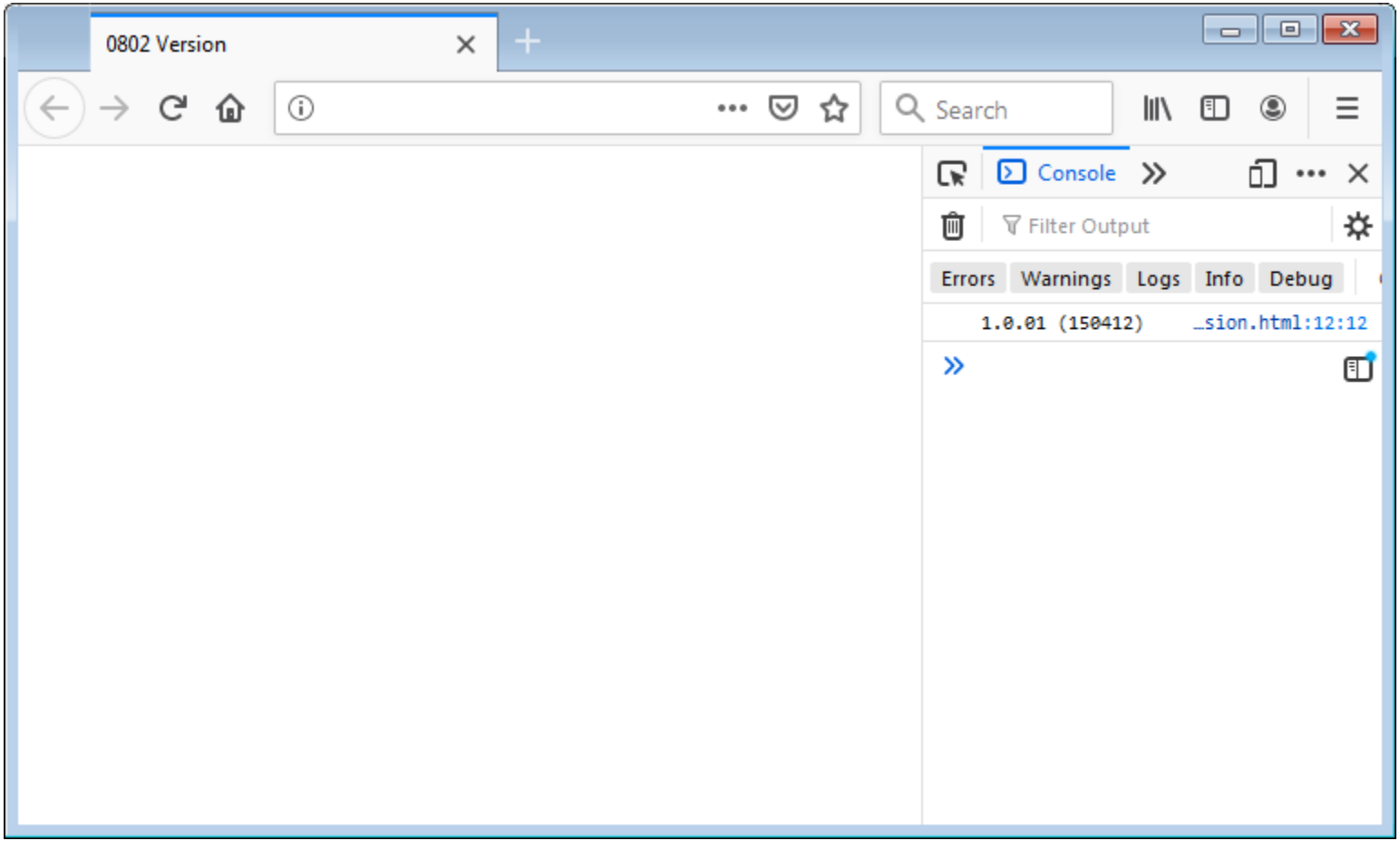
- Suica initialization expects the web page is fully loaded
- Capturing the onload event, activated after the page is loaded
- Calling the main function
- The element <noscript> shows warning if JavaScript is unavailable or turned off

```
<body onload="main()">

   <noscript>
      No JavaScript!
   </noscript>

</body>
```

The main function

- Showing the Suica version number

```
function main()
{
   console.log(Suica.version);
}
```


0802 Version
Search
Console
Filter Output
Errors
Warnings
Logs
Info
Debug
1.0.01 (150412)
…sion.html:12:12

TRY IT

Graphical window

Element <canvas>

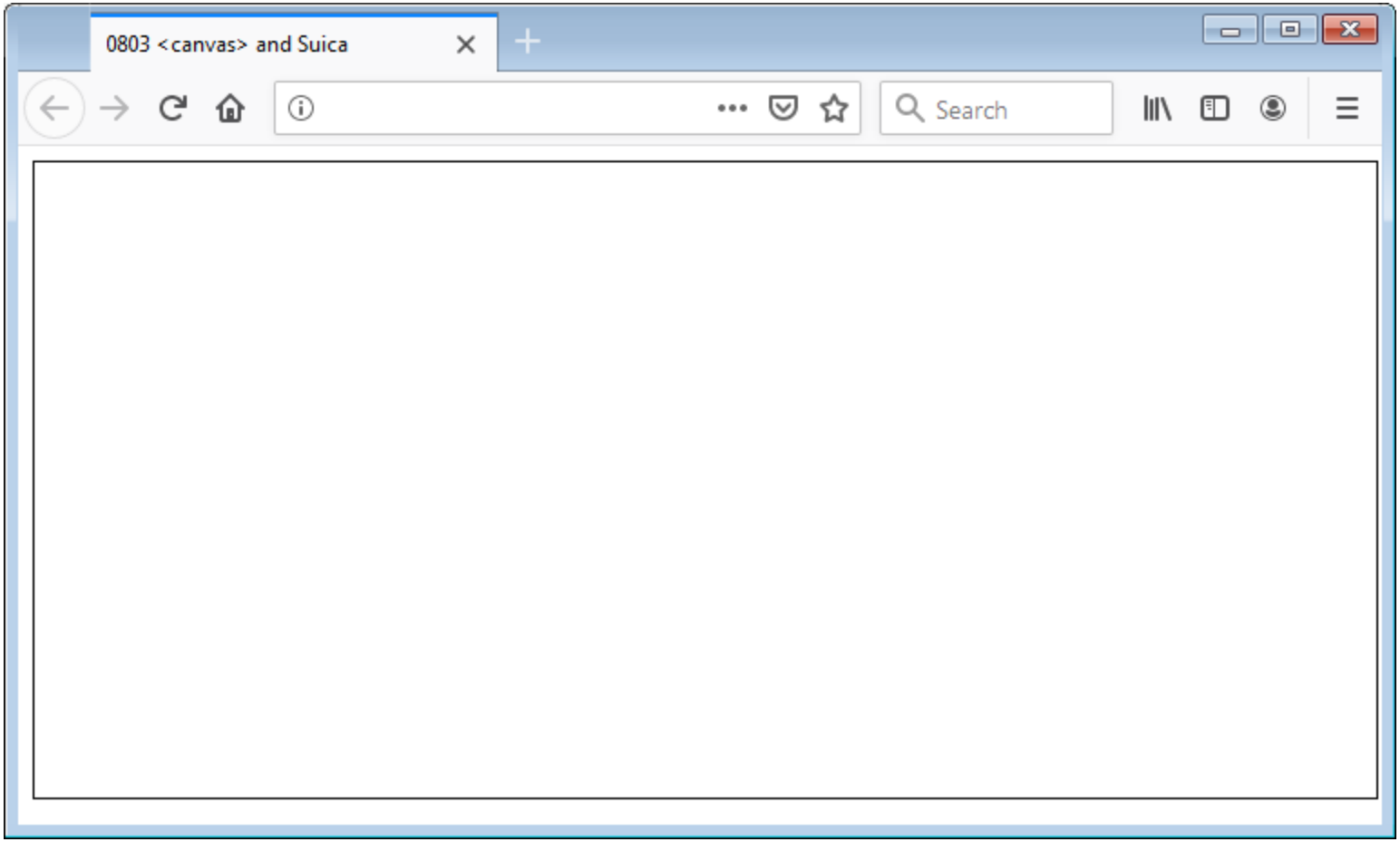
- WebGL draws only in <canvas>
- Every <canvas> is a graphical window to Suica
- Creating and configuring the canvas (size, border, …)
- Checking browser support for <canvas>

```
<canvas width="740" height="350"
                    style="border: solid 1px Black;">
     No canvas!
</canvas>
```

Suica instance

- The Suica library defines an object (instance) called Suica
- For every <canvas> there must be separate Suica object
- Creating a new instance with new

```
function main()
{
   new Suica();
}
```



TRY IT

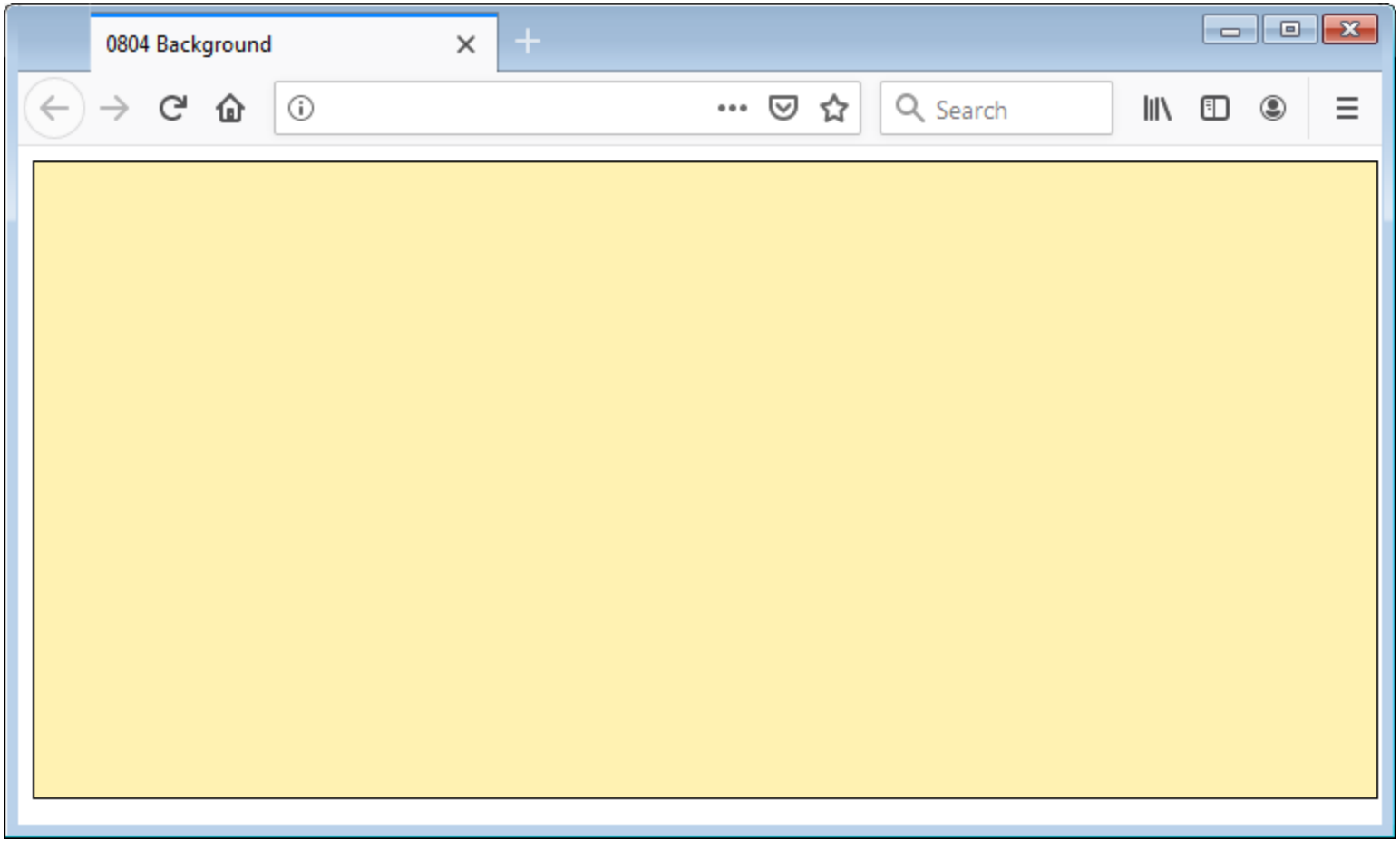
Content of <canvas>

- Cannot be controlled by HTML and CSS
- Controlled by Suica

Background colour

- Colours are arrays of 3 numbers – red, green and blue
- Values are floatin point numbers from 0 to 1
- The colour is defined with background (*colour*)

```
background([1,0.95,0.7]);
```

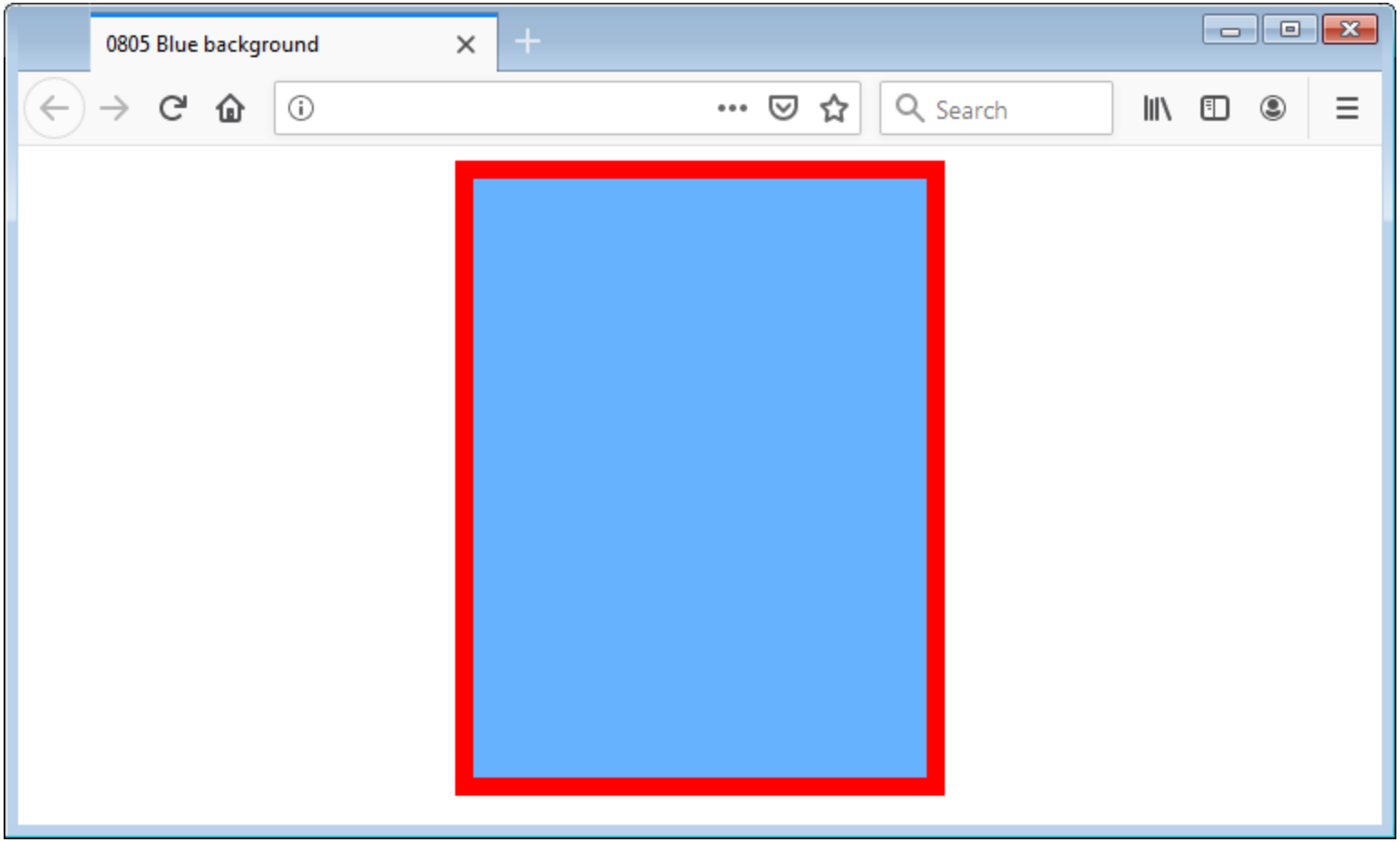


TRY IT

Another example

- Bluish background
- Centered graphical area
- Thick red border

```
<style>
   body   { text-align: center; }
   canvas { border: solid 10px Red; }
</style>
:
background([0.4,0.7,1.0]);
:
<canvas width="250" height="330">...</canvas>
```



TRY IT

Colours

Colours

Colours in Suica

- They use the RGB model
- Arrays of three numbers: **red (R)**, **green (G)** and **blue (B)**
- Each number is from 0 to 1
- Minimal intensity at 0, maximal – at 1

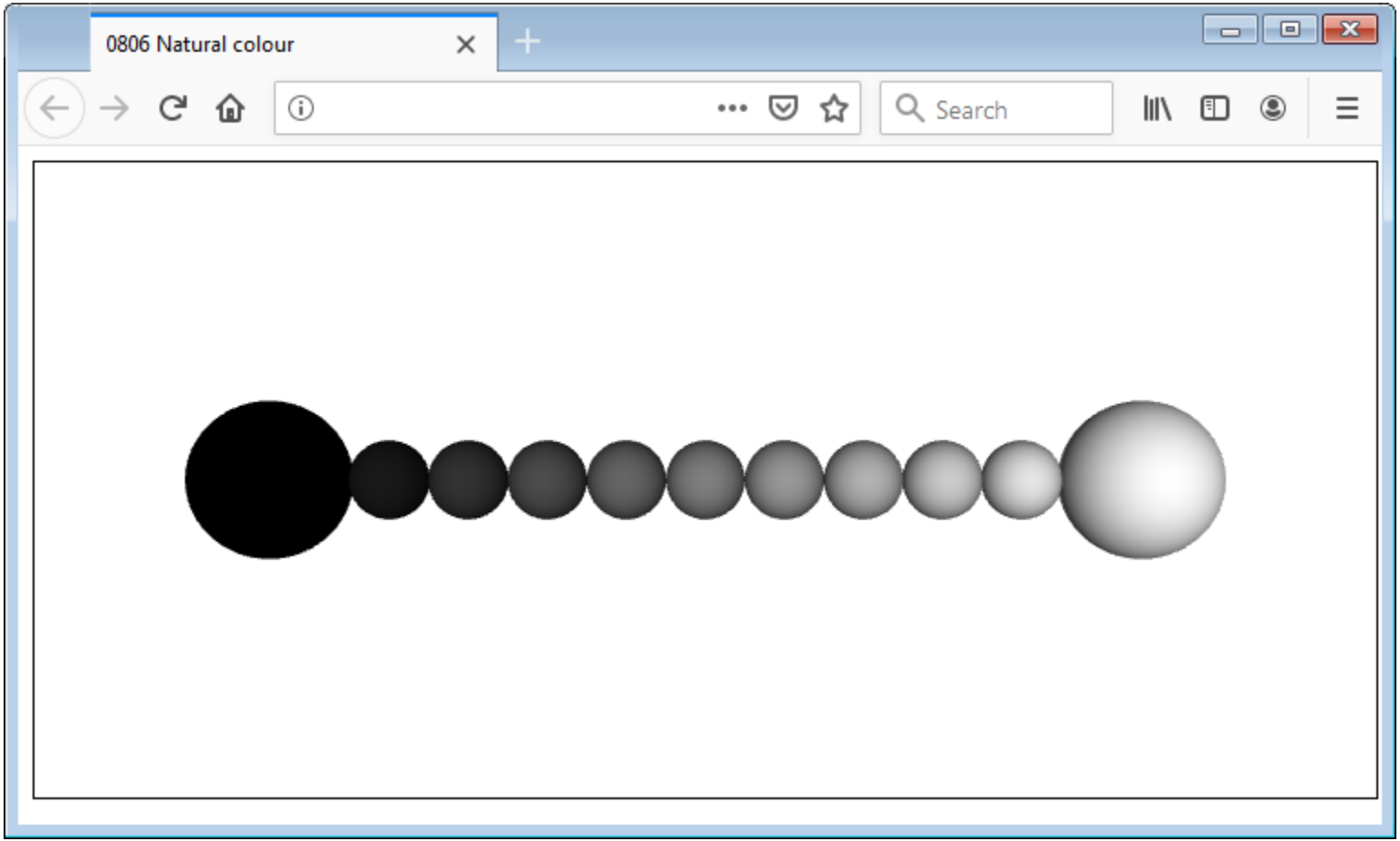
Neutral colours

Neutral colours R=G=B

- White (1,1,1) and black (0,0,0)
- And all shades of gray (x,x,x)

Calibration

- If the monitor is not colour-calibrated, the neutral colours may have colour tint



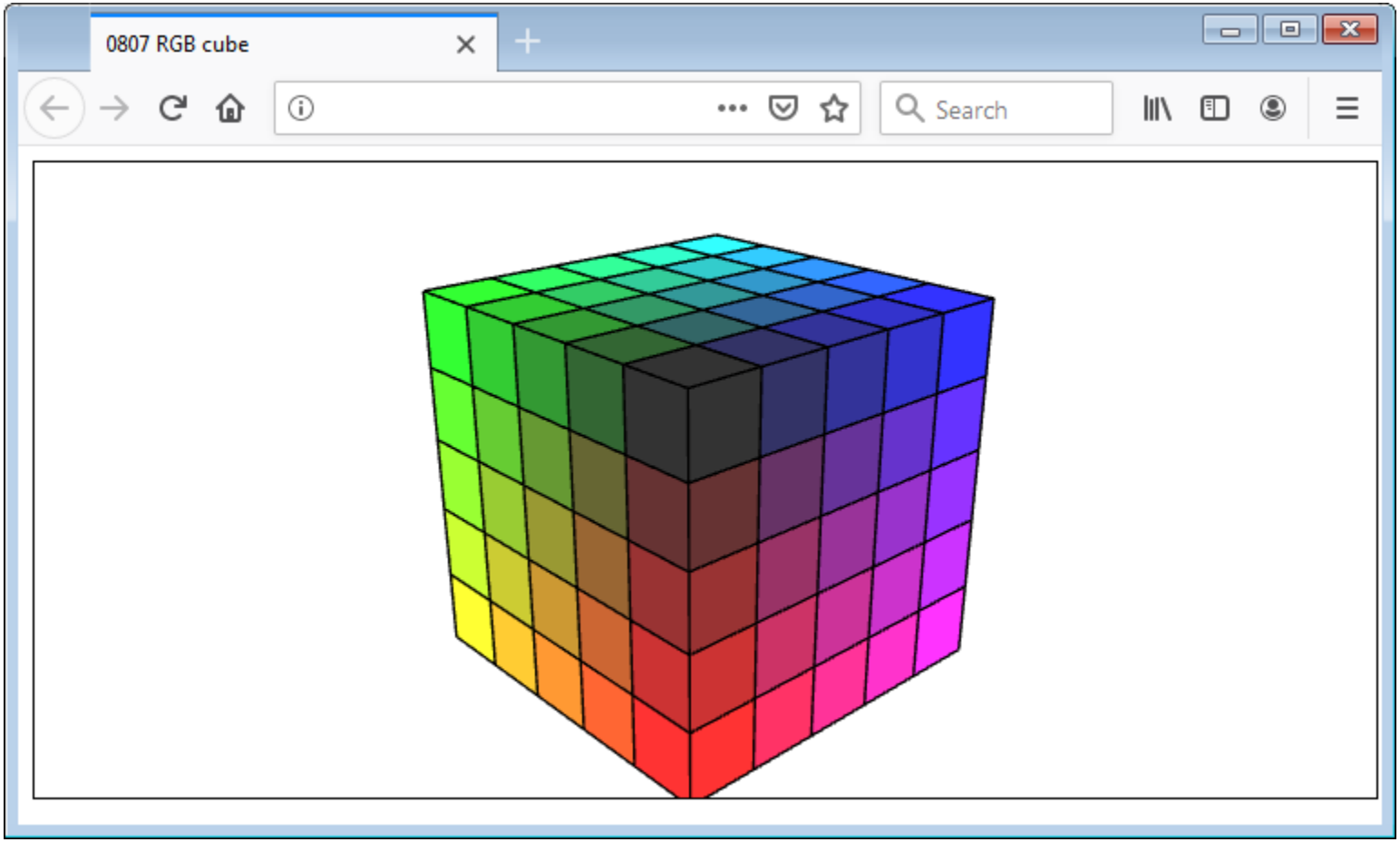
Only for illustration

TRY IT

Colour arithmetic

Primary colours

- 0 0 0 = black
- 1 0 0 = red
- 0 1 0 = green
- 0 0 1 = blue
- 1 1 0 = red+green = yellow
- 1 0 1 = red+blue = magenta
- 0 1 1 = green+blue = cyan
- 1 1 1 = red+green+blue = white



Only for illustration

TRY IT

Help commands

Coordinate system

Coordinate system in Suica

- Cartesian, 3D
- Always exists
- Direction of axes are relative

Visualizing the coordinate system

- Not possible, it is an abstract concept
- However, objects could be drawn instead of the axes

Help command oxyz

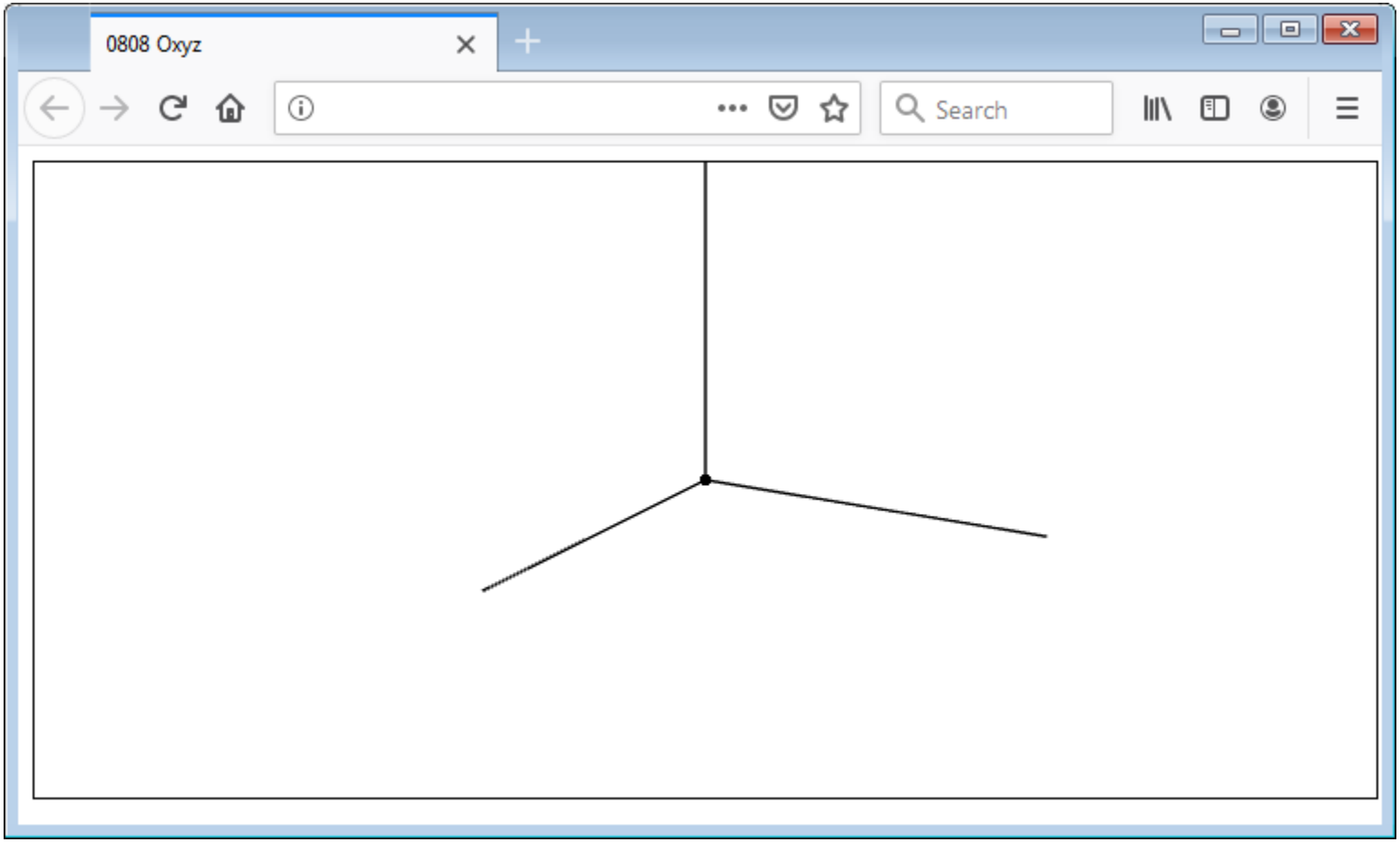
- Create a dot for the origin and thee segments for the axes
- The parameter of the command is the length of the axes (by default it is 30)

```
oxyz();
```

```
oxyz(100);
```

Orientation

- The origin is in the center of <canvas>
- X is downwards to the left, Y is to the right, Z is upwards



TRY IT

Orbiting

Help command demo

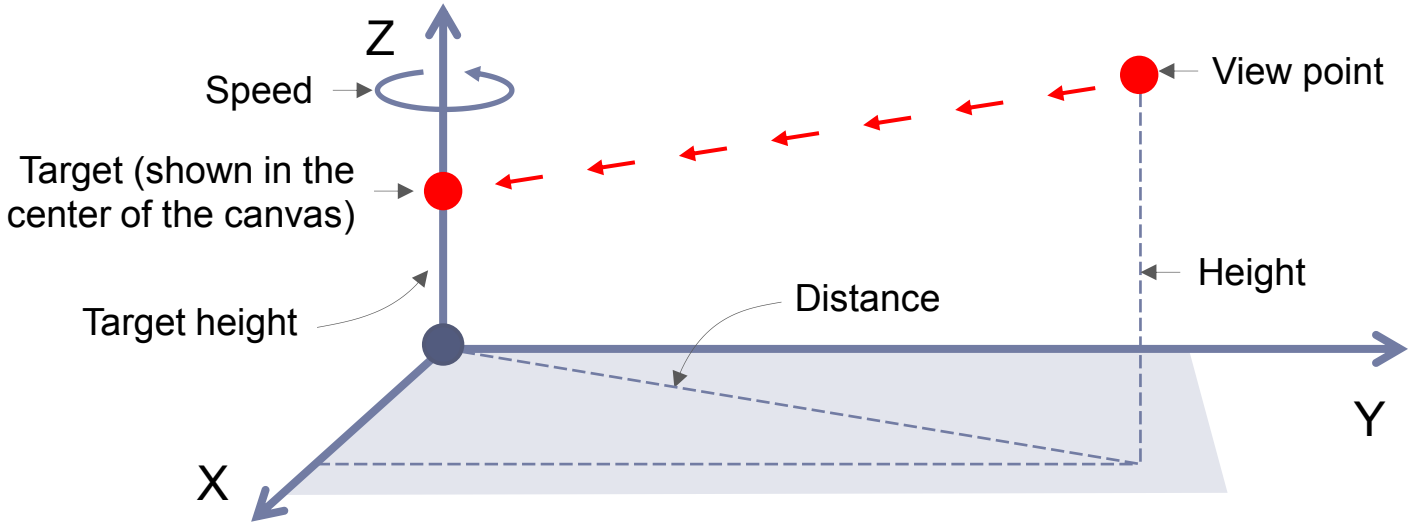
- Activates automatic rotation mode
- Alternative representation – view point orbiting

Parameters

- Horizontal distance
- Speed (18° in a second, full turn in 20 seconds)
- Height (% of distance)
- Target height (% of distance)

Also

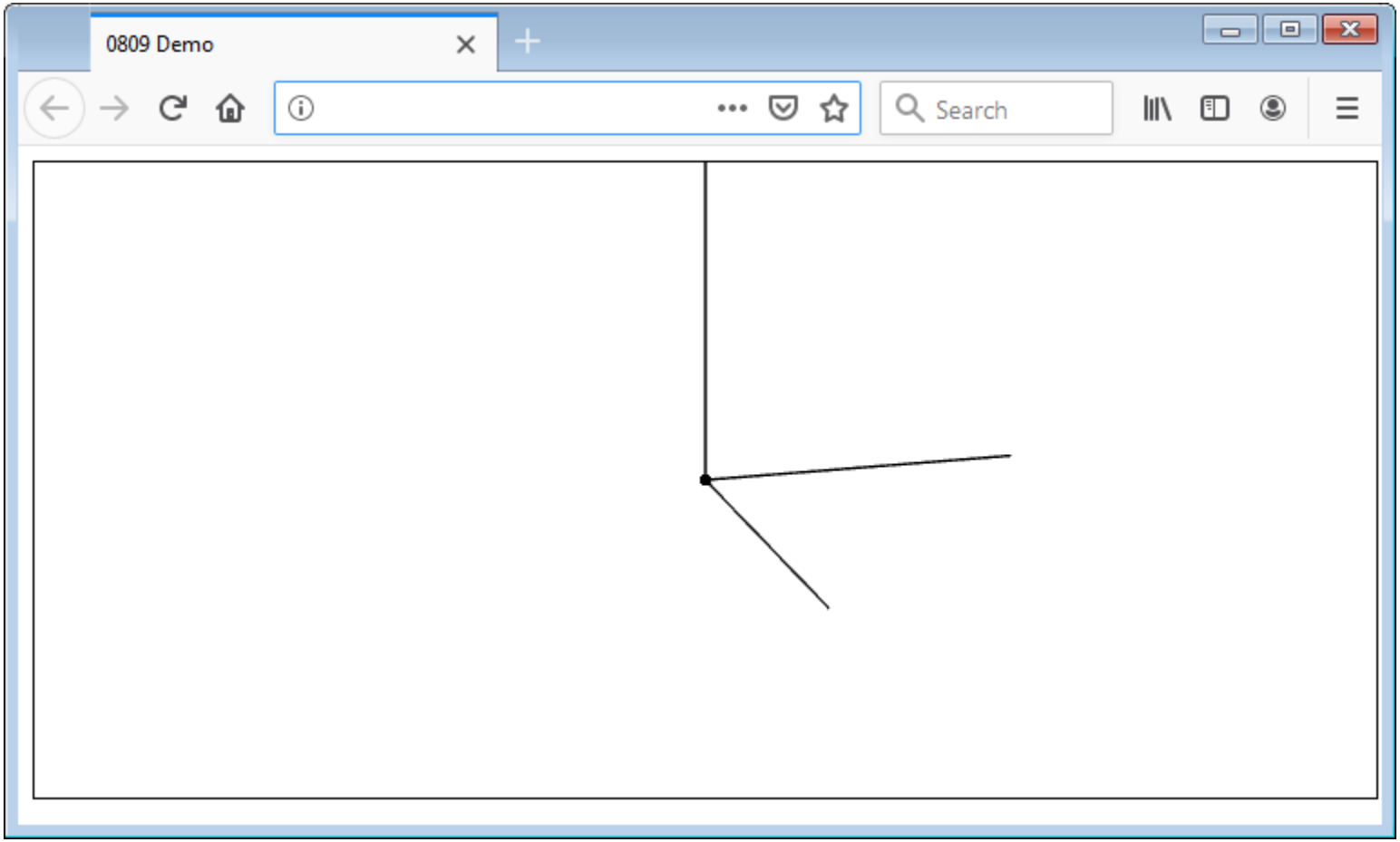
- All parameters are optional
- Only the trailing parameters could be skipped



Example

- Without parameters
- Default distance is 100
- Height is 0.3
- Speed is 1
- Target height is 0.1

```
function main()
{
   new Suica();
   oxyz();
   demo();
}
```

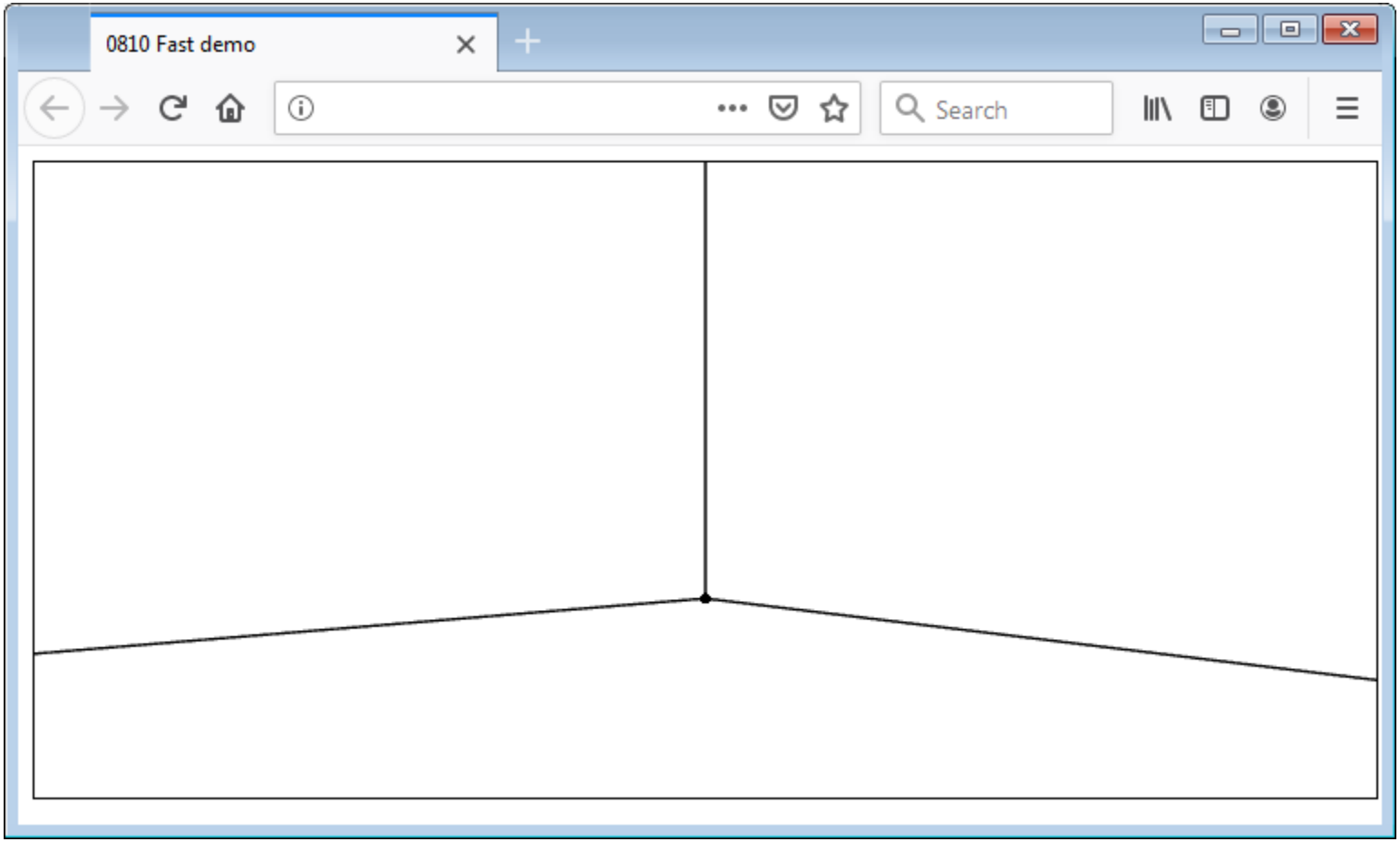


TRY IT

Example 2

- Shorter distance (50 units)
- 9 times faster rotation
- Just above the XY plane (5 units)
- Horizontal view to the target (also 5 units)

```
function main()
{
   new Suica();
   oxyz();
   demo(50,9,0.1,0.1);
}
```



TRY IT

Projects

Help functions in projects

- Function oxyz and demo are just help functions
  - Visualization of the coordinate system
  - Automatic orbiting
- They are not expected to be use din the projects
  - Projects have more complex visualization
  - Motion and animations are also more complex
- Using oxyz and demo in projects is not evaluated

Summary

Online graphics

WebGL

- Technology for 3D graphics in a browser
- Multiplatform

Suica

- Uses WebGL
- Simplifies the creation of graphics

Using Suica

- Loading with suica.min.js or suica.js
- Activation of the main function in onload of <body>
- Drawing only in the <canvas>
- One Suica for each <canvas>

More about Suica

- Suica – class Suica
- version – current version of the library
- background – background colour
- oxyz – an image of the coordinate system
- demo – automatic orbiting of the view point

The end

Comments, questions