



Basics of Creating a Google map on your page

Creating google maps on your web page does not have to be confusing or difficult, as with any other thing in coding it is all about one piece of code acting upon another.

The difficulties arise, when large scale Copy – Paste and Pray happens!

I've done it myself, you hunt for a solution online, see something that looks close and pop it in, it seems to work but not the way you wanted, and trying to get it to work is a frustrating and ultimately demoralising experience.

The big problem here is you are looking for a needle in someone else's haystack, there are multiple ways to create a map and you may be trying to piece together different methods, like trying to weld the front end of a ford onto the backend of a Honda and expect it to pass as a Mercedes.

Anyway, what this document does, is show you:

- How to add a map.
- Add a marker.
- Add multiple markers.
- Add an information window.

Each stage has screenshots of working code, and each line is commented with a full explanation.

But, before you begin:

- Goto <https://developers.google.com/maps/gmp-get-started> and follow the instructions to get your own API Key.



Basics of Creating a Google map on your page

Stage 1 – Create a Map

1. Setup your Html with a map container and your map api key:

The screenshot shows a Visual Studio Code interface with a dark theme. The left sidebar has icons for file operations like Open, Save, Find, and Copy/Paste. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The title bar indicates the file is "map.html - customize-website - Visual Studio Code". The left sidebar shows a file tree with files: "mapjs" (2), "maps_various_ways.js", "map.html" (marked with a dot), and "map.css". The main editor area contains the following HTML code:

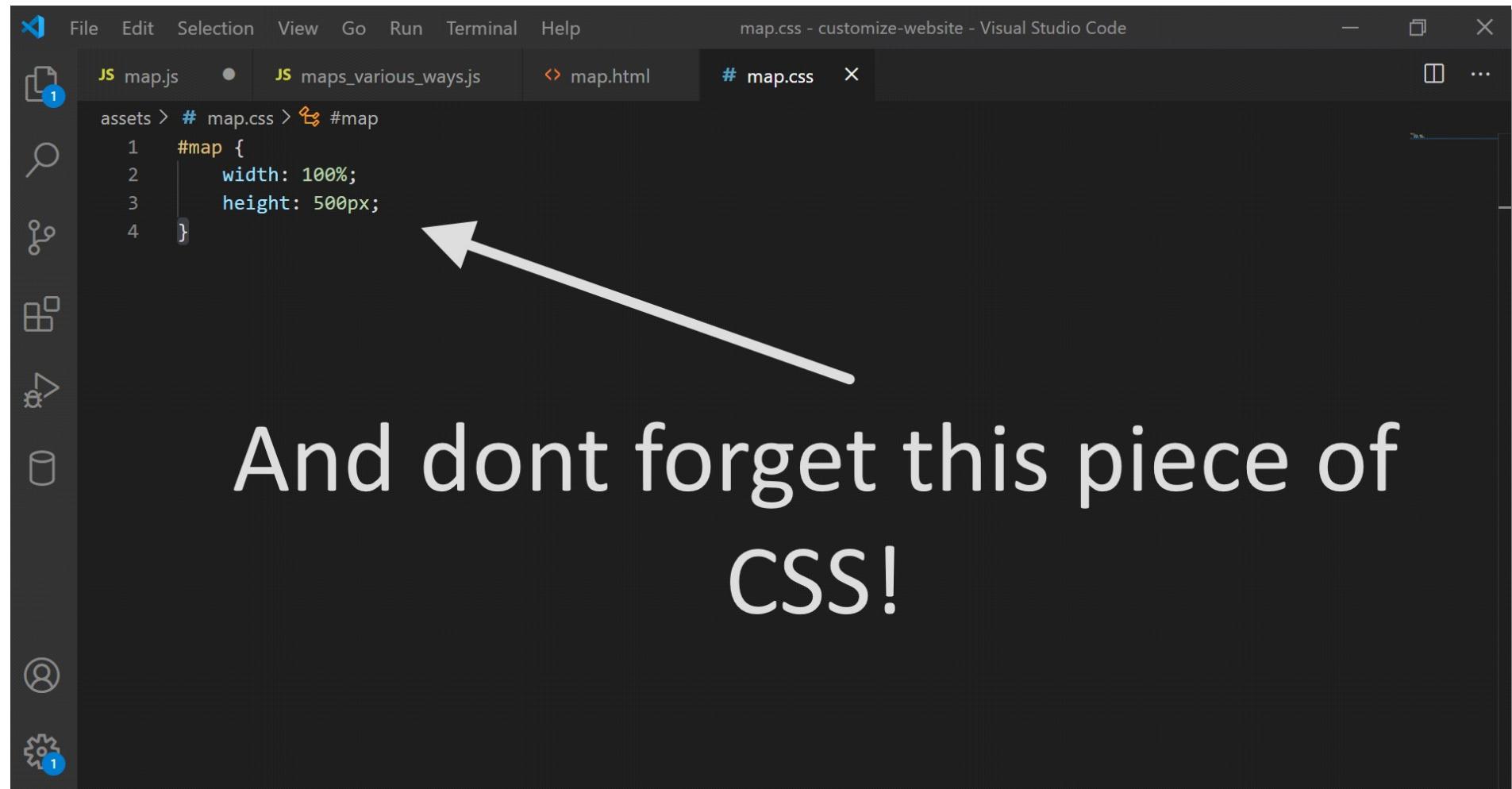
```
File Edit Selection View Go Run Terminal Help
● map.html - customize-website - Visual Studio Code
mapjs ● maps_various_ways.js • map.html • map.css
assets > map.html > html > body > script
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5    <meta charset="UTF-8" />
6    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7    <link rel="stylesheet" href="map.css" />
8    <title>Map Example</title>
9  </head>
10
11 <body>
12
13
14 <div id="map" class="box-shadow"></div>
15
16
17 <script src="https://maps.googleapis.com/maps/api/js?libraries=localContext&v=beta&key= Your Key Here
18 <script src="map.js"></script>
19 </body>
20
21 </html>
22
```

The line "Your Key Here" is highlighted in yellow, indicating where the user should enter their own Google API key.



Basics of Creating a Google map on your page

2. Dont forget your CSS, and make sure it is linked in your HTML:



A screenshot of the Visual Studio Code interface. The title bar says "map.css - customize-website - Visual Studio Code". The left sidebar has icons for files, search, symbols, and other development tools. The main editor area shows a CSS file with the following code:

```
assets > # map.css > #map
1  #map {
2    width: 100%;
3    height: 500px;
4 }
```

A large white arrow points from the text "And dont forget this piece of CSS!" below to the "#map" selector in the CSS code.

And dont forget this piece of CSS!



Basics of Creating a Google map on your page

3. Setup a JS file, and make sure it is linked in your html:



Basics of Creating a Google map on your page

4. Begin your map – setup a map function:

```
File Edit Selection View Go Run Terminal Help
● map.js • map.html # map.css
assets > JS map.js > ...
1
2  /** DECLARE A FUNCTION TO CREATE THE MAP */
3  function myMap() {
4
5  }
6 |
```



Basics of Creating a Google map on your page

5. Setup a mapProp (call it what you like) – but this will contain the properties of your map.

A screenshot of the Visual Studio Code interface. The title bar shows "map.js - customize-website - Visual Studio Code". The left sidebar has icons for File, Edit, Selection, View, Go, Run, Terminal, Help, and a search function. The main editor area shows the following code:

```
1  /** DECLARE A FUNCTION TO CREATE THE MAP */
2  function myMap() {
3
4      /** THE mapProp VARIABLE DEFINES THE PROPERTIES OF THE MAP. */
5      const mapProp = {}
6
7  }
```

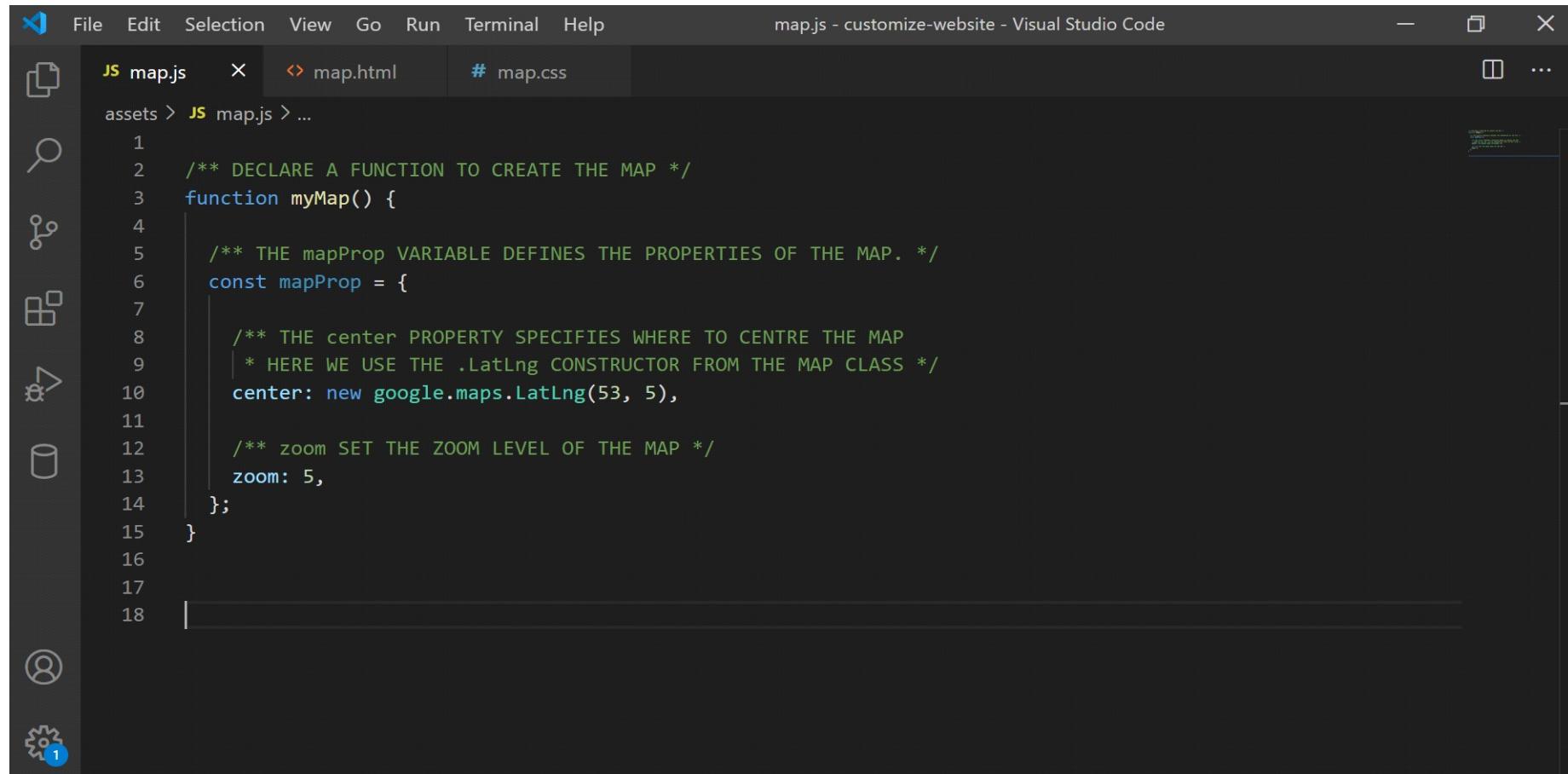
The code is written in JavaScript, defining a function named `myMap`. Inside the function, a variable `mapProp` is declared and set to an empty object. The code uses JSDoc-style comments to describe the purpose of the function and the variable.



Basics of Creating a Google map on your page

6. Now add the properties you want for your map.

- On line 10 below i am calling a new instance of the .LatLng google constructor and i am passing into this constructor the latitude and longitude i want for the center of my map.
All this constructor does is inform that the first number is lat and the 2nd in lng. To see another way of doing this – see step 7 below.
- On line 13 – i set up a default zoom of 5.



The screenshot shows a Visual Studio Code interface with a dark theme. The left sidebar has icons for file operations like Open, Save, Find, and Run. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The title bar says "map.js - customize-website - Visual Studio Code". The main editor area contains the following JavaScript code:

```
File Edit Selection View Go Run Terminal Help
map.js - customize-website - Visual Studio Code
JS map.js X map.html # map.css
assets > JS map.js > ...
1
2  /** DECLARE A FUNCTION TO CREATE THE MAP */
3  function myMap() {
4
5      /** THE mapProp VARIABLE DEFINES THE PROPERTIES OF THE MAP. */
6      const mapProp = {
7
8          /** THE center PROPERTY SPECIFIES WHERE TO CENTRE THE MAP
9             * HERE WE USE THE .LatLng CONSTRUCTOR FROM THE MAP CLASS */
10         center: new google.maps.LatLng(53, 5),
11
12         /** zoom SET THE ZOOM LEVEL OF THE MAP */
13         zoom: 5,
14     };
15 }
16
17
18 |
```



Basics of Creating a Google map on your page

7. As an option to above number 6, you can create a variable (line-2 - below) and link your center to that properties value – both methods will work.

```
File Edit Selection View Go Run Terminal Help
map.js - customize-website - Visual Studio Code
JS map.js X map.html # map.css
assets > JS map.js > ...
1
2 const properties = {"lat": 12, "lng": 100}
3
4 /** DECLARE A FUNCTION TO CREATE THE MAP */
5 function myMap() {
6
7     /** THE mapProp VARIABLE DEFINES THE PROPERTIES OF THE MAP. */
8     const mapProp = {
9
10         /** THE center PROPERTY SPECIFIES WHERE TO CENTRE THE MAP
11         | * OR PASS IN THE COORDINATES FROM AN EXTERNAL VARIABLE/FUNCTION */
12         center: properties,
13
14         /** zoom SET THE ZOOM LEVEL OF THE MAP */
15         zoom: 5,
16     };
17 }
18
```



Basics of Creating a Google map on your page

8. Create the map!

- Line 19 creates the map, it calls a new instance of the google maps constructor, and gets the #map id from the HTML. It then calls the properties to find out how you want the map to render. This is all assigned to the variable map.

The screenshot shows a Visual Studio Code interface with a dark theme. The left sidebar has icons for file operations like Open, Save, Find, and Run. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The title bar says "map.js - customize-website - Visual Studio Code". The main editor area contains the following code:

```
File Edit Selection View Go Run Terminal Help
● map.js • map.html # map.css
assets > JS map.js > myMap
1
2 const properties = {"lat": 12, "lng": 100}
3
4 /** DECLARE A FUNCTION TO CREATE THE MAP */
5 function myMap() {
6
7     /** THE mapProp VARIABLE DEFINES THE PROPERTIES OF THE MAP. */
8     const mapProp = {
9
10         /** THE center PROPERTY SPECIFIES WHERE TO CENTRE THE MAP */
11         center: properties,
12
13         /** zoom SET THE ZOOM LEVEL OF THE MAP */
14         zoom: 5,
15     };
16
17     /** CREATES A NEW MAP INSIDE THE DIV THAT HAS THE ID OF "map" SET
18     * IT THEN CALLS THE PROPERTIES TO SEE HOW TO RENDER THE MAP */
19     const map = new google.maps.Map(document.getElementById("map"), mapProp);
20
21 }
```

A yellow highlight is placed over the line of code: "const map = new google.maps.Map(document.getElementById("map"), mapProp);".



Basics of Creating a Google map on your page

9. Now call the map function.

The screenshot shows a Visual Studio Code interface with a dark theme. The left sidebar has icons for search, file operations, and other development tools. The top bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help menus. The title bar says "map.js - customize-website - Visual Studio Code". The main editor area contains the following JavaScript code:

```
const mapProp = {  
    /** THE center PROPERTY SPECIFIES WHERE TO CENTRE THE MAP */  
    center: properties,  
  
    /** zoom SET THE ZOOM LEVEL OF THE MAP */  
    zoom: 5,  
};  
  
/** CREATES A NEW MAP INSIDE THE DIV THAT HAS THE ID OF "map" SET  
 * IT THEN CALLS THE PROPERTIES TO SEE HOW TO RENDER THE MAP */  
const map = new google.maps.Map(document.getElementById("map"), mapProp);  
  
/** THIS IS JUST HOW I CALL THE MAP */  
myMap()  
myMap()
```

A yellow highlight bar is visible under the word "myMap" in the last two lines of the code.



Basics of Creating a Google map on your page

10. View your beautiful map:





There are many other
ways to do this..
And this is one of the problems
people have.

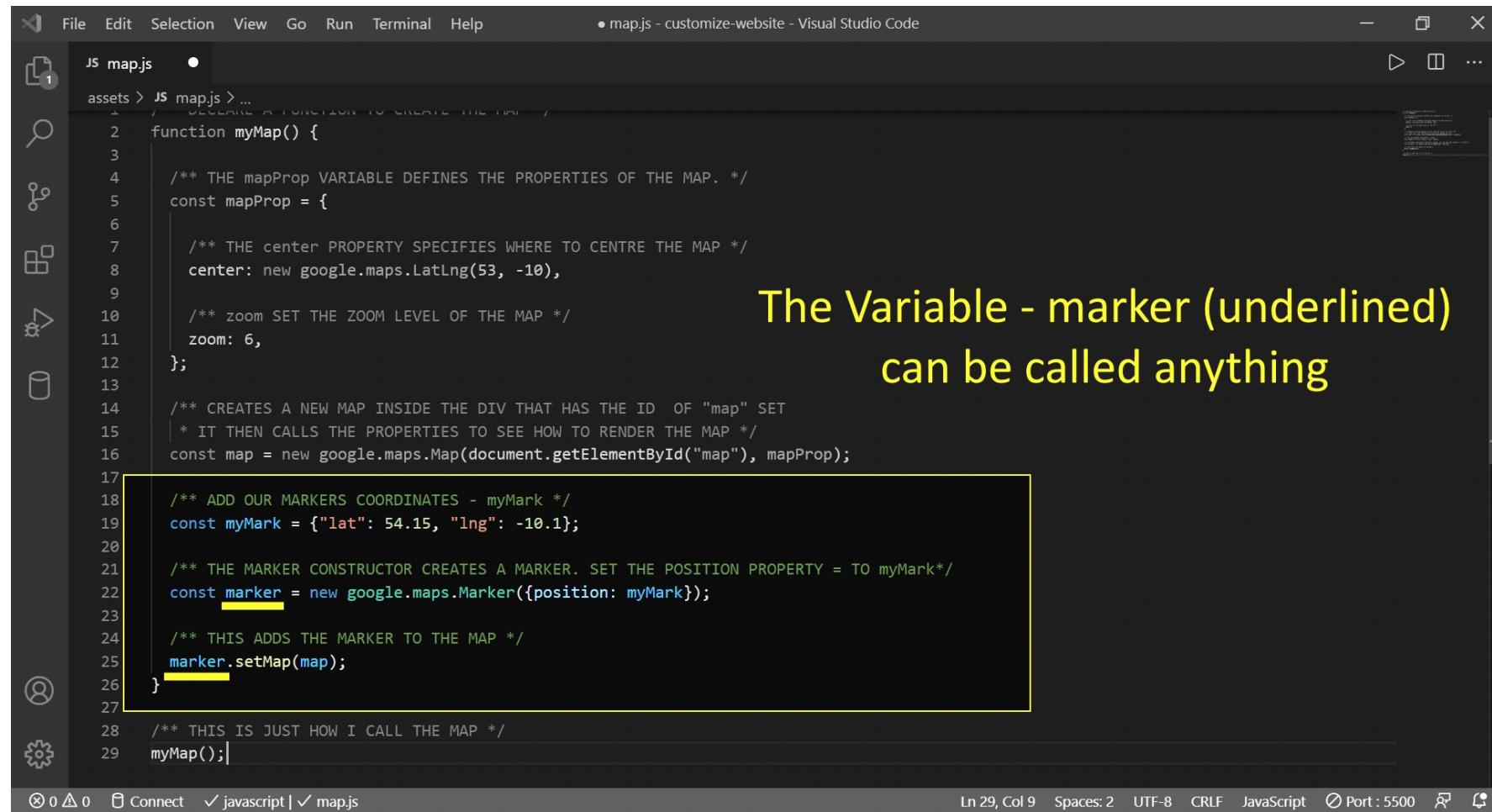


Basics of Creating a Google map on your page

Stage 2 – Create and add Markers

1. Underneath where we created our map, in the last stage.

- Create a single marker



```
File Edit Selection View Go Run Terminal Help • map.js - customize-website - Visual Studio Code
JS map.js
assets > JS map.js > ...
1  // USE THIS FUNCTION TO CREATE THE MAP
2  function myMap() {
3
4      /** THE mapProp VARIABLE DEFINES THE PROPERTIES OF THE MAP. */
5      const mapProp = {
6
7          /** THE center PROPERTY SPECIFIES WHERE TO CENTRE THE MAP */
8          center: new google.maps.LatLng(53, -10),
9
10         /** zoom SET THE ZOOM LEVEL OF THE MAP */
11         zoom: 6,
12     };
13
14     /** CREATES A NEW MAP INSIDE THE DIV THAT HAS THE ID OF "map" SET
15     * IT THEN CALLS THE PROPERTIES TO SEE HOW TO RENDER THE MAP */
16     const map = new google.maps.Map(document.getElementById("map"), mapProp);
17
18     /** ADD OUR MARKERS COORDINATES - myMark */
19     const myMark = {"lat": 54.15, "lng": -10.1};
20
21     /** THE MARKER CONSTRUCTOR CREATES A MARKER. SET THE POSITION PROPERTY = TO myMark*/
22     const marker = new google.maps.Marker({position: myMark});
23
24     /** THIS ADDS THE MARKER TO THE MAP */
25     marker.setMap(map);
26 }
27
28 /** THIS IS JUST HOW I CALL THE MAP */
29 myMap();
```

The Variable - marker (underlined)
can be called anything

Ln 29, Col 9 Spaces: 2 UTF-8 CRLF JavaScript Port : 5500 ⌂ ⌃



Basics of Creating a Google map on your page

2. Underneath where we created our map, in the last stage.

- Create a multiple markers – Each object in the array of objects gets inserted as we loop over the myMarks array. Each object is in the perfect format for inserting using the .Marker class.

```
File Edit Selection View Go Run Terminal Help map.js - customize-website - Visual Studio Code
JS map.js X
assets > JS map.js > myMap > myMarks
7
8    /** THE center PROPERTY SPECIFIES WHERE TO CENTRE THE MAP */
9    center: new google.maps.LatLng(53, -10),
10
11   /** zoom SET THE ZOOM LEVEL OF THE MAP */
12   zoom: 6,
13 };
14
15 /** CREATES A NEW MAP INSIDE THE DIV THAT HAS THE ID OF "map" SET
16 * IT THEN CALLS THE PROPERTIES TO SEE HOW TO RENDER THE MAP */
17 const map = new google.maps.Map(document.getElementById("map"), mapProp);
18
19 /** ADD OUR MARKERS COORDINATES TO myMarks OBJECT */
20 const myMarks = [{"lat": 53.33, "lng": -6.24,},
21   {"lat": 51.5, "lng": -0.1,},
22   {"lat": 53.40, "lng": -2.99,},
23   {"lat": 51.466, "lng": -9.73,},
24 ];
25
26 /** USING SIMILAR CODE TO BEFORE WE CAN LOOP OVER myMark */
27 for(let i=0; i < myMarks.length; i++) {
28   /** EXCEPT THIS TIME WE ADD IN THAT i AFTER MY MARK AND PASS IN THE LAT, LNG OBJECT */
29   const marker = new google.maps.Marker({position: myMarks[i]});
30   marker.setMap(map);
31 }
32
33 /** THIS IS JUST HOW I CALL THE MAP */
34 myMap();
35
```

Ln 21, Col 48 Spaces: 2 UTF-8 CRLF JavaScript ⚡ Port : 5500 ⌂ ⌂



Basics of Creating a Google map on your page

3. Add a title to the marker.

- We alter the code here slightly on line 30. The object(myMark) as it was, was perfect for just inserting using the .Marker class. But now myMarks is not only a lat, lng object, because we have added a name key, which we will target for the title of the marker. So now we will use the LatLng constructor – which accepts 2 values – a lat, and a lng. And as we loop over myMarks we can simply use dot notation to point to the key that we need the value of.

```
File Edit Selection View Go Run Terminal Help
map.js - customize-website - Visual Studio Code
JS map.js X
assets > JS map.js > myMap
14
15  /**
16   * Creates a new map inside the div that has the id of "map" set
17   * It then calls the properties to see how to render the map
18  const map = new google.maps.Map(document.getElementById("map"), mapProp);
19
20  /**
21   * Add our markers coordinates and title name of location
22  const myMarks = [{lat: 53.33, "lng": -6.24, "name": "Dublin"}, 
23    {"lat": 51.5, "lng": -0.1, "name": "London"}, 
24    {"lat": 53.4, "lng": -2.99, "name": "Liverpool"}, 
25    {"lat": 51.466, "lng": -9.73, "name": "Crookhaven"}];
26
27  /**
28   * We loop over the new myMark array of objects
29  for(let i=0; i < myMarks.length; i++) {
30    /**
31     * Except this time we add in that i after myMark
32     * const marker (below) is not needed here - but JSHINT does not like a new constructor call
33     * when it is not assigned to anything - problem is also that JSHINT also does not
34     * like unused variables, so you can decide in or out */
35    const marker = new google.maps.Marker({
36      position: new google.maps.LatLng(myMarks[i].lat, myMarks[i].lng),
37      map: map,
38      title: myMarks[i].name
39    });
40
41  /**
42   * This is just how I call the map */
43  myMap();
44
```

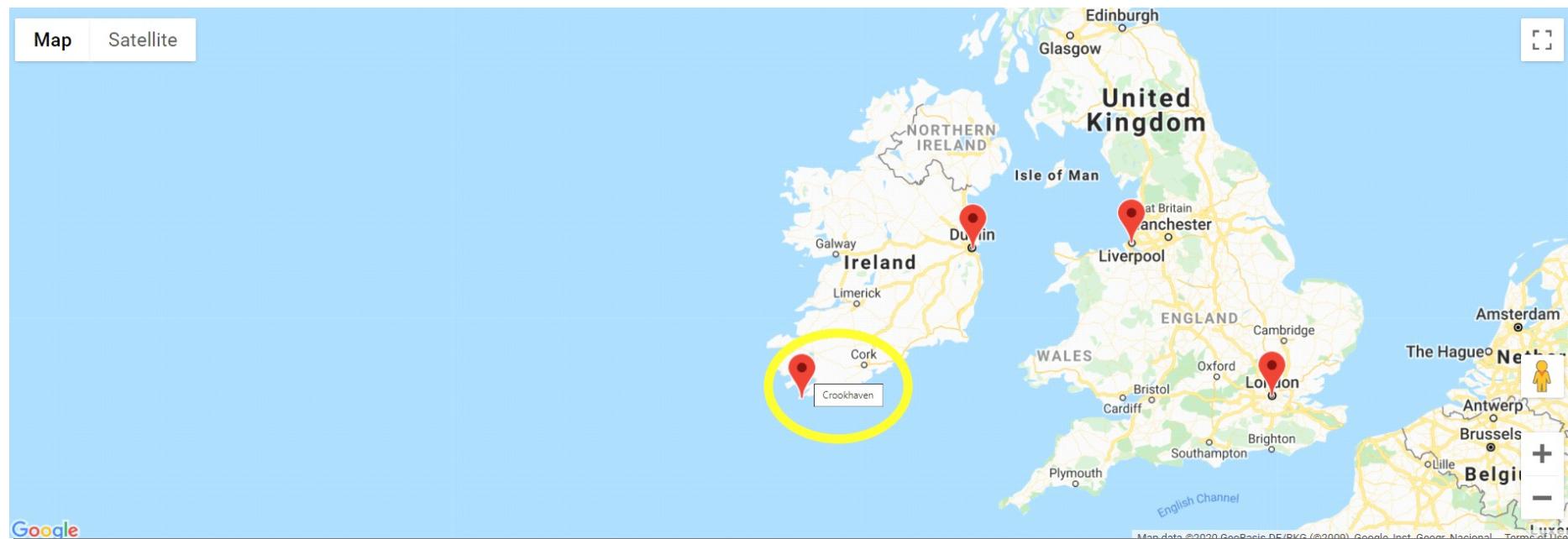
Add the title for each location into the array of objects

We have also altered the code slightly - We are now using the google map constructor LatLng. LatLng can take in two numbers, and use them as Lat and Lng coordinates. So over here we take each iteration of myMarks - achieved by adding [i], then using dot notation point to the object key we require.



Basics of Creating a Google map on your page

This gives us a map with a small title when we hover over the marker:





Basics of Creating a Google map on your page

Stage 3 – Add info to those markers

This will add a small pop up window with information related to each marker, clicking on another marker will close the old marker and open the new one. Or it will be possible to close an info window by clicking the x in the top corner.

As the following code can appear quite complicated a support video is available at: <https://www.youtube.com/watch?v=Xptz0GO2D04> on this topic and it also uses very similar code.

1. First we will update our sample myMarks array of objects, with some information for the info box (use backticks):

```
15  /** ADD OUR MARKERS COORDINATES AND TITLE NAME OF LOCATION AND INFORMATION */
16  const myMarks = [ {"lat": 53.33, "lng": -6.24, "name": "Dublin", "information": `Dublin, capital of the Republic of Ireland,
17  is on Ireland's east coast at the mouth of the River Liffey.
18  Its historic buildings include Dublin Castle, dating to the
19  13th century, and imposing St Patrick's Cathedral, founded
20  in 1191. City parks include landscaped St Stephen's Green and
21  huge Phoenix Park, containing Dublin Zoo. The National Museum
22  of Ireland explores Irish heritage and culture.`},
23  {"lat": 51.5, "lng": -0.1, "name": "London", "information": `London, the capital of England and the United Kingdom,
24  is a 21st-century city with history stretching back to Roman times.
25  At its centre stand the imposing Houses of Parliament, the iconic
26  'Big Ben' clock tower and Westminster Abbey, site of British monarch
27  coronations.`},
28  {"lat": 53.4, "lng": -2.99, "name": "Liverpool", "information": `Liverpool is a maritime city in northwest England,
29  where the River Mersey meets the Irish Sea. A key trade and
30  migration port from the 18th to the early 20th centuries, it's
31  also, famously, the hometown of The Beatles. Ferries cruise the
32  waterfront, where the iconic mercantile buildings known as the
33  "Three Graces" - Royal Liver Building, Cunard Building and Port
34  of Liverpool Building - stand on the Pier Head.`},
35  {"lat": 51.466, "lng": -9.73, "name": "Crookhaven", "information": `Crookhaven is a village in County Cork, Ireland,
36  on the most southwestern tip of the island of Ireland.
37  An out-of-season population of about sixty swells in the
38  summer season to about four hundred, with the occupants of
39  the seasonal holiday homes arriving.`}
40 ];
```



Basics of Creating a Google map on your page

2. Initialize the InfoObj:

- This will store the information in each info window, it is assigned this each time we click on a marker. See step 6.

```
41  /* Initialize an InfoObj - this will store the information displayed in each info window */  
42  var InfoObj = [];  
43  
44
```

3. Open our for loop to loop over the myMarks, just as we did before, and add in the content string which we want to display in our info window. It will reference the information we added in step 1.

```
46  /* LOOP OVER MyMarks - INSIDE THIS LOOP WE WILL CALL OUR MARKERS */  
47  for (i = 0; i < myMarks.length; i++) {  
48    let contentString = '<h3>' + myMarks[i].name + '</h3>' +  
49    '<p>' + myMarks[i].information + '</p>' +  
50    '<a href="https://developers.google.com/maps/documentation/javascript/overview">>' + 'Click me!' + '</a>'  
51
```



Basics of Creating a Google map on your page

4. Call our makers onto the map, we reintroduce our own marker variable on line 44, this is because we need to reference each marker and add an click event listener to it later, and we need something to grab onto.

Also added in an animation effect for the markers, bounce is another common option you could try.

```
52  |  /** I ADD BACK IN THE MARKER VARIABLE AS CALLING THE .OPEN METHOD ON OUR INFOWINDOW VARIABLE
53  |  * NEEDS A REFERENCE TO THE MAP ( SET AT THE START) AND A REFERENCE TO THE MARKER PASSED IN */
54  const marker = new google.maps.Marker({
55
56    position: new google.maps.LatLng(myMarks[i].lat, myMarks[i].lng),
57    map: map,
58    title: myMarks[i].name,
59    animation: google.maps.Animation.DROP,
60  });
```

5. Now we can setup our info window, remember this is all inside our for loop.

- Content string, setup at the start of the loop, is loaded into the variable infowindow.

```
62  |  /** HERE WE SETUP OUR INFO WINDOW, THIS IS WHAT WE CALL   */
63  const infowindow = new google.maps.InfoWindow({
64    content: contentString,
65    maxWidth: 500
66  });
```



Basics of Creating a Google map on your page

6. The remainder of the required code.

- Add an event listener onto each marker in the loop.
- When clicked it calls the function to handle closing all other windows, and resets the window (InfoObj) data.
- Then opens a new window and saves the current infowindow variable content into the InfoObj array.

```
68     /** HERE WE ADD A CLICK LISTENER TO THE MARKER */
69     marker.addListener("click", function () {
70         /* CLOSE OTHER WINDOWS */
71         closeOtherInfo();
72         /* CREATE NEW WINDOW */
73         infowindow.open(map, marker);
74         InfoObj[0] = infowindow;
75     });
76 }
77
78 /** This function clears out any old information */
79 function closeOtherInfo() {
80     if( InfoObj.length > 0 ){
81         InfoObj[0].set("marker", null);
82         InfoObj[0].close();
83         InfoObj[0].length = 0;
84     }
85 }
86 }
87 /** THIS IS JUST HOW I CALL THE MAP */
88 myMap();
```



Basics of Creating a Google map on your page

Completed code

```
/** DECLARE A FUNCTION TO CREATE THE MAP */
function myMap() {
  /** THE mapProp VARIABLE DEFINES THE PROPERTIES OF THE MAP. */
  const mapProp = {
    /** THE center PROPERTY SPECIFIES WHERE TO CENTRE THE MAP */
    center: new google.maps.LatLng(53, -10),

    /** zoom SET THE ZOOM LEVEL OF THE MAP */
    zoom: 4,
  };
  /** CREATES A NEW MAP INSIDE THE DIV THAT HAS THE ID OF "map" SET
   * IT THEN CALLS THE PROPERTIES TO SEE HOW TO RENDER THE MAP */
  const map = new google.maps.Map(document.getElementById("map"), mapProp);

  /** ADD OUR MARKERS COORDINATES AND TITLE NAME OF LOCATION AND INFORMATION */
  const myMarks = [{"lat": 53.33, "lng": -6.24, "name": "Dublin", "information": `Dublin, capital of the Republic of Ireland, is on Ireland's east coast at the mouth of the River Liffey. Its historic buildings include Dublin Castle, dating to the 13th century, and imposing St Patrick's Cathedral, founded in 1191. City parks include landscaped St Stephen's Green and huge Phoenix Park, containing Dublin Zoo. The National Museum of Ireland explores Irish heritage and culture.`},
    {"lat": 51.5, "lng": -0.1, "name": "London", "information": `London, the capital of England and the United Kingdom, is a 21st-century city with history stretching back to Roman times. At its centre stand the imposing Houses of Parliament, the iconic 'Big Ben' clock tower and Westminster Abbey, site of British monarch coronations.`},
    {"lat": 53.4, "lng": -2.99, "name": "Liverpool", "information": `Liverpool is a maritime city in northwest England, where the River Mersey meets the Irish Sea. A key trade and migration port from the 18th to the early 20th centuries, it's also, famously, the hometown of The Beatles. Ferries cruise the waterfront, where the iconic mercantile buildings known as the "Three Graces" – Royal Liver Building, Cunard Building and Port of Liverpool Building – stand on the Pier Head.`},
    {"lat": 51.466, "lng": -9.73, "name": "Crookhaven", "information": `Crookhaven is a village in County Cork, Ireland, on the most southwestern tip of the island of Ireland. An out-of-season population of about sixty swells in the summer season to about four hundred, with the occupants of the seasonal holiday homes arriving.`}
  ];
  /* Initialize an InfoObj - this will store the information displayed in each info window */
  var InfoObj = [];

  /* LOOP OVER MyMarks - INSIDE THIS LOOP WE WILL CALL OUR MARKERS */
  for (i = 0; i < myMarks.length; i++) {
    let contentString = '<h3>' + myMarks[i].name + '</h3>' +
      '<p>' + myMarks[i].information + '</p>' +
      '<a href="https://developers.google.com/maps/documentation/javascript/overview">>' + 'Click me!' + '</a>';
  }
}
```

```
/** I ADD BACK IN THE MARKER VARIABLE AS CALLING THE .OPEN METHOD ON OUR
 * INFOWINDOW VARIABLE
 * NEEDS A REFERENCE TO THE MAP (SET AT THE START) AND A REFERENCE
 * TO THE MARKER PASSED IN */
const marker = new google.maps.Marker({
  position: new google.maps.LatLng(myMarks[i].lat, myMarks[i].lng),
  map: map,
  title: myMarks[i].name,
  animation: google.maps.Animation.DROP,
});

/** HERE WE SETUP OUR INFO WINDOW, THIS IS WHAT WE CALL */
const infowindow = new google.maps.InfoWindow({
  content: contentString,
  maxWidth: 500
});

/** HERE WE ADD A CLICK LISTENER TO THE MARKER */
marker.addListener("click", function () {
  /* CLOSE OTHER WINDOWS */
  closeOtherInfo();
  /* CREATE NEW WINDOW */
  infowindow.open(map, marker);
  InfoObj[0] = infowindow;
});

/** This function clears out any old information */
function closeOtherInfo() {
  if( InfoObj.length > 0 ){
    InfoObj[0].set("marker", null);
    InfoObj[0].close();
    InfoObj[0].length = 0;
  }
}

/** THIS IS JUST HOW I CALL THE MAP */
myMap();
```



Basics of Creating a Google map on your page

I highly recommend following these steps bit by bit, renaming variables and adding in some of your own code, while continually looking at the results of each change you make. Play with it, break it, fix it and add something to it from the extensive google maps library.

In doing this you will gain a much stronger understanding of maps and markers.

I hope this has proved useful to you in trying to make sense of how google maps does what it does.

Online you will find numerous variations on this, but if you are struggling to understand this, i would advise mastering one method, understand it and know it inside out.

Once you have that, Once you know what each line of code is doing, you will then have the confidence and knowledge to chop and change and understand other methods.

Any Questions on what is covered in this document, please let me know in the comments, and i will do my best to answer them.

Written by: Eamonn Smyth