FMA WD Task 2 Code Explanation

**A page can't be manipulated safely until the document is "ready." jQuery detects this state of readiness for you. Code included inside function will only run once the page Document Object Model (DOM) is ready for JavaScript code to execute.**

$(document).ready(function() {

**When the value in the DOM element with the id = countries changes the function will execute**

$('#countries').change(function() {

**Concatenates the value of the DOM element with the id = cities with the string ‘-cities.html'**

$('#cities').load($(this).val() + '-cities.html');

**When the value of the DOM element with the id = cities changes the function will execute**

$('#cities').change(function() {

**Save the content of DOM element with the id = cities in the variable city;**

var city = $("#cities").val();

**Function getInfoApi(city) take the value saved earlier as parameter and executes**

getInfoApi(city);

});

});

});

function getInfoApi(city){

**Saving our api unique key in the variable apiKey;**

var apiKey = '01371e4a0bb4415601c7c8b191ca5ab2';

**Retrieve data from URL using the $.ajax( ) function , if successfully retrieve we execute the function with the parameter response that start processing the data**

$.ajax({

**Constructing our data URL using the API documentation, concatenating the address offered with the value of the parameter of the function and the apiKey saved earlier**

url:'http://api.openweathermap.org/data/2.5/weather?q='+city+',uk&APPID='+apiKey ,

type: 'GET' ,

dataType: 'json' ,

success: function(response) {

**Declaring variables for each of the information we want to retrieve from our API**

var cityName;

var condition;

var temperature;

var windSpeed;

var windDegree;

**Iterates over each JSON object and from our API using an anonymous function to access the current data index and saves the response in the variable name;**

$.each(response, function(index) {

date = response.dt ;

cityName = response.name;

return false;

});

**Iterates over each JSON object from our API using an anonymous function to access the current data index and saves the response in the variable condition;**

$.each(response.weather, function(index, value) {

condition = response.weather[index].main;

});

**Iterates over each JSON object from our API using an anonymous function to access the current data index and saves the response in the variable temperature;**

$.each(response.main, function(index, value) {

temperature = response.main.temp;

return false;

});

**Iterates over each JSON object from our API using an anonymous function to access the current data index and saves the response in the variable windspeed and windDegree;**

$.each(response.wind, function(index, value) {

windSpeed = response.wind.speed;

windDegree = response.wind.deg;

return false;

});

**Empty the DOM element with the class = InfoSection**

$('.InfoSection').empty();

**Save the date restived from the API in variable date**

var date = response.dt;

**Empty the Empty the DOM element with the class = weatherInfo and append it with the structure constructed with all the data obtained from our API by concatenating them together with string values of the DOM structure;**

$('.weatherInfo').empty().append(

'<section class="apiInfo"><p>'+displayWeatherIcon(condition)+'</p><p>City Name: ' + cityName + '</p><p>Date: '+formatDate(date)+'</p><p>Condition: '+condition+'</p><p>Temperature: '+temperatureConverter(temperature)+' °C</p><p>Wind Speed: '+toMph(windSpeed)+' Mph '+'('+toKmph(windSpeed)+' km/h)'+'</p><p>Wind Degree: '+degreeValue(windDegree)+'</p><p></section>');

**If the temperature is over 35 or below the -5 the the DOM element with the class = weatherInfo will be populated with a warning**

if ((temperatureConverter(temperature) > 35) ||(temperatureConverter(temperature)< -5)) {

$('.InfoSection').html('<p>Severe Weather ! Only travel if necesarry !</p>');

}

**If the Windspeed is over 50 Mph or over 80km/h the DOM element with the class = weatherInfo will be populated with a warning .**

if ((toMph(windSpeed) > 50) ||(toKmph(windSpeed))> 80) {

$('.InfoSection').html('<p>Severe Wind ! Only travel if necesarry !</p>');

}

},

**If there’s a problem recovering the data or the data isn’t valid the <div class = InfoSection> element will be populated with an error paragraph <p></p> and the <div class = weatherInfo> will remain empty.**

error: function() {

$('.InfoSection').html('<p>Error fetching the data !</p>');

$('.weatherInfo').empty();

}

});

}

**Function that takes an integer parameter and converts it into Celsius degrees**

function temperatureConverter(kelvin) {

var tempCelsius = Math.round(kelvin - 273.15);

return tempCelsius

}

**Function that takes an integer parameter and formats into a date format as DD-MM-YYYY.**

function formatDate(apiDate){

apiDate = new Date();

var dd = String(apiDate.getDate()).padStart(2, '0');

var mm = String(apiDate.getMonth() + 1).padStart(2, '0');

var yyyy = apiDate.getFullYear();

return dd + "/" + mm+"/" + yyyy;

}

**Function that takes an integer parameter and converts it Miles per Hour (Mph)**

function toMph(metersPerSec){

var speedtoMph = Math.round(metersPerSec \* 3600 / 1610.3\*1000)/1000;

return Math.round(speedtoMph);

}

**Function that takes an integer parameter and converts it kilometres per hour (Km/h)**

function toKmph(speedToMph) {

var speedToMph = toMph(speedToMph);

var KmH = Math.round(speedToMph \* 1.609);

return KmH;

}

**Function that takes an integer parameter and return the corresponding wind direction**

function degreeValue(degree) {

var windDir = degree % 360

windDir = Math.round(windDir/45)

var arr = ['Northerly','North Easterly','Easterly','South Easterly','Southerly','South Westerly','Westerly','North Westerly'];

return windDir+'° '+arr[(windDir % 8)];

}

**Function displayWeatherIcon() takes a String as a parameter. If the parameter is equal with the string value of any of the cases inside the switch block it will return an “icon” variable that contains in form of a string the address of the image corresponding with the value of the parameter.**

function displayWeatherIcon(icon) {

switch (icon) {

case "Clouds":

icon = '<img src="./weather\_icons/cloud.png" style =" width: 8%;">';

break;

case "Hail":

icon = '<img src="./weather\_icons/hail.png" style =" width: 8%;">';

break;

case "Heavy Cloud":

icon = '<img src="./weather\_icons/heavy cloud.png" style =" width: 8%;">';

break;

case "Heavy Rain":

icon = '<img src="./weather\_icons/heavy rain.png" style =" width: 8%;">';

break;

case "Rain":

icon = '<img src="./weather\_icons/rain.png" style =" width: 8%;">';

break;

case "Sleet":

icon = '<img src="./weather\_icons/sleet.png" style =" width: 8%;">';

break;

case "Snow":

icon = '<img src="./weather\_icons/snow.png" style =" width: 8%;">';

break;

case "Sun and Clouds":

icon = '<img src="./weather\_icons/sun and cloud.png" style =" width: 8%;">';

break

case "Sun":

icon = '<img src="./weather\_icons/sun.png" alt="sun" style =" width: 8%;">';

break;

case "Thunderstorm":

icon = '<img src="./weather\_icons/thunderstorm.png" style =" width: 8%;">';

break;

default:

icon = '';

break;

}

return icon;

}