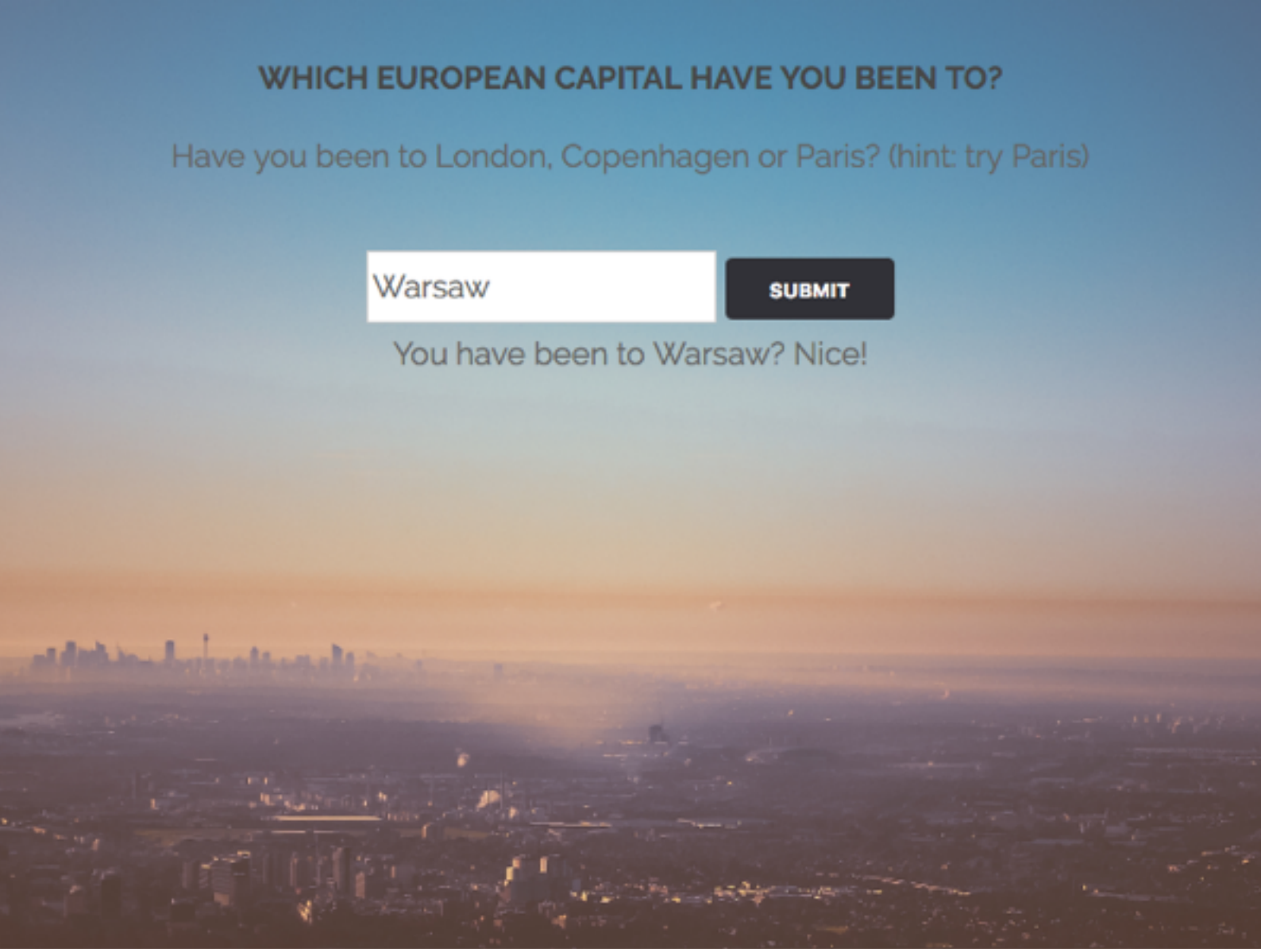

European Capitals - Javascript Autocomplete

In this exercise you will build an autocomplete function which will allow the user to select from a list while typing. A message will also appear when the user has typed in a city.

Dependencies:
Uses the awesomplete.js from http://leaverou.github.io/awesomplete/
Styling inspired by Spatial by TEMPLATED

You will work with:
Javascript
Events
Element & Text nodes
Functions
Conditionals (if / else if / else)

Recommended Knowledge:
HTML
CSS
Basic Programming Principals
Intro to Javascript
The DOM

Level:
Medium

Step 1

Loading the CSS and Javascript files

1) Load the CSS files

Open ***"autocomplete.php"***.
In order to style the page, add a link to the css file called ***"style.css"***.
After that, load the css file called ***"awesomplete.css"***.

Both should be loaded in the <head>.

The <link> tag should be used. E.g.

```
<link rel="stylesheet" type="text/css" href="path/fileName.css">
```

2) Load the Javascript file

At the bottom of the HTML, just before the </body>, add a script source reference to "awesomplete.min.js". This should look something like:

```
<script src="js/awesomplete.min.js"></script>
```

3) Initiate the javascript

Give the <input> element a class name of "awesomplete".

Check if it works!

Resources:

How to load a CSS file
http://www.w3schools.com/css/css_howto.asp
https://helpx.adobe.com/dreamweaver/using/link-external-css-style-sheet.html

How to load a Javascript file:
http://www.w3schools.com/js/js_whereto.asp
http://javascript.info/tutorial/adding-script-html

Step 2

Add a dynamic message that appears when the user has selected a city.

1) Create a paragraph for the messages

Insert a <p> element in which the message can appear. It should be after the input field. Give it an id attribute with the value of "message".

2) Create a submit button

We now need to add a submit button that can be pressed when the user has selected a city.

In between the <p> and the <input>, add a <button> element. It should have an id of "submit".

3) Add event listener to button

On the button element, add an onclick event listener. E.g.

```
<elm onclick="something;">
```

Make the onclick event listener call a javascript function called submitCity().

4) Craft the message

Now it is time to create the javascript function that will get the selected city.

Before the </body> add a <script> tag. Remember to close it.
In this script tag, define a javascript function called "submitCity". It take no parameters.
E.g.

```
function someName() {
}
```

5) Get the input element

Within this function we need to get the selected city from the input field. By using the getElementsByTagName("???") method, it is possible to get a list of all <???> elements in this document.
E.g.

```
document.getElementsByTagName("a")
```

will return a list of all <a> elements.

Use this method to get a list of all <input> elements and assign these to a variable called input. E.g.

```
var input = …….
```

Since the getElementsByTagName method returns an array of all matching elements, we need to limit it to only the first. To get the first element in an array, the [0] index call can be added. This tells the array to return the element that is in position 0, i.e. the first position.
E.g.
```
document.getElementsByTagName("a")[0]
```
would return the first <a> element from the array of all <a> elements in the document.

6) Get the city from input

The input variable now contains the <input> element. Try to log it in the console to see it, i.e. `console.log(input);`

The city that the user has selected is in the input element's value property. So create a new variable called city which is assigned the value property of the input.
E.g.
```
var elm = elm.value;
```

7) Display the message

In order to display the message, we first have to access the text node of the p#message element and assign it the message we want to appear.

Start of by using the getElementById(id) method to get the element. e.g.
```
document.getElementById("name_of_id")
```

Then the element's innerText property can be assigned a new value, which will then update this element's text node. E.g.
```
elm.innerText = "new text value";
```

Finally craft a dynamic message that uses the city variable. E.g.
```
"You have been to "+city+"? Nice!"
```

This message can be assigned to the innerText node of the #message element.

Step 3

Change the design of the page when Paris is selected.

1) Create conditional "if" to check for Paris

Within this function we can check the city the user has selected and make the code do something based upon the choice of city.

A conditional if statement can look like this:

```
if(condition) {
  // This is run if the condition is true
}
```

A condition could look like:

`name === "Hans"`, or

`age > 17`

What we want to check is if the city variable is "Paris".

2) Change the background image

With Javascript it is possible to access the CSS properties that have been applied to an element. These are found in the style property of that object.

E.g. the style property for the body element can be accessed like this:

```
document.body.style
```

If you want to access any of the CSS properties, simple access them on the style property, like so:

```
document.body.style.backgroundImage
```

Give the <body> a new background image ("Paris.jpg").

```
document.body.style.backgroundImage = "url('images/paris.jpg')";
```

3) Change the background color

Now also change the background color to #1F1F1F.

Step 4 [Optional]

Make it change back to the default view when any other city than Paris is selected.

1) [Optional] Create "else" for default view

Create an else condition that will change the background to the following:
image: "url('images/default.jpg')"
color: "#553B2F"

An else statement looks like this:

```
if(condition) {
  // This is run if the condition is true
} else {
  // This is run if the condition is false
}
```

Step 5 [Optional]

Now lets add more cities that trigger a change in the design.

1) [Optional] Add more cities with "else if"

By adding an else if statement, you can add more checks fo cities, so that if a user selects Berlin, the background image and color can change as well.

An else if statement looks like this:

```
if(condition) {
  // This is run if the condition is true
} else if (condition) {
  // This is run if the condition is true
}
else {
  // This is run if all conditions are false
}
```