GRAPHICS SYSTEMS AND INTERACTION

Lesson 2

Abstract

Project "Basic Watch"
Project "Interactive Watch"

João Paulo Pereira
jjp@isep.ipp.pt

Table of Contents

Table of Figures

Table of Equations

Project “Basic Watch”

The aim of this three.js [1] project is to create and animate a scene composed by five instances of an analogue watch that display the time in several cities around the world (Figure 1).

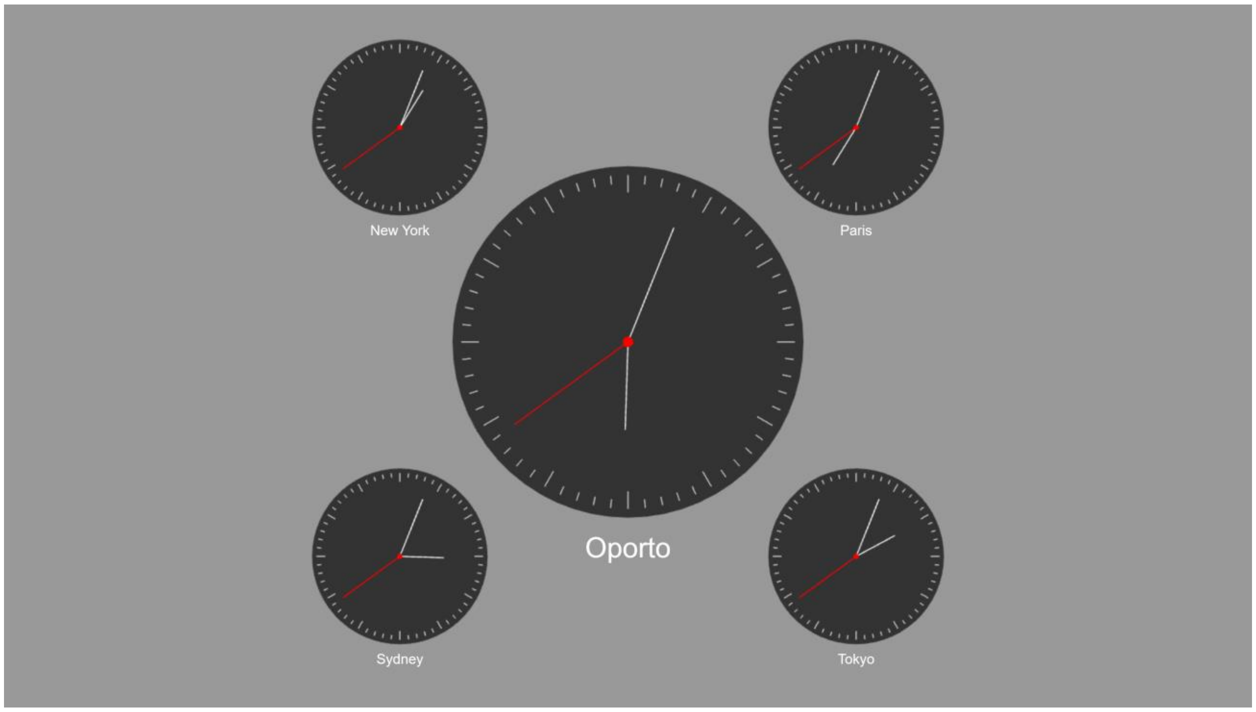

Figure 1 – Project “Basic Watch”

Download the folder “Basic_Watch_template”. The project is composed by two files:

- “Watch_template.html”
- “watch_template.js”

The scene, camera and renderer are created in the first file, along with the instantiation of the five watches, and the definition of an animation function and a callback that processes window resizing events.

The class Watch is defined in file “watch_template.js”. It includes the constructor, which models the watch in 2D, and the update method, responsible for regularly compute the hands angles.

The model itself is a group of objects: the dial (a circle), sixty markers (line segments of two different sizes), the hour hand (a line segment), the minute hand (another line segment) and the second hand (a subgroup of objects comprising a line segment and a small circle).

Your assignment is to write the code that models most of these objects, computes the angles of the hour and minute hands and animates the scene. The second hand has been previously modeled, and its angle is already being updated; the main watch is running already.

The coordinate system

Assume that the 3D Cartesian coordinate system is defined as represented in Figure 2. Since this is a 2D project, the Z-axis and Z-coordinates can be ignored most of the time.

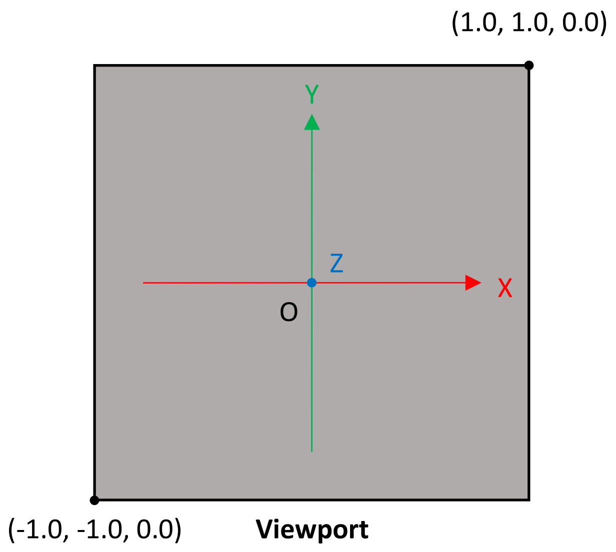

Figure 2 – The coordinate system

To-do #1 – Create the dial

Open the file "watch_template.js" and look for comment "To-do #1". Follow the instructions.

Code examples and classes to consider: CircleGeometry [2], MeshBasicMaterial [3], Mesh [4].

To-do #2 – Create the sixty markers

Start by considering three imaginary circles centered on the origin of the coordinate system, with different radii (Figure 3). Each of the twelve main markers is a line segment connecting a point on the inner circle to the corresponding point on the outer one. The remaining markers are line segments connecting points on the middle circle to the equivalent points on the outer one.

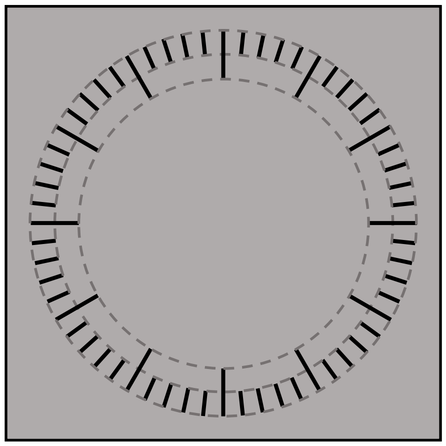

Figure 3 – Markers

Use the parametric form of the circle equation to compute the points coordinates (Figure 4 and Equation 1) [5].

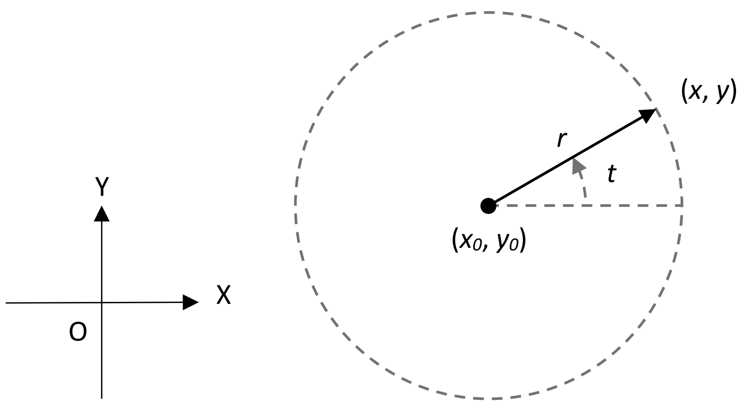

Figure 4 – A generic circle

$$\begin{cases} x = r * \cos(t) + x_0 \\ y = r * \sin(t) + y_0 \end{cases}$$

Equation 1 – Parametric form of the circle equation

Where:

- (x, y) are the point coordinates
- (x_0, y_0) = (0.0, 0.0) are the center coordinates
- r is the radius
- t is a parametric variable in the range $0.0 \leq t < 2.0 * \pi$ (pi)

Don't forget that angles must be expressed in radians (180.0 degrees = π radians).

Code examples and classes to consider: Drawing lines [6], Vector3 [7], BufferGeometry [8], LineBasicMaterial [9], Line [10], LineSegments [11].

Look for comment "To-do #2" and follow the instructions.

To-do #3 – Create the hour hand

Look for comment "To-do #3" and follow the instructions.

To-do #4 – Create the minute hand

Look for comment "To-do #4" and follow the instructions.

To-do #5 – Compute the minute hand angle

The minute hand angle depends mostly on the current minutes value, but you will get a more accurate result if you make it depend on the seconds value as well.

Look for comment "To-do #5" and follow the instructions.

To-do #6 – Compute the hour hand angle

The hour hand angle depends mainly on the current hours value. Nevertheless, you will get a much better result if you make it also depend on the minutes and seconds values.

Look for comment "To-do #6" and follow the instructions.

To-do #7 – Animate the remaining watches of the scene

Open the file “Watch_template.html” and look for comment “To-do #7”. Follow the instructions.

To-dos #8 to #12 – Play with transformations

Play with linear (scaling, rotation) and affine (translation) transformations.

To-do #8 – Scaling

Look for the line following comment “To-do #8” and uncomment it. You will notice that the main watch width has changed.

To-do #9 – Rotation

Look for the line following comment “To-do #9” and uncomment it. The watch has now rotated 45 degrees counterclockwise.

To-do #10 – Translation

Re-comment the two previous instructions. Look for the line following comment “To-do #10” and uncomment it. The watch moved to the right.

To-do #11 – Reflection (a special case of scale transformation)

Re-comment the previous instruction. Look for the line following comment “To-do #11” and uncomment it. The watch appears now to be reflected in a mirror.

To-do #12 – Restore the previous state

Re-comment the previous instruction.

Project “Interactive Watch”

This is a refinement of project “Basic Watch”. Differences are as follows (Figure 5):

- The sixty markers were created in a different way
- The hour, minute and second hands look better now
- Five instances of a drop-down menu were included to allow users to select cities

Your assignment is to adapt project “Basic Watch” to accommodate these refinements.

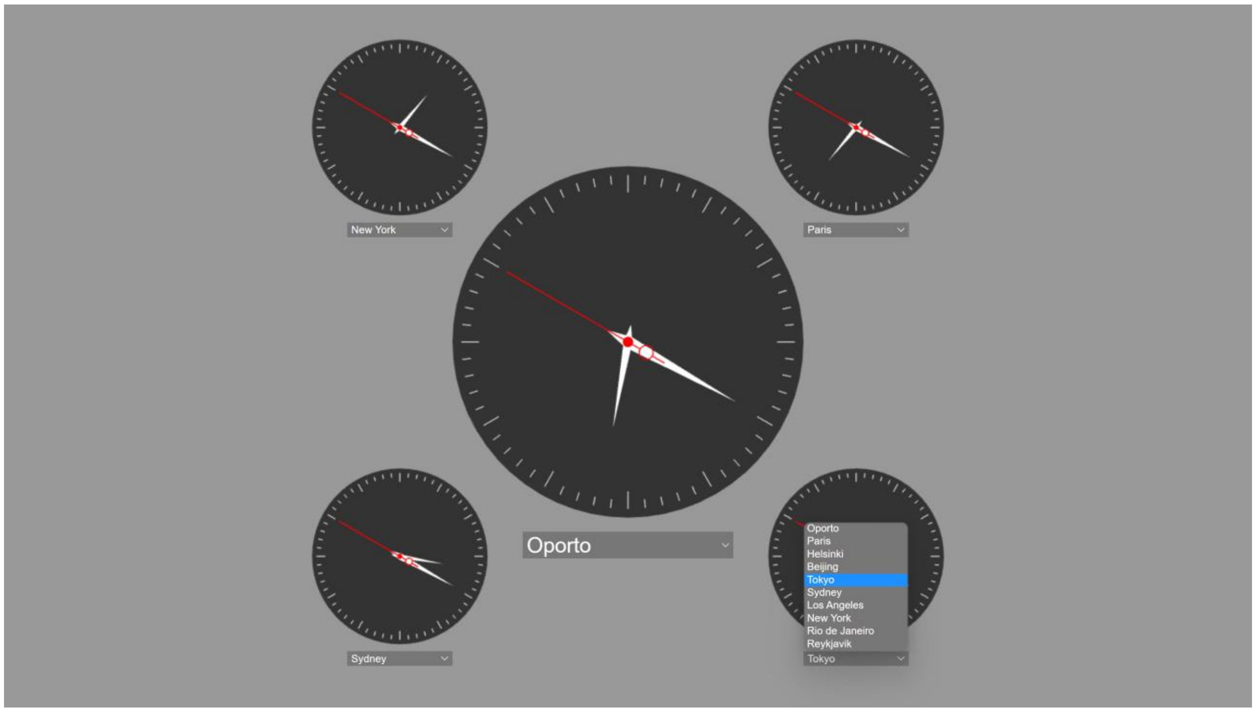

Figure 5 – Project “Interactive Watch”

To-do at home #1 – Create the sixty markers

To create the sixty markers, consider using the class EllipseCurve [12] instead of the circle equation (includes code example).

To-do at home #2 – Create the hour and minute hands

To create the hour and minute hands, consider using two-triangle meshes instead of line segments.

Note that the relative order in which the vertices of each triangle are specified is important: it should be counterclockwise (CCW winding). Check Figure 6 and Figure 7 for details.

Code examples and classes to consider: BufferGeometry [8], BufferAttribute [13].

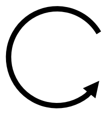

Figure 6 – CCW winding

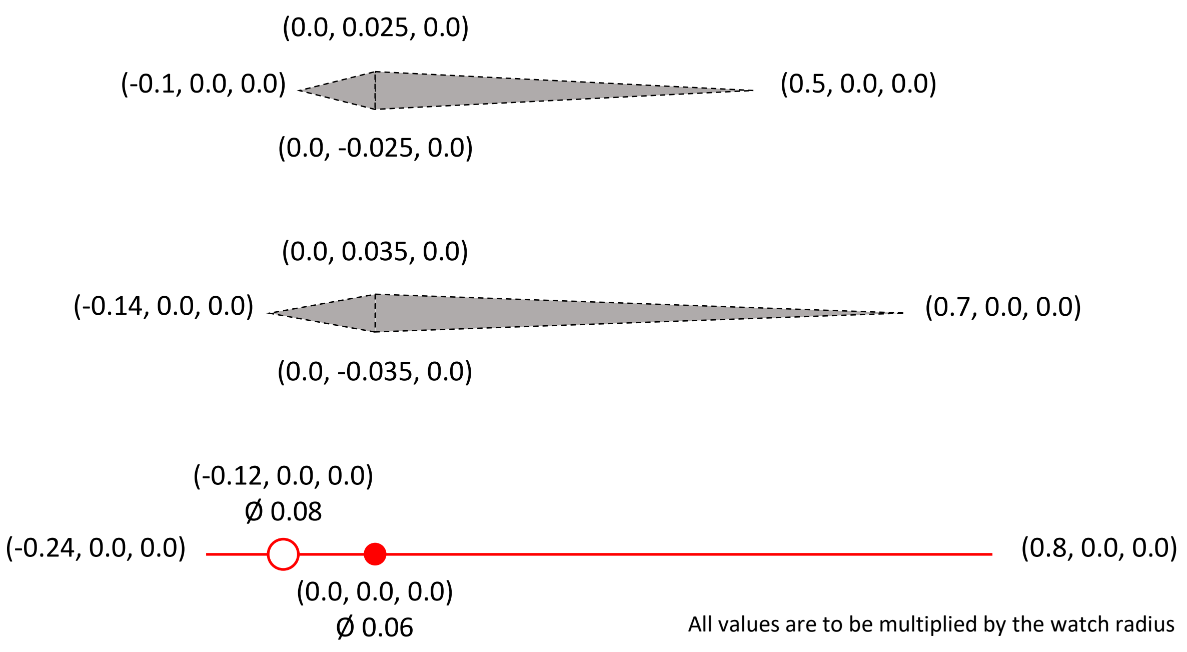

Figure 7 – Hour, minute and second hands

To-do at home #3 – Create the second hand

To create the second hand consider using a group of objects comprising two line segments, a small circumference, and a tiny circle (Figure 7). The circumference can be created in one of the following ways:

- By using the parametric form of the circle equation plus classes BufferGeometry [8] and LineLoop [14]
- By using classes EllipseCurve [12], BufferGeometry [8] and Line [10]
- By using classes CircleGeometry [2], EdgesGeometry [15] (includes code example) and LineSegments [11]

To-do at home #4 – Create a drop-down menu

To create a drop-down menu consider using the <select> HTML element (Figure 8) [16] [17].

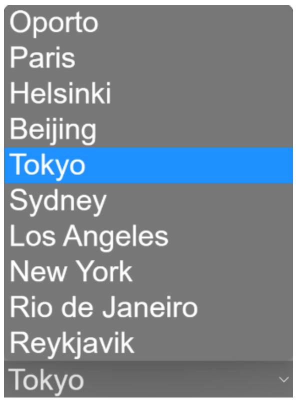

Figure 8 – Drop-down menu

- Open the file “watch_template.js” and look for the following two lines:

```
// Then create a "label" <div> element and append it as a child of
"container"

this.label = document.createElement("div");
```

- Replace `<div>` with `<select>` in the first line and `"div"` with `"select"` in the second
- You may wish to slightly reduce the element’s font size. Look for the line that follows:

```
this.label.style.fontSize = (8.0 * radius).toString() + "vmin";
```

- Replace `8.0` with `6.0`
- Configure the element by entering the options corresponding to the names of the different cities. You will find them in array `this.cities[]`
- Finally, define a callback that handles user choices by changing the value of variable `this.cityIndex` accordingly

To-do at home #5 – Play with project parameters

Create a different layout by changing the number of watches, their positions and sizes, the colors, and so on (Figure 9). Dare to make deeper changes to the program. For example, include other cities with different time zones, modify the dial, markers and hands shapes, etc. Let your imagination run wild.

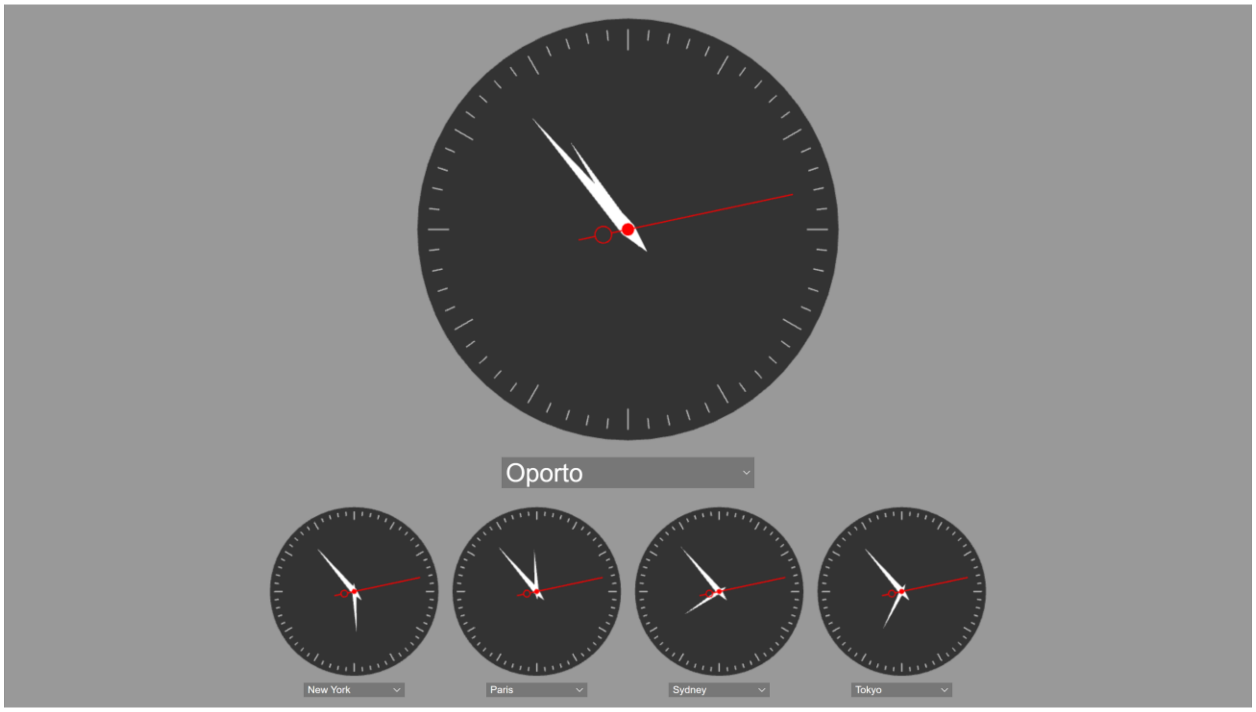

Figure 9 – Another example of an interactive watch

References

[1] Three.js, "Three.js – JavaScript 3D Libray," [Online]. Available: https://threejs.org. [Accessed 25 July 2021].

[2] Three.js, "CircleGeometry," [Online]. Available: https://threejs.org/docs/api/en/geometries/CircleGeometry.html. [Accessed 25 July 2021].

[3] Three.js, "MeshBasicMaterial," [Online]. Available: https://threejs.org/docs/api/en/materials/MeshBasicMaterial.html. [Accessed 25 July 2021].

[4] Three.js, "Mesh," [Online]. Available: https://threejs.org/docs/api/en/objects/Mesh.html. [Accessed 25 July 2021].

[5] Wikipedia, "Circle," [Online]. Available: https://en.wikipedia.org/wiki/Circle. [Accessed 25 July 2021].

[6] Three.js, "Drawing lines," [Online]. Available: https://threejs.org/docs/manual/en/introduction/Drawing-lines.html. [Accessed 25 July 2021].

[7] Three.js, "Vector3," [Online]. Available: https://threejs.org/docs/api/en/math/Vector3.html. [Accessed 25 July 2021].

[8] Three.js, "BufferGeometry," [Online]. Available: https://threejs.org/docs/api/en/core/BufferGeometry.html. [Accessed 25 July 2021].

[9] Three.js, "LineBasicMaterial," [Online]. Available: https://threejs.org/docs/api/en/materials/LineBasicMaterial.html. [Accessed 25 July 2021].

[10] Three.js, "Line," [Online]. Available: https://threejs.org/docs/api/en/objects/Line.html. [Accessed 25 July 2021].

[11] Three.js, "LineSegments," [Online]. Available: https://threejs.org/docs/api/en/objects/LineSegments.html. [Accessed 25 July 2021].

[12] Three.js, "EllipseCurve," [Online]. Available: https://threejs.org/docs/api/en/extras/curves/EllipseCurve.html. [Accessed 25 July 2021].

[13] Three.js, "BufferAttribute," [Online]. Available: https://threejs.org/docs/api/en/core/BufferAttribute.html. [Accessed 25 July 2021].

[14] Three.js, "LineLoop," [Online]. Available: https://threejs.org/docs/api/en/objects/LineLoop.html. [Accessed 03 March 2022].

[15] Three.js, "EdgesGeometry," [Online]. Available: https://threejs.org/docs/api/en/geometries/EdgesGeometry.html. [Accessed 03 March 2022].

[16] Mozilla, "<select>: The HTML Select element," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/select. [Accessed 25 July 2021].

[17] W3Schools, "HTML <select> Tag," [Online]. Available: https://www.w3schools.com/tags/tag_select.asp. [Accessed 25 July 2021].