```
/*
I used https://developers.google.com/maps/documentation/javascript/examples/places-
to assist with most of the code, utilising the notes that google has
provided i was able to modify the code how i need it, such as selected
countries, default locations, as well if statements to assist me in search
not just hotels (default filter that google has provided).
*/

/*This example uses the autocomplete feature of the Google Places API.
It allows the user to find all lodging, resturants and tourist attractions in a giv
country. It then displays markers for all the selected filter returned.*/
let map;
let places;
let infoWindow;
let markers = [];
let autocomplete;
let countryRestrict = {
    country: "default"
};
let MARKER_PATH =
    "https://developers.google.com/maps/documentation/javascript/images/marker_gree
const hostnameRegexp = new RegExp("^https?://.+?/");

//Chosen countries and their selected coordinates
let countries = {
    default: {
        center: {
            lat: 34.610235,
            lng: 25.234609
        },
        zoom: 3,
    },
    lb: {
        center: {
            lat: 33.89231046259173,
            lng: 35.49453087184063
        },
        zoom: 8,
    },
    ae: {
        center: {
            lat: 25.12123680578581,
            lng: 55.19897185939658
        },
        zoom: 8,
    },
    tr: {
        center: {
            lat: 39.359113,
            lng: 34.905181
        },
        zoom: 6,
    },
    eg: {
        center: {
            lat: 26.952501,
            lng: 30.490249
        },
        zoom: 6,
    },
    jo: {
        center: {
            lat: 31.954123555014338,
            lng: 35.915992265811
        },
        zoom: 9,
    },
};

/*Features within the map it self which would allow you add zoom controls map types
function initMap() {
    map = new google.maps.Map(document.getElementById("map"), {
        zoom: countries["default"].zoom,
        center: countries["default"].center,
        mapTypeControl: false,
        panControl: false,
        zoomControl: true,
        streetViewControl: false,
    });
```

CONFIGURE
JSHint
version 2.12.0
(https://github.com/jshint/j
About (/about)
Documentation (/docs)
Install (/install)
Contribute (/contribute)
Blog (/blog)

Metrics

There are 22 functions in this fil
Function with the largest signatu
median is 0.
Largest function has 25 stateme
The most complex function has
of 7 while the median is 1.5.

44 warnings

- 12 'let' is available in ES6 (us extensions (use moz).
- 13 'let' is available in ES6 (us extensions (use moz).
- 14 'let' is available in ES6 (us extensions (use moz).
- 15 'let' is available in ES6 (us extensions (use moz).
- 16 'let' is available in ES6 (us extensions (use moz).
- 17 'let' is available in ES6 (us extensions (use moz).
- 20 'let' is available in ES6 (us extensions (use moz).
- 22 'const' is available in ES6 JS extensions (use moz).
- 25 'let' is available in ES6 (us extensions (use moz).
- 112 'let' is available in ES6 (us extensions (use moz).
- 122 'let' is available in ES6 (us extensions (use moz).
- 132 'let' is available in ES6 (us extensions (use moz).
- 165 'let' is available in ES6 (us extensions (use moz).
- 169 'arrow function syntax (=> 'esversion: 6').
- 176 'let' is available in ES6 (us extensions (use moz).
- 177 'let' is available in ES6 (us extensions (use moz).
- 178 'let' is available in ES6 (us extensions (use moz).
- 199 'let' is available in ES6 (us extensions (use moz).
- 203 'arrow function syntax (=> 'esversion: 6').
- 210 'let' is available in ES6 (us extensions (use moz).

```
80
81          //Restrict the search to the default country, and to place type "cities".
82          infoWindow = new google.maps.InfoWindow({
83              content: document.getElementById("places-info-window"),
84          });
85
86          autocomplete = new google.maps.places.Autocomplete(
87              document.getElementById("city-location"), {
88                  types: ["(cities)"],
89                  componentRestrictions: countryRestrict,
90              }
91          );
92
93          places = new google.maps.places.PlacesService(map);
94          autocomplete.addListener("place_changed", getPlacesInfo);
95
96          mapDragSearch();
97          document.getElementById('tourist_attraction').addEventListener('change', getPl
98          document.getElementById('restaurant').addEventListener('change', getPlacesInfo
99          document.getElementById('lodging').addEventListener('change', getPlacesInfo);
100
101         document
102             .getElementById("myCountries")
103             .addEventListener("change", setAutocompleteCountry);
104     }
105
106     /*
107     When the user selects a city, and checks one of the filters, it will zoom into the
108     displaying what has been searched. The user can select one of the 3 choices, Touris
109     */
110     function getPlacesInfo() {
111         if ($("#tourist_attraction").is(':checked')) {
112             let place = autocomplete.getPlace();
113
114             if (place.geometry) {
115                 map.panTo(place.geometry.location);
116                 map.setZoom(15);
117                 searchAttraction();
118             } else {
119                 $('#city-location').attr("placeholder", "Please type a city or town");
120             }
121         } else if ($("#restaurant").is(':checked')) {
122             let place = autocomplete.getPlace();
123
124             if (place.geometry) {
125                 map.panTo(place.geometry.location);
126                 map.setZoom(15);
127                 searchRestaurant();
128             } else {
129                 $('#city-location').attr("placeholder", "Please type a city or town");
130             }
131         } else if ($("#lodging").is(':checked')) {
132             let place = autocomplete.getPlace();
133
134             if (place.geometry) {
135                 map.panTo(place.geometry.location);
136                 map.setZoom(15);
137                 searchLodging();
138             } else {
139                 $('#city-location').attr("placeholder", "Please type a city or town");
140             }
141         }
142     }
143
144     //when moving the map to a different map manually, it will reset the search and se
145     //area that is currently on the screen
146     function mapDragSearch() {
147         map.addListener("center_changed", function () {
148             map.addListener("dragend", function () {
149                 if ($("#tourist_attraction").is(':checked')) {
150                     searchAttraction();
151                 } else if ($("#restaurant").is(':checked')) {
152                     searchRestaurant();
153                 } else if ($("#lodging").is(':checked')) {
154                     searchLodging();
155                 } else {
156                     clearResults();
157                     clearMarkers();
158                 }
159             });
160         });
```

CONFIGURE

JSHint

version 2.12.0

(https://github.com/jshint/j

About (/about)

Documentation (/docs)

Install (/install)

Contribute (/contribute)

Blog (/blog)

Metrics

There are 22 functions in this fil
Function with the largest signatu
median is 0.
Largest function has 25 stateme
The most complex function has
of 7 while the median is 1.5.

44 warnings

- 12 'let' is available in ES6 (us extensions (use moz).
- 13 'let' is available in ES6 (us extensions (use moz).
- 14 'let' is available in ES6 (us extensions (use moz).
- 15 'let' is available in ES6 (us extensions (use moz).
- 16 'let' is available in ES6 (us extensions (use moz).
- 17 'let' is available in ES6 (us extensions (use moz).
- 20 'let' is available in ES6 (us extensions (use moz).
- 22 'const' is available in ES6 JS extensions (use moz).
- 25 'let' is available in ES6 (us extensions (use moz).
- 112 'let' is available in ES6 (us extensions (use moz).
- 122 'let' is available in ES6 (us extensions (use moz).
- 132 'let' is available in ES6 (us extensions (use moz).
- 165 'let' is available in ES6 (us extensions (use moz).
- 169 'arrow function syntax (=> 'esversion: 6').
- 176 'let' is available in ES6 (us extensions (use moz).
- 177 'let' is available in ES6 (us extensions (use moz).
- 178 'let' is available in ES6 (us extensions (use moz).
- 199 'let' is available in ES6 (us extensions (use moz).
- 203 'arrow function syntax (=> 'esversion: 6').
- 210 'let' is available in ES6 (us extensions (use moz).

```
}

//Search for tourist_attractions in the selected city within the viewport of the ma
function searchAttraction() {
    let search = {
        bounds: map.getBounds(),
        types: ["tourist_attraction"],
    };
    places.nearbySearch(search, (results, status) => {
        if (status === google.maps.places.PlacesServiceStatus.OK) {
            clearResults();
            clearMarkers();

            // Create a marker for each attraction found, and
            // assign a letter of the alphabetic to each marker icon.
            for (let i = 0; i < results.length; i++) {
                let markerLetter = String.fromCharCode("A".charCodeAt(0) + (i % 26)
                let markerIcon = MARKER_PATH + markerLetter + ".png";

                // Use marker animation to drop the icons incrementally on the map
                markers[i] = new google.maps.Marker({
                    position: results[i].geometry.location,
                    animation: google.maps.Animation.DROP,
                    icon: markerIcon,
                });

                // If the user clicks a tourist_attraction marker, show the details
                markers[i].placeResult = results[i];
                google.maps.event.addListener(markers[i], "click", showInfoWindow)
                setTimeout(dropMarker(i), i * 100);
                addResult(results[i], i);
            }
        }
    });
}

//Search for resturants in the selected city within the viewport of the map
function searchRestaurant() {
    let search = {
        bounds: map.getBounds(),
        types: ["restaurant"],
    };
    places.nearbySearch(search, (results, status) => {
        if (status === google.maps.places.PlacesServiceStatus.OK) {
            clearResults();
            clearMarkers();

            // Create a marker for each resturant found, and
            // assign a letter of the alphabetic to each marker icon.
            for (let i = 0; i < results.length; i++) {
                let markerLetter = String.fromCharCode("A".charCodeAt(0) + (i % 26)
                let markerIcon = MARKER_PATH + markerLetter + ".png";

                // Use marker animation to drop the icons incrementally on the map
                markers[i] = new google.maps.Marker({
                    position: results[i].geometry.location,
                    animation: google.maps.Animation.DROP,
                    icon: markerIcon,
                });

                // If the user clicks a resturant marker, show the details of that
                markers[i].placeResult = results[i];
                google.maps.event.addListener(markers[i], "click", showInfoWindow)
                setTimeout(dropMarker(i), i * 100);
                addResult(results[i], i);
            }
        }
    });
}

//Search for lodging in the selected city within the viewport of the map
function searchLodging() {
    let search = {
        bounds: map.getBounds(),
        types: ["lodging"],
    };
    places.nearbySearch(search, (results, status) => {
        if (status === google.maps.places.PlacesServiceStatus.OK) {
            clearResults();
            clearMarkers();

```

CONFIGURE

JSHint

Metrics

version 2.12.0

There are 22 functions in this fil
(https://github.com/jshint/j
Function with the largest signatu
median is 0.

About (/about)

Largest function has 25 stateme
Documentation (/docs)
The most complex function has
Install (/install)
of 7 while the median is 1.5.

Contribute (/contribute)

Blog (/blog)

44 warnings

- 12 'let' is available in ES6 (us extensions (use moz).
- 13 'let' is available in ES6 (us extensions (use moz).
- 14 'let' is available in ES6 (us extensions (use moz).
- 15 'let' is available in ES6 (us extensions (use moz).
- 16 'let' is available in ES6 (us extensions (use moz).
- 17 'let' is available in ES6 (us extensions (use moz).
- 20 'let' is available in ES6 (us extensions (use moz).
- 22 'const' is available in ES6 JS extensions (use moz).
- 25 'let' is available in ES6 (us extensions (use moz).
- 112 'let' is available in ES6 (us extensions (use moz).
- 122 'let' is available in ES6 (us extensions (use moz).
- 132 'let' is available in ES6 (us extensions (use moz).
- 165 'let' is available in ES6 (us extensions (use moz).
- 169 'arrow function syntax (=> 'esversion: 6').
- 176 'let' is available in ES6 (us extensions (use moz).
- 177 'let' is available in ES6 (us extensions (use moz).
- 178 'let' is available in ES6 (us extensions (use moz).
- 199 'let' is available in ES6 (us extensions (use moz).
- 203 'arrow function syntax (=> 'esversion: 6').
- 210 'let' is available in ES6 (us extensions (use moz).

```
242             // Create a marker for each lodging found, and
243             // assign a letter of the alphabetic to each marker icon.
244             for (let i = 0; i < results.length; i++) {
245                 let markerLetter = String.fromCharCode("A".charCodeAt(0) + (i % 26)
246                 let markerIcon = MARKER_PATH + markerLetter + ".png";
247
248                 // Use marker animation to drop the icons incrementally on the map
249                 markers[i] = new google.maps.Marker({
250                     position: results[i].geometry.location,
251                     animation: google.maps.Animation.DROP,
252                     icon: markerIcon,
253                 });
254
255                 // If the user clicks a lodging marker, show the details of that p
256                 markers[i].placeResult = results[i];
257                 google.maps.event.addListener(markers[i], "click", showInfoWindow)
258                 setTimeout(dropMarker(i), i * 100);
259                 addResult(results[i], i);
260             }
261         }
262     });
263 }
264
265 // Set the country restriction based on user input, also centers and zoomz the map
266 // The restrictions are taken from 'myCountries' dropdown value.
267 function setAutocompleteCountry() {
268     const country = document.getElementById("myCountries").value;
269
270     if (country == "default") {
271         autocomplete.setComponentRestrictions({
272             country: []
273         });
274         map.setCenter({
275             lat: 15,
276             lng: 0
277         });
278         map.setZoom(2);
279     } else {
280         autocomplete.setComponentRestrictions({
281             country: country
282         });
283         map.setCenter(countries[country].center);
284         map.setZoom(countries[country].zoom);
285     }
286     clearResults();
287     clearMarkers();
288 }
289
290 function addResult(result, i) {
291     const results = document.getElementById("results");
292     const markerLetter = String.fromCharCode("A".charCodeAt(0) + (i % 26));
293     const markerIcon = MARKER_PATH + markerLetter + ".png";
294     const tr = document.createElement("tr");
295     tr.style.backgroundColor = i % 2 === 0 ? "#ef7b18" : "#f7bf90";
296
297     tr.onclick = function () {
298         google.maps.event.trigger(markers[i], "click");
299     };
300     const iconTd = document.createElement("td");
301     const nameTd = document.createElement("td");
302     const icon = document.createElement("img");
303     icon.src = markerIcon;
304     icon.setAttribute("class", "placeIcon");
305     icon.setAttribute("className", "placeIcon");
306     const name = document.createTextNode(result.name);
307     iconTd.appendChild(icon);
308     nameTd.appendChild(name);
309     tr.appendChild(iconTd);
310     tr.appendChild(nameTd);
311     results.appendChild(tr);
312 }
313
314 /* Get the place details for the chosen filter. Show the information in an info wi
315 anchored on the marker for the filter that the user selected. */
316 function showInfoWindow() {
317     const marker = this;
318     places.getDetails({
319             placeId: marker.placeResult.place_id
320         },
321         (place, status) => {
322             if (status !== google.maps.places.PlacesServiceStatus.OK) {
```

CONFIGURE
JS Hint
Metrics
version 2.12.0
There are 22 functions in this fil
(https://github.com/jshint/j
Function with the largest signatu
median is 0.
About (/about)
Largest function has 25 stateme
Documentation (/docs)
The most complex function has
Install (/install)
of 7 while the median is 1.5.
Contribute (/contribute)
Blog (/blog)
44 warnings
12 'let' is available in ES6 (us
extensions (use moz).
13 'let' is available in ES6 (us
extensions (use moz).
14 'let' is available in ES6 (us
extensions (use moz).
15 'let' is available in ES6 (us
extensions (use moz).
16 'let' is available in ES6 (us
extensions (use moz).
17 'let' is available in ES6 (us
extensions (use moz).
20 'let' is available in ES6 (us
extensions (use moz).
22 'const' is available in ES6
JS extensions (use moz).
25 'let' is available in ES6 (us
extensions (use moz).
112 'let' is available in ES6 (us
extensions (use moz).
122 'let' is available in ES6 (us
extensions (use moz).
132 'let' is available in ES6 (us
extensions (use moz).
165 'let' is available in ES6 (us
extensions (use moz).
169 'arrow function syntax (=>
'esversion: 6').
176 'let' is available in ES6 (us
extensions (use moz).
177 'let' is available in ES6 (us
extensions (use moz).
178 'let' is available in ES6 (us
extensions (use moz).
199 'let' is available in ES6 (us
extensions (use moz).
203 'arrow function syntax (=>
'esversion: 6').
210 'let' is available in ES6 (us
extensions (use moz).
```
```

```
                return;
            }
            infoWindow.open(map, marker);
            buildIWContent(place);
        }
    );
}

// Load the place information into the HTML elements used by the info window.
function buildIWContent(place) {
    document.getElementById("iw-icon").innerHTML =
        '<img class="hotelIcon" ' + 'src="' + place.icon + '"/>';
    document.getElementById("iw-url").innerHTML =
        '<b><a href="' + place.url + '">' + place.name + "</a></b>";
    document.getElementById("iw-address").textContent = place.vicinity;

    if (place.formatted_phone_number) {
        document.getElementById("iw-phone-row").style.display = "";
        document.getElementById("iw-phone").textContent =
            place.formatted_phone_number;
    } else {
        document.getElementById("iw-phone-row").style.display = "none";
    }

    // Assign a five-star rating to the hotel, using a black star ('&#10029;')
    // to indicate the rating the hotel has earned, and a white star ('&#10025;')
    // for the rating points not achieved.
    if (place.rating) {
        let ratingHtml = "";

        for (let i = 0; i < 5; i++) {
            if (place.rating < i + 0.5) {
                ratingHtml += "&#10025;";
            } else {
                ratingHtml += "&#10029;";
            }
            document.getElementById("iw-rating-row").style.display = "";
            document.getElementById("iw-rating").innerHTML = ratingHtml;
        }
    } else {
        document.getElementById("iw-rating-row").style.display = "none";
    }

    // The regexp isolates the first part of the URL (domain plus subdomain)
    // to give a short URL for displaying in the info window.
    if (place.website) {
        let fullUrl = place.website;
        let website = String(hostnameRegexp.exec(place.website));

        if (!website) {
            website = "http://" + place.website + "/";
            fullUrl = website;
        }
        document.getElementById("iw-website-row").style.display = "";
        document.getElementById("iw-website").textContent = website;
    } else {
        document.getElementById("iw-website-row").style.display = "none";
    }
}

//This drops the marker onto the map
function dropMarker(i) {
    return function () {
        markers[i].setMap(map);
    };
}

//This removes the marker from the map, it is also activated when you press the res
function clearMarkers() {
    for (let i = 0; i < markers.length; i++) {
        if (markers[i]) {
            markers[i].setMap(null);
        }
    }
    markers = [];
}

//The function that clears the results
function clearResults() {
    let results = document.getElementById("results");
    while (results.childNodes[0]) {
```

CONFIGURE
JS Hint
version 2.12.0
Metrics
There are 22 functions in this fil
(https://github.com/jshint/j
Function with the largest signatu
median is 0.
About (/about)
Largest function has 25 stateme
Documentation (/docs)
The most complex function has
Install (/install)
of 7 while the median is 1.5.
Contribute (/contribute)
Blog (/blog)
44 warnings
12 'let' is available in ES6 (us
extensions (use moz).
13 'let' is available in ES6 (us
extensions (use moz).
14 'let' is available in ES6 (us
extensions (use moz).
15 'let' is available in ES6 (us
extensions (use moz).
16 'let' is available in ES6 (us
extensions (use moz).
17 'let' is available in ES6 (us
extensions (use moz).
20 'let' is available in ES6 (us
extensions (use moz).
22 'const' is available in ES6
JS extensions (use moz).
25 'let' is available in ES6 (us
extensions (use moz).
112 'let' is available in ES6 (us
extensions (use moz).
122 'let' is available in ES6 (us
extensions (use moz).
132 'let' is available in ES6 (us
extensions (use moz).
165 'let' is available in ES6 (us
extensions (use moz).
169 'arrow function syntax (=>
'esversion: 6').
176 'let' is available in ES6 (us
extensions (use moz).
177 'let' is available in ES6 (us
extensions (use moz).
178 'let' is available in ES6 (us
extensions (use moz).
199 'let' is available in ES6 (us
extensions (use moz).
203 'arrow function syntax (=>
'esversion: 6').
210 'let' is available in ES6 (us
extensions (use moz).
```
```

```
404             results.removeChild(results.childNodes[0]);
405         }
406     }
407
408     //Clears the results, the search fields, the filters (radio buttons) and the map w
409     function resetCountries() {
410         clearResults();
411         clearMarkers();
412         $('#myCountries')[0].selectedIndex = 0;
413         $('#city-location').val("");
414         $('input[type=radio]').prop('checked', false);
415         map.setZoom(3);
416         map.setCenter(countries["default"].center);
417         map.componentRestrictions = {
418             'country': []
419         };
420     }
421
```

CONFIGURE

JSHint

version 2.12.0

(https://github.com/jshint/j

About (/about)

Documentation (/docs)

Install (/install)

Contribute (/contribute)

Blog (/blog)

Metrics

There are 22 functions in this fil
Function with the largest signatu
median is 0.
Largest function has 25 stateme
The most complex function has
of 7 while the median is 1.5.

44 warnings

- 12 'let' is available in ES6 (us extensions (use moz).
- 13 'let' is available in ES6 (us extensions (use moz).
- 14 'let' is available in ES6 (us extensions (use moz).
- 15 'let' is available in ES6 (us extensions (use moz).
- 16 'let' is available in ES6 (us extensions (use moz).
- 17 'let' is available in ES6 (us extensions (use moz).
- 20 'let' is available in ES6 (us extensions (use moz).
- 22 'const' is available in ES6 JS extensions (use moz).
- 25 'let' is available in ES6 (us extensions (use moz).
- 112 'let' is available in ES6 (us extensions (use moz).
- 122 'let' is available in ES6 (us extensions (use moz).
- 132 'let' is available in ES6 (us extensions (use moz).
- 165 'let' is available in ES6 (us extensions (use moz).
- 169 'arrow function syntax (=> 'esversion: 6').
- 176 'let' is available in ES6 (us extensions (use moz).
- 177 'let' is available in ES6 (us extensions (use moz).
- 178 'let' is available in ES6 (us extensions (use moz).
- 199 'let' is available in ES6 (us extensions (use moz).
- 203 'arrow function syntax (=> 'esversion: 6').
- 210 'let' is available in ES6 (us extensions (use moz).