

Chapter 10 – Interactive Map

Step 1:

1. Setup an account on MapQuest Developer.
 - a. Begin by clicking on 'Get Started' and follow the directions.
2. Once the account is created, sign into MapQuest Developer.
 - a. Select 'Profile'.
 - b. Select 'Manage Keys' and then select 'Create a New Key'.
 - i. Give a name for the App and select 'Create'.
 - ii. Once the API has been created, expand the approved app.
 - iii. You will need to copy the API key into the code you will be adding to the js10b.js file.
3. Use your code editor to open the **js10b_txt.html** and **js10b_txt.js** files. Enter your name and the date in the comment section of each file and then save them as **js10b.html** and **js010b.js**, respectively.
 - a. Organize your files in the appropriate folders.

```
<title>Oak Top House: Driving Directions</title>

<link rel="stylesheet" href="styles/stylesb.css" />

<link type="text/css" rel="stylesheet" href="https://api.mqcdn.com/sdk/mapquest-js/
v1.3.2/mapquest.css"/>

<script src="scripts/js10b.js" defer></script>

<script src="https://api.mqcdn.com/sdk/mapquest-js/v1.3.2/mapquest.js"></script>

</head>
```

4. Add the following highlighted code to the <head> of the HTML file.
5. Add the API key and MapQuest map object to the **js10b.js** file.

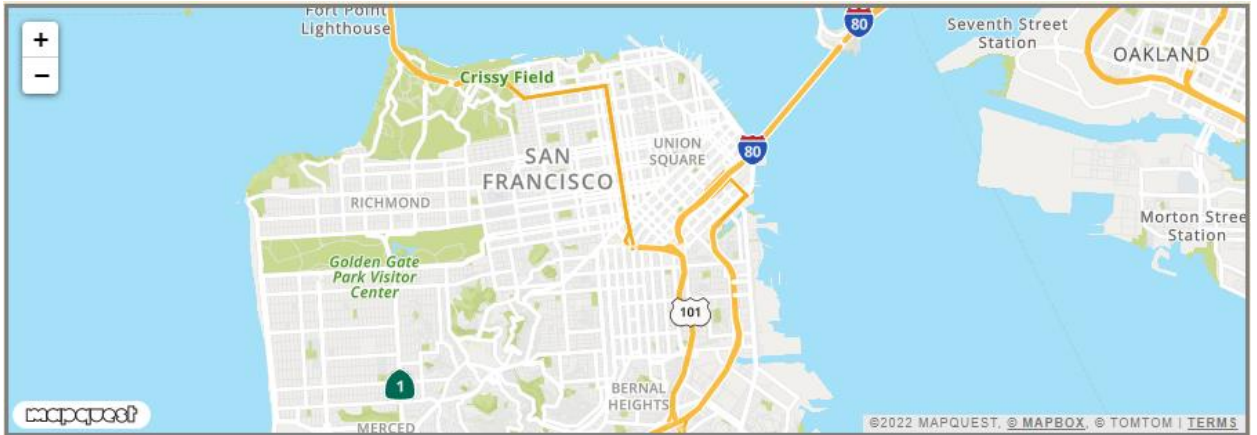
```
// Setup API key and initialize the map

L.mapquest.key = 'Enter your API key here.'; // Add your API key.

// 'map' refers to a <div> element with the ID map

let map = L.mapquest.map('map', { // Adds map at <div id="map">.
  center: [37.7749, -122.4194], // latitude , longitude
  zoom: 12
});
```

6. Save the file and view the map on the Oak Top House page.



Step 2:

1. Add the code for the Map layer (styles) control to the **js10b.js** file.

```
L.mapquest.key = 'lYrP4vF3Uk5zgTiGGuEzQGwGIVDGuy24';
var baseLayer = L.mapquest.tileLayer('map');
var map = L.mapquest.map('map', {
  center: [37.7749, -122.4194],
  layers: baseLayer,
  zoom: 12
});

// Add the Map layers
L.control.layers({
  'Map': baseLayer,
  'Hybrid': L.mapquest.tileLayer('hybrid'),
  'Satellite': L.mapquest.tileLayer('satellite'),
  'Light': L.mapquest.tileLayer('light'),
  'Dark': L.mapquest.tileLayer('dark')
}).addTo(map);
```

Save default map style to a variable.

Setup default map layer style.

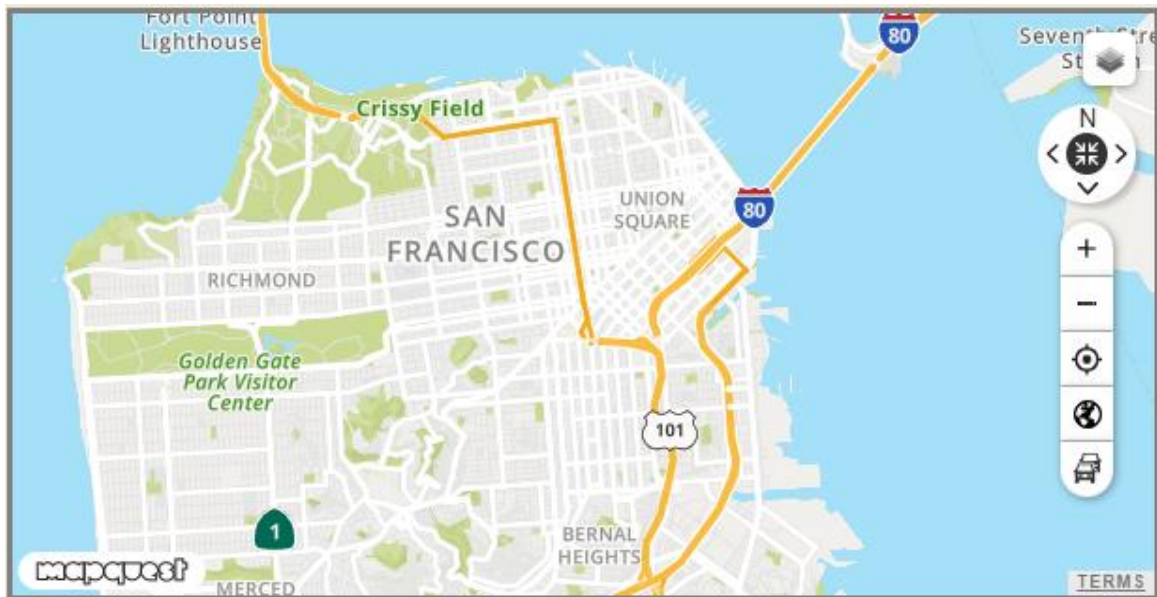
Map style. Options are: 'map', 'hybrid', 'satellite', 'light' and 'dark'.

2. Save the file.
3. Test the Map Layer Control.
4. Add the code for the MapQuest Control to the **js10b.js** file.

```
// Add the MapQuest Control
map.addControl(L.mapquest.control());

} // end .onload event
```

5. Save the file.
6. Test the MapQuest Control.



Step 3:

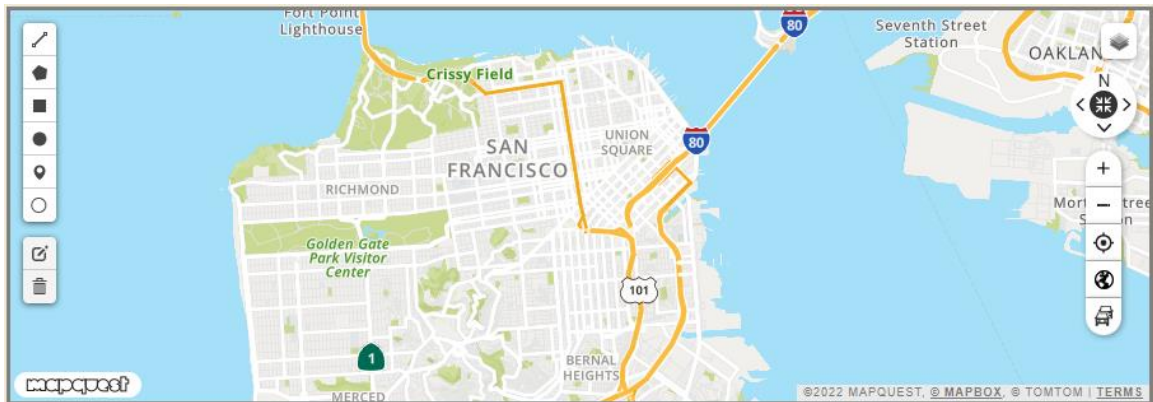
1. Add code for the Leaflet Draw Control to the **js10b.js** file.

```
// Map with Leaflet Draw Control
var drawnItems = L.featureGroup().addTo(map);
map.addControl(new L.Control.Draw({
  edit: {
    featureGroup: drawnItems,
    poly: {
      allowIntersection: false
    }
  },
  draw: {
    polygon: {
      allowIntersection: false,
      showArea: true
    }
  }
}));
map.on(L.Draw.Event.CREATED, function (event) {
  var layer = event.layer;
  drawnItems.addLayer(layer);
})
```

Tells which FeatureGroup contains the layers that should be editable. Can contain 0 or more features with geometry type Point, LineString, Polygon.

The type of layer this is. One of: polyline; polygon; rectangle; circle; marker. Triggered when a new vector or marker has

2. Save the file and view in the browser.



Step 4:

1. Add Basic Geocoding to the **js10b.js** file. Add in the address for FVTC.

```
// Basic Geocoding
```

```
L.mapquest.geocoding().geocode('1825 N Bluemound Dr, Appleton WI');
```

2. Comment out center: [37.7749, -122.4194], and change to center: [0,0], in the **js10b.js** file.
(Note: The basic geocoding will still point to FVTC even if this is not changed. The change to [0,0] is a non-specific location.)

```
var map = L.mapquest.map('map', {  
  /* center: [37.7749, -122.4194], */  
  center: [0, 0],  
  layers: baseLayer,  
  zoom: 12  
});
```

3. Save the file and view the map.



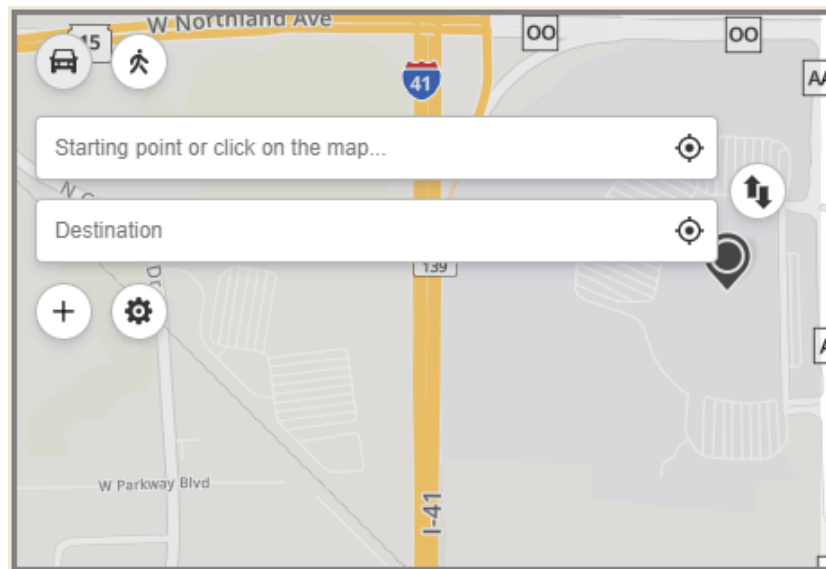
Step 5:

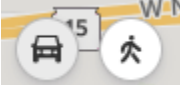




1. Add the code for the Directions Control to the **js10b.js** file.

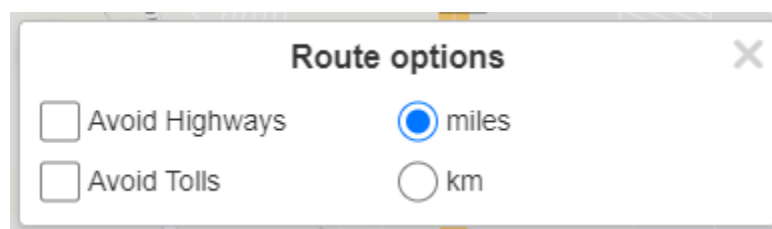
```
// Map with a Directions Control

L.mapquest.directionsControl({
  routeSummary: {
    enabled: false,
    draggable: true ← Allows the ability to drag the markers.
  },
  narrativeControl: {
    enabled: true, ← Allows the ability to drag the markers.
    compactResults: false
  },
  routeRibbon: {
    showTraffic: true
  },
  alternateRouteRibbon: {
    showTraffic: true
  }
}).addTo(map);
```

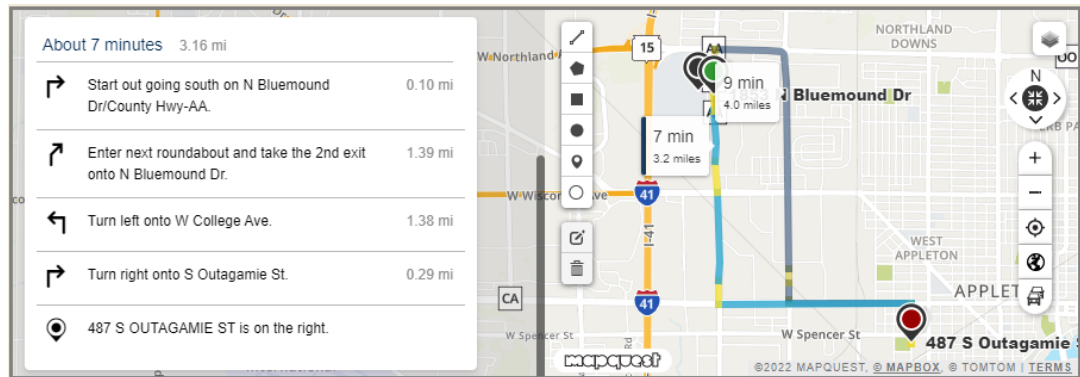
2. Save the map and enter some locations to see how the directions control functions.



- a.  Determines if the directions are for traveling by car or by walking. The 'grayed' out icon is the one selected.
- b.  Clicking the location icon, found on the right side of the textbox, will put in your current location provided your device allows the program to know your location. You can also just click on the map for a location to be entered.
- c.  When there are only two locations, this icon allows you to switch directions.
- d.  Allows you to add up to 10 stops.
- e.  Clicking on this icon opens the following dialog box where you can choose various options.



- f. Text narrative, with a scroll bar is provided with step-by-step directions. The map shows the alternate routes, with times and distance, along with traffic patterns.



Step 6:

1. Add the code to the **js10b.js** file to toggle the button to hide and show the directions control.
 - a. Create a global variable to hold the code for the button element.

```
const btnDir = document.getElementById("directions");
```

```
window.onload = function() {
```

```
// Setup API key and initialize the map
```

```
L.mapquest.key = 'lYrP4vF3Uk5zgTiGGuEzQGwGIVDGuy24'
```

- b. Add an event listener for a 'click' event on the button.

```
// Map Directions Click Event
```

```
btnDir.addEventListener('click', () => {
```

```
// Map with a Directions Control
```

```
L.mapquest.directionsControl({
```

```
  routeSummary: {
```

```
    enabled: false,
```

```
    draggable: true
```

```
    ...
```

```
    alternateRouteRibbon: {
```

```
      showTraffic: true
```

```
    }
```

```
  }).addTo(map);
```

```
}); // end btnDir click event
```

2. To toggle the button a *switch* statement is added, within the 'click' event, that will show and hide the directions control and change the text on the button.

```
// Add switch to add and remove the directions panel.
switch (btnDir.value) {
  case "Map Directions":
    console.log("showing");
    btnDir.value = "Hide Directions";

    // Map with a Directions Control
    L.mapquest.directionsControl({

      ● ● ●

    }).addTo(map);

    break;

  case "Hide Directions":
    console.log("hiding");
    btnDir.value = "Map Directions";
    location.reload();
    break;
} // end of switch statement
}); // end btnDir click event
```

3. Save the file.
4. Test that the button will show/hide the directions control and change the text on the button.