Implementing Tic-Tac-Toe with Web Standards Technologies - Free Sample Pages

Caleb Josue Ruiz Torres

December 18, 2023

1 Why Tic-Tac-Toe?

Tic-Tac-Toe is a simple game that can be used to showcase the entire set of ingredients required to implement a video game and how web technologies interact among them.

2 Considerations.

The implementation we are depicting here it is not optimized to always win or draw, what's the point of having users frustrated with no possibilities to draw a smile on their faces after the feeling of a triumph? Still, if the user does not pay sufficient attention it can be defeated easily by the computer.

3 Why Web Standards only?

It is the case that you can implement this game in any programming language you may like or even assisted with a framework, but using web technologies allows us to deliver the final product through a web browser, most users already have one installed. Using Web Standards like HTML, CSS, and JavaScript, keeps the Software stack simple, requiring from you only a text editor. Some testing will be performed using Jasmine which it is not a web standard.

4 Prerequisites.

I am expecting you have gone trough an introductory course on computer programming, i.e. You know what variables are, how loops are implemented, and how to define functions. Code beyond these three constructs will be explained in order of appearance.

5 Source Code.

The source code is available at https://bitbucket.org/calebjosuedotcom/noughts_and_crosses and it is licensed under GPLv3. If you need help navigating the commits please take a look at the following video.

Enough talk. Let's get started with our implementation. Now, the approach we are taking here is a practical one. We shall point you to some web resources in the case you feel like going deeper with any given technology or definition. No questions without answers, send them over to `calebjosue-dev@outlook.com`

6 The Board.

In the hope you know how to play this game or at least you took a look to the URL at the beginning of this document. You now know that we are in need of a board to place our movements.

6.1 HTML Boilerplate.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Your Tic-Tac-Toe</title>
</head>
<body>
    <header>
        <h1>Your Tic-Tac-Toe</h1>
    </header>
    <main>
        <section id="board">
        </section>
    </main>
</body>
</html>
```

Listing 1: index.html - HTML Boilerplate

Listing 1 shows the minimum HTML code to present content for our users, well, in reality you can get away with less than that but it is a good practice to retain the semantic meaning in our documents. Open your favorite text editor and type it into it, you can copy-paste it too. Save the file as **index.html** in the directory you are pretending to exercise this material.

Do you know what HTML is, and why we have written the code the way it is presented? HTML stands for HyperText Markup Language and it is a technology that allows us to write

websites by means of specifying how our content is structured. The sole inclusion of the *Language* word tell us that this is all about communication. We are going to communicate the web browser how to display our content.

<words> within brackets are called *tags* and are the building blocks of our document. These tags happen to convey meaning for the web browser to interpret, e.g The `<h1>` tag does represent a header text, it will stand out from the other present text. You need to close most tags you open, for example `</h1>` will close the just mentioned `<h1>` tag. Double click the **index.html** file to see the results in your web browser. You should see something similar to the next image.

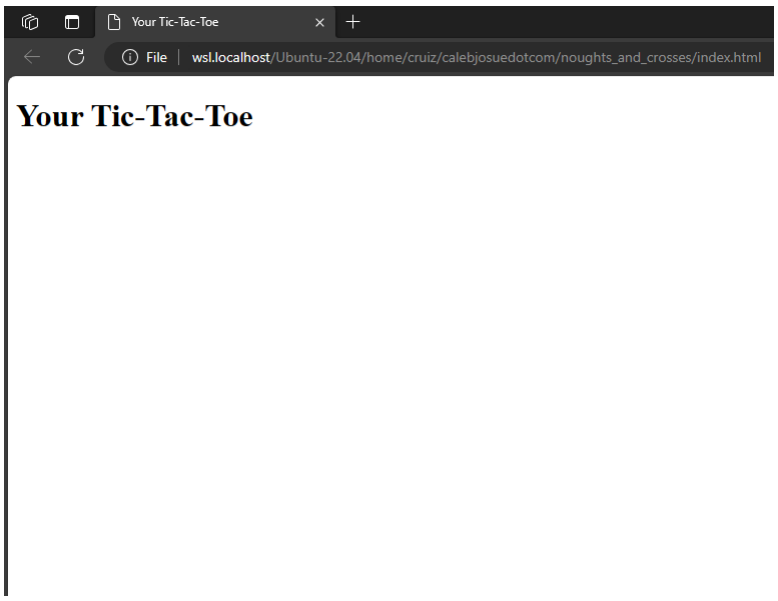

Figure 1: index.html bare-bones

How would you know which tags to close and which ones not to close. Well, you will need to take a deep dive into an HTML course, there are lots of them over the Internet. And this is not one of them, we are only using the minimum set of tags for our web application to work. And what about the other tags present you may ask? Do not let them intimidate you or to stop you from achieving your ultimate goal which is to present a game for your users. Still, at some point you will need to accept that some tags are just meant to be there in order to have well-formed document, e.g. <DOCTYPE>. And from this very moment you can also experiment with this basic set up, try changing the text between the `<h1>` or the <title> tags for example by putting your name instead.

The results of your recent changes can be visualized by first saving the file you are editing and then heading to the web browser and click the circular arrow icon to the right of the URL address bar. Or press `Ctrl + R` if you are on Windows.

6.2 Drawing the board.

Let us meet another Web Standard, SVG (Scalable Vector Graphics. We are going to use it to draw our board, and let's just jump right into it. We are presenting you six lines of code to be typed or copied-pasted inside the <section> opening and closing tags, line of code 13 and 14 in Listing 1.

```
1 <svg id="playground" height="350" width="350">
2     <polyline id="vleft" points="116, 10 116, 340" stroke="green"
    stroke-width="7" />
3     <polyline id="vright" points="232, 10 232, 340" stroke="green"
    stroke-width="7" />
4     <polyline id="htop" points="10, 116 340, 116" stroke="green" stroke-width=
    "7" />
5     <polyline id="hbottom" points="10, 232 340, 232" stroke="green"
    stroke-width="7" />
6 </svg>
```

Listing 2: index.html - Tic-Tac-Toe Board

Refresh the content in your web browser to see the recent changes brought by the six lines of code above.

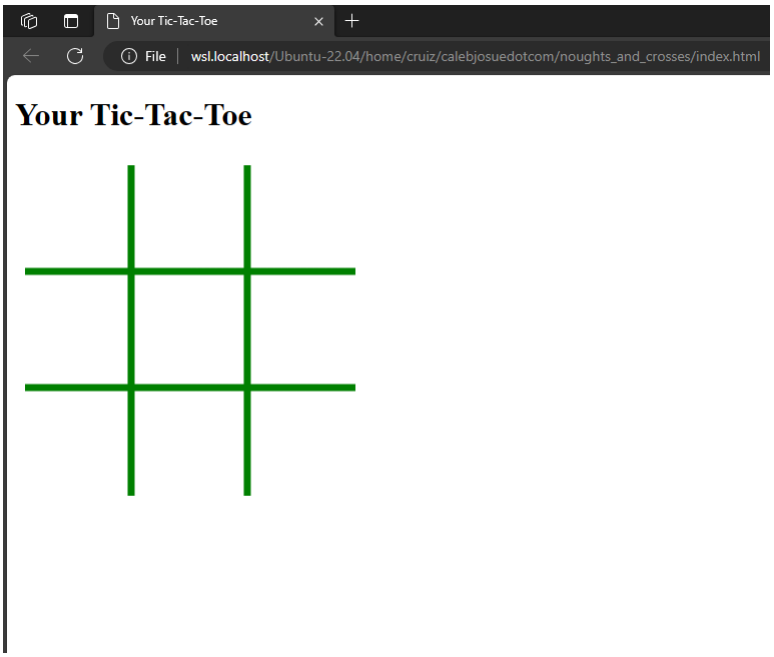

Figure 2: index.html board

What's all that gibberish we have just added anyway? Can you make the lines red color or maybe blue? Of course you can! It is just a matter of changing the color value for the

stroke property. What? Locate the word "green" in Listing 2, and change it to another color you like to see.

Let's start with a less daunting task, let's draw a rectangle. Create a file and name it **mysvg.html**. What content will we be putting inside? The same content as **index.html**, so copy the text inside **index.html** and paste it over **mysvg.html**, now remove the six lines with have added in Listing 2. That is, replace those six lines of code with the following code snippet, it will live inside the <section> opening and closing tags.

```
<svg id="playground" height="350" width="350">
    <rect x="50" y="100" width="150" height="100"
        fill="yellow" stroke="navy" stroke-width="10"  />
</svg>
```

Listing 3: mysvg.html - Rectangle element

Here's how it looks like.

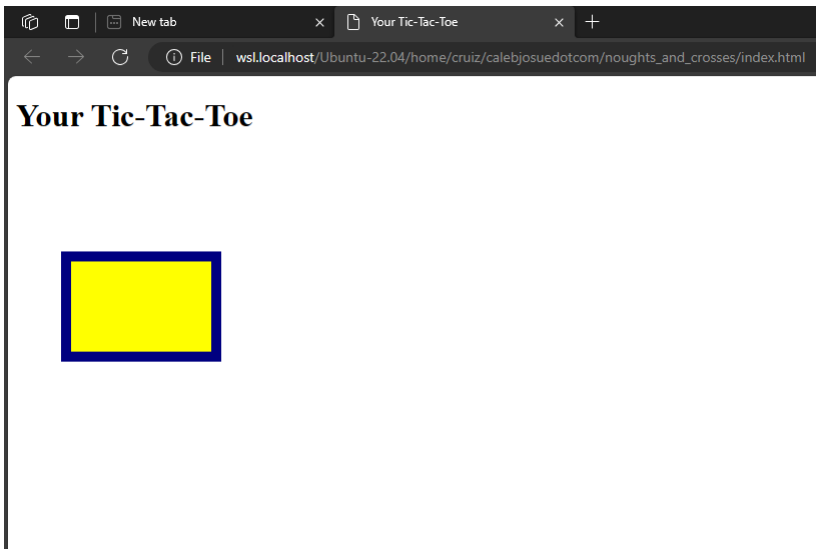

Figure 3: mysvg.html rectangle

At this point it is necessary to add more vocabulary to our discussion. Every tag defines what's called an *HTML element*. Thus, <h1> defines an HTML element in our document. Our entire HTML document contains several HTML elements. Elements have *attributes*, every *attribute* will give more information to the web browser about any given HTML element you are defining.

SVG is a language too! And as you already saw, it can be mixed with HTML. That is, you can add SVG elements to your HTML documents. We just did that.