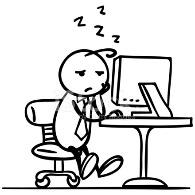
We have gone through some of the basic functionality of jQuery, now it’s time for us to have some hands-on experience!

Before delving deep into the project, let’s briefly discuss how HTML5, CSS and JavaScript (jQuery) become the main stream for web development.

Back in year 1996, Adobe Flash was introduced and it quickly became a popular method for adding animation and interactivity to web pages. Developers used it to create interactive webpages and browser-based game which is also known as Flash game. With the launch of HTML5, mobile has emerged as one of the publishing world’s biggest advantages. HTML5, CSS and JavaScript are widely adopted by the major web browser and are compatible with touch screens. Flash on the other hand, was designed for desktop, it consumes more power causes the device to heat up easily. Hence developers quickly adopted HTML5, CSS and JavaScript taking advantage of their efficiency and cross-platform compatibility. Today, although Flash websites still exist, they are mostly in the process of shifting towards the modern web technologies. A very good example will be Scratch, a block programming platform to learn programming, is launching a Beta version for Scratch 3.0 which is built upon the modern web technologies and will be fully releasing the latest version of Scratch in January 2019. Unlike its predecessor, Scratch 3.0 does not require Flash player plugins and is designed for touch-screen devices.

After the lengthy and boring historical explanation, now it is time for us to jump right into the project which you must have been waiting for.



We are going to take advantage of jQuery cross-platform compatibility, to create our very own Snake Game, which is not only **playable on** a computer but also on your **TOUCH-SCREEN MOBILE DEVICES!!!**

**Hands-on Project: Snake Game**

Download and unzip the folder [], go to the “Snake Game”, you should be able to see 3 files and 1 folder namely “snake.html”, “stylesheet.css”, “script.js” and “snake\_arrow”.

Open the snake.html with your favorite text editor and you will see the following code.



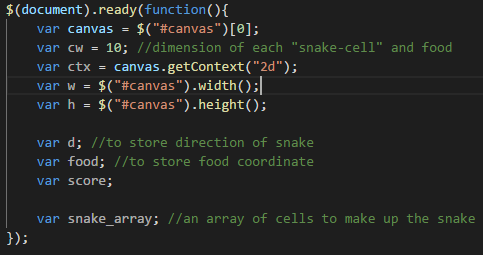
In the <head> tag, we included jQuery, and our own CSS and JavaScript file. Meanwhile in the <body> tag, we have a container containing **canvas** object. Canvas is the place where our game will go.

In our own CSS file, stylesheet.css, will be some size and positioning definition of the DOM in our html. (If you would like to know more, please refer to this awesome website <https://www.w3schools.com/css/>)

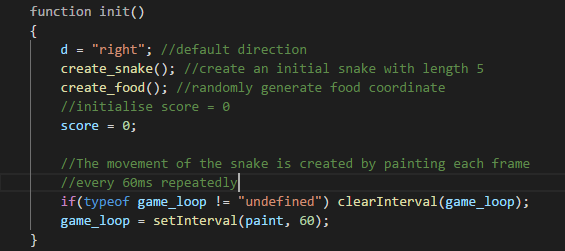
Now open style.js with your favorite text editor, we start a jQuery program with the following line:

This document ready event prevents any jQuery code from running before the webpage is fully loaded.

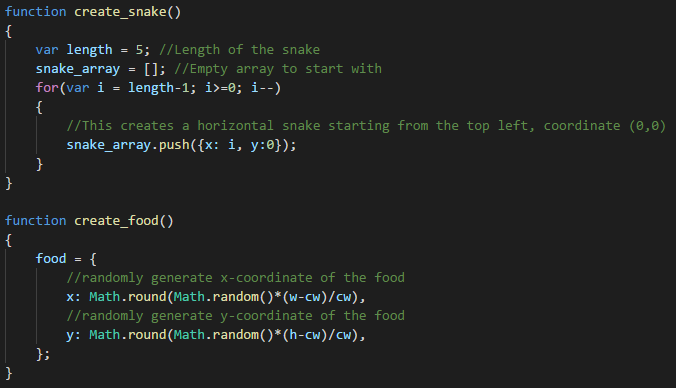
Add the following lines.



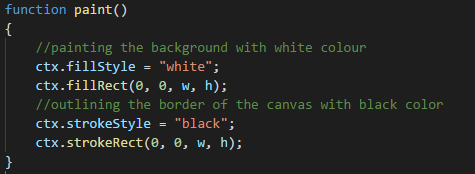
Here we declared some variable for later use. The variable canvas is a variable that store the canvas DOM in our html file. The ctx variable on the other hand contains the context identifier defining the drawing context associated to the canvas. Here we use jQuery id selector $(“#canvas”) to select the html tag with id = “canvas”.



Then we add a function init() which will initialize a new game whenever we start the game or game is over. The movement of the snake is created by painting frame by frame every 60ms.



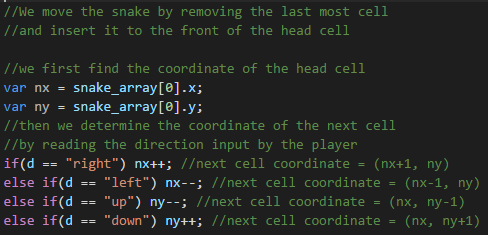
We now create two function, where create\_snake will generate a new snake of length 5 whereas create\_food will generate new coordinate for the food.



We will then work on the function paint, which will be called every 60ms. We start by painting the background white and outlining the canvas.

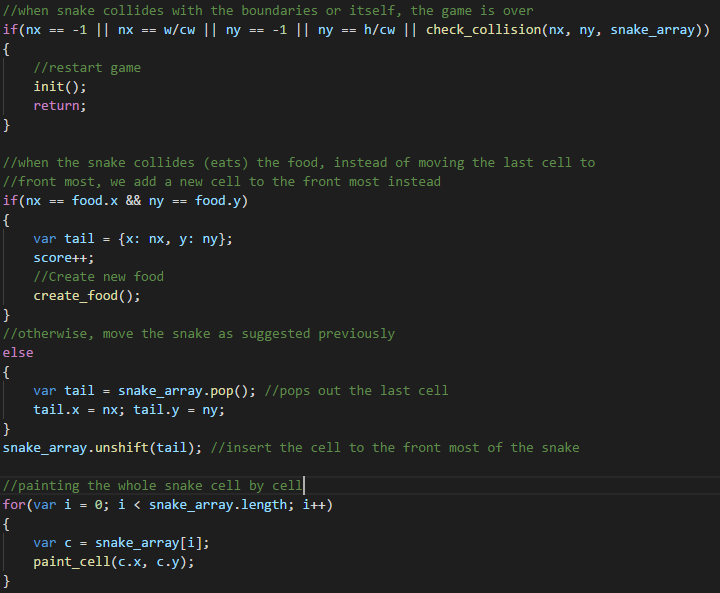
Next we will be looking into how we are going to move the snake.

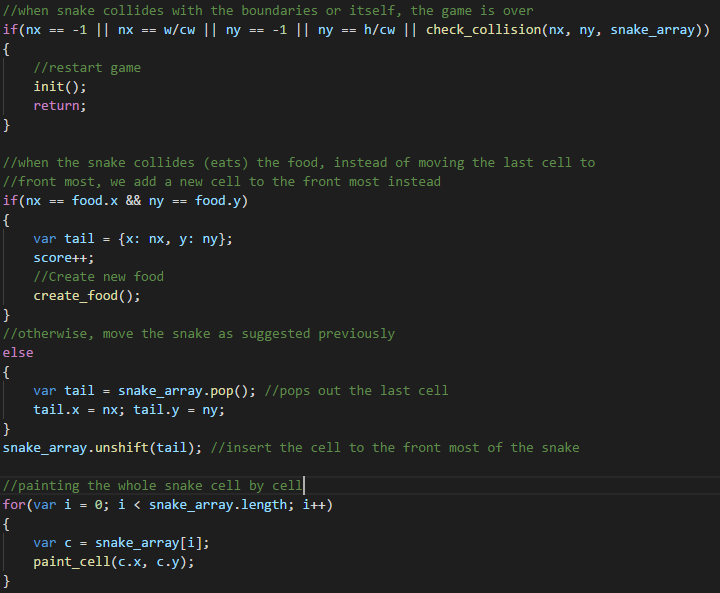
We move the snake simply by taking the last cell of snake and move it to the front of that current head cell. Add the following code to paint() function.



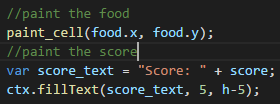
Here, we determine the coordinate of where the snake should head from the user key input.

We will now define how the snake should behave when it collides with boundaries or itself and when the snake eats the food. Continue by adding the following code to the paint function.

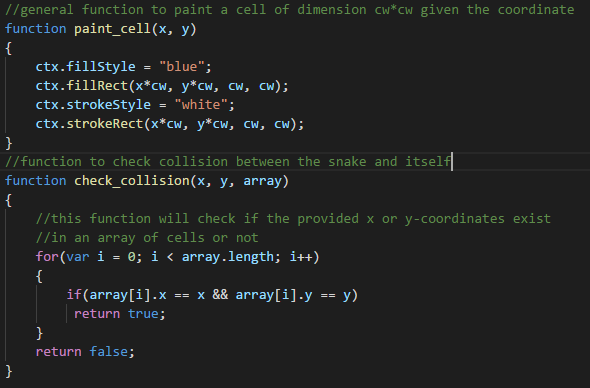




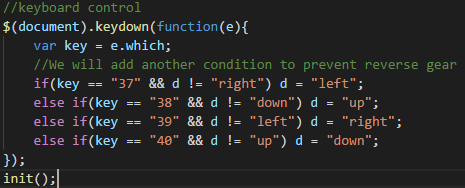
When the snake collides with boundaries or itself, game is over and a new game is initiated. The length of the snake is increased by one when it eats the food. Then we paint out the whole snake cell by cell.



We end the paint() function by painting the food and scores on the canvas.



We then add two new functions which will paint a new cell given the coordinate and the other will check for collision of the snake with itself.

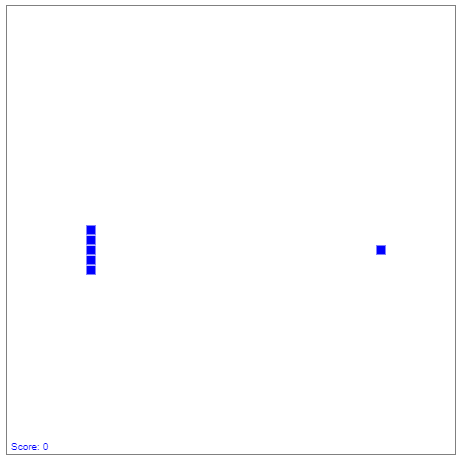


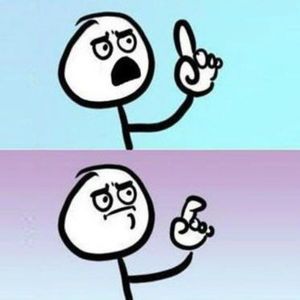
Last but not least we will add the keyboard control to the snake. The function will be triggered when a keyboard key is pressed down, then the function will check for the key which is being pressed down, if the key is one of the arrow keys, corresponding action will be taken by changing the direction of d. We then call the init() function to start the game.

We are now done with the code. Now let’s try to run our very own snake game created with HTML and jQuery.

To run our code, right click on the snake.html, and open it with you preferred browser.

Tadaa!!! You have successfully your Snake Game that can be played on a browser.





Wait a minute…

How can we the Snake Game be mobile compatible? Are we going to connect a keyboard to our mobile devices just to play the game?

Well definitely not! Our project is not yet done, let’s carry on.

In order for us to control the snake one touch-screen devices without using keyboard, we will need to have arrow key on the screen. Browse to the folder “snake\_arrow”, you will be able to see 5 arrow images. We will add the arrow images into our HTML.

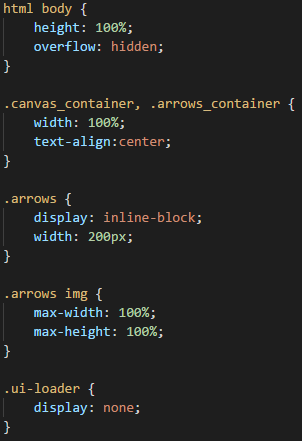
Now open the snake.html with text editor, add the following lines into the <body> tag.



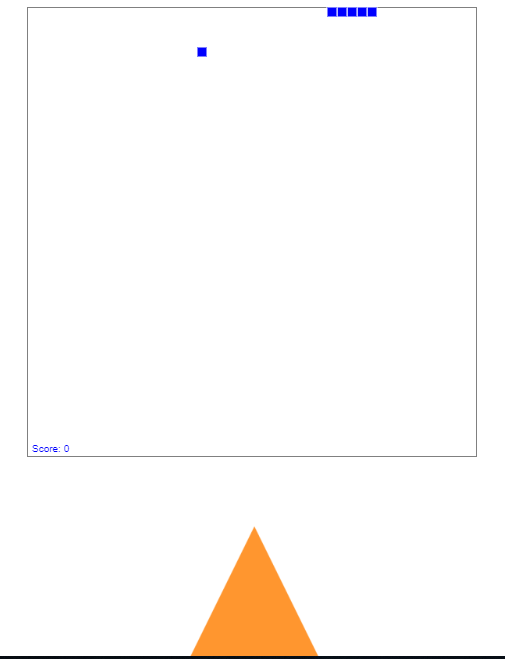


In the <img> tags, the src attribute is used to state the location of the image. All the images are located in the folder “snake\_arrow”.

Then open the stylesheet.css make the changes accordingly.

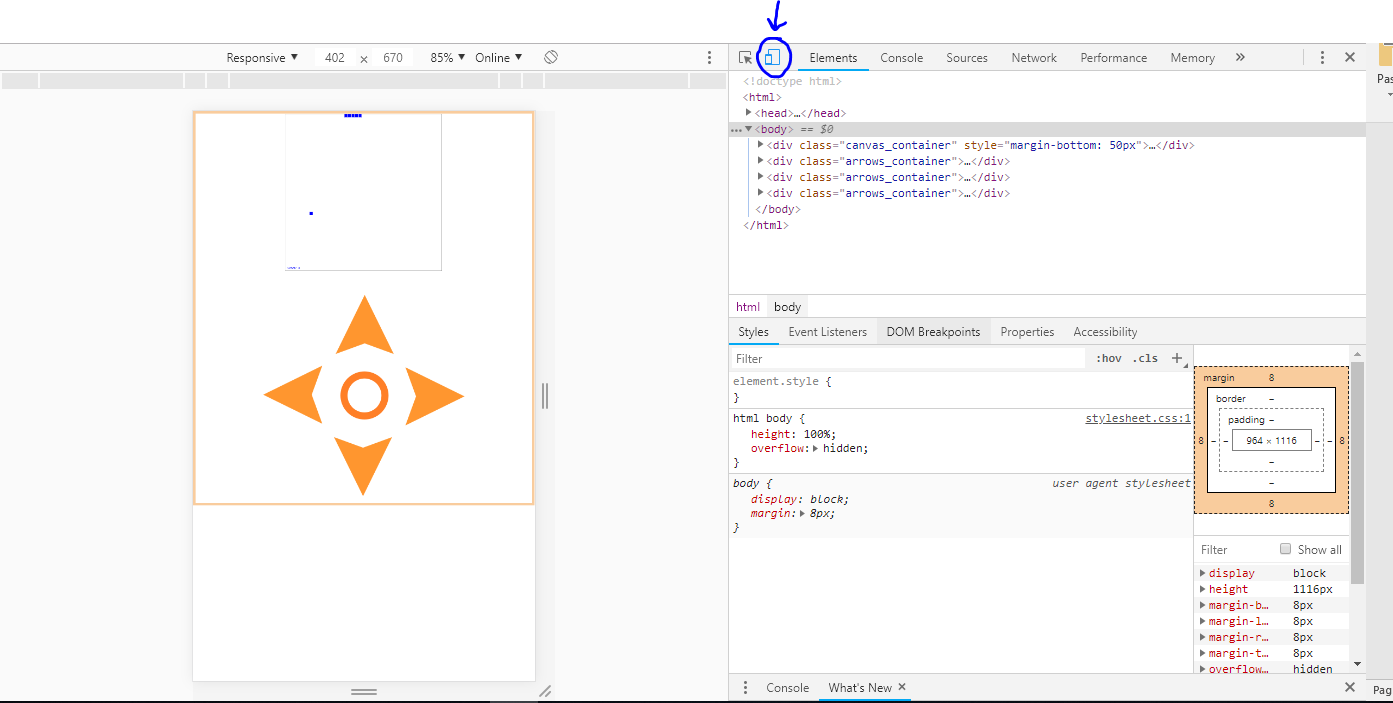


Open the snake.html with your preferred browser and noticed the change:



Well, the result is not really what we have been expecting. This is because we size the arrows such that it is suitable for touch-screen device, hence if you are viewing the program on a computer, the images size might not be appropriate. Nevertheless, if you see changes as compared to the previous one you should be good to proceed.

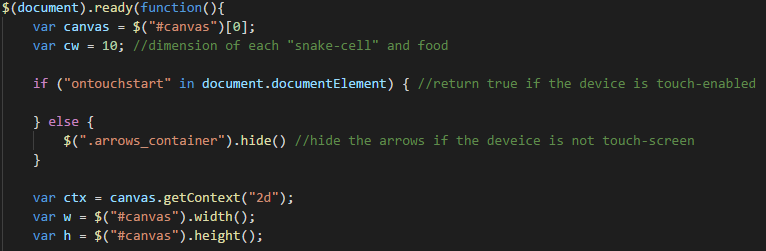
However, if you are using Google Chrome to run the html file, you can press on the key F12, and click on the circled icon below.



Now you should be able to simulate the browser as if it is being run from a mobile device, the arrow keys size is just nice ☺ !

So we are now done with the user interface part. We will now proceed with the JavaScript part.

Open script.js with text editor. We do not want the arrow keys to show up when we are not using touch-screen device, hence we will need a way to detect whether the device is touch-enabled and show or hide the arrow keys accordingly.



“ontouchstart” in document.documentElement will return true if the device is touch-enabled. Here we use jQuery class selector to select all the arrows\_container and hide all of them on computer browser.

Aside from the arrow keys, we will also need to make adjustment to the canvas size such that it fits better to mobile device screen.

Add the following line to the part when the device is touch-enabled.



We increase the canvas width, canvas height and cell dimension by 150% from the original.

Now if you open the snake.html on a web browser from a computer, you will notice that the arrow keys are no longer there ~

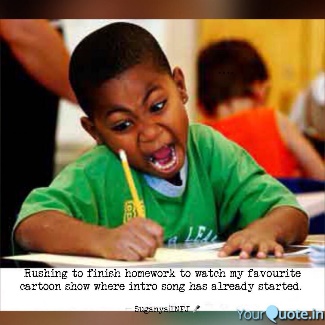
To make the arrow keys responsive to mobile tap, we will now include the jQuery mobile, jQuery that is specially designed for mobile devices. Add the following line into <head> tag in snake.html.



Also notice that each arrow keys are associated with a special id attribute, we are going to use their respective id to select each of them and capture the event when they are tapped using jQuery mobile.



Add the code above into script.js. The example above shows how you can use jQuery to target element with specific id and capture the tap event using the element.on(“tap”, func()). In the callback function, we will make appropriate changes to the direction.

The code above only handles the event when arrow key up is tapped, now treat it as a practice, write codes to capture the tap event for the rest of the arrow keys.

Now that when you are done with it, try to open snake.html with web browser. You should not see any changes as the last Snake Game. This is because the added code will only work on mobile devices.