# CS 312
## Internet Concepts

# Web Programming/Scripting:
# Google Maps API v3

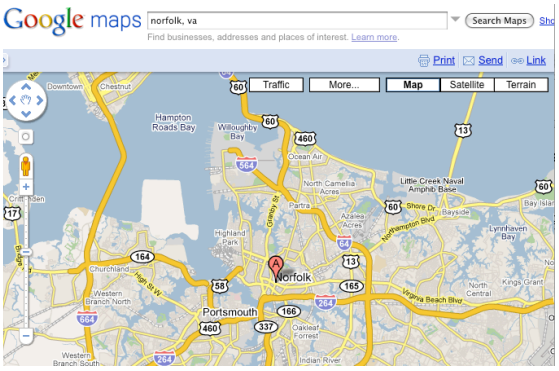
*Dr. Michele Weigle*
Department of Computer Science
Old Dominion University
*mweigle@cs.odu.edu*

***http://www.cs.odu.edu/~mweigle/CS312-F09/***



## Google Maps API v3
### Outline

- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3

# Google Maps API v3
## References

- **Tutorial**
  - » http://code.google.com/apis/maps/documentation/v3/introduction.html
- **API Reference**
  - » http://code.google.com/apis/maps/documentation/v3/reference.html
- Samples
  - » http://code.google.com/apis/maps/documentation/v3/examples/
- Demo Gallery
  - » http://code.google.com/apis/maps/documentation/v3/demogallery.html
- Third-Party Tutorial
  - » http://www.svennerberg.com/2009/06/google-maps-api-3-the-basics/



# Google Maps API v3
## Important!

- Your map **will not load** if you include the DOCTYPE line.
  - » Version 3 is technically still in beta, so there will be a few kinks

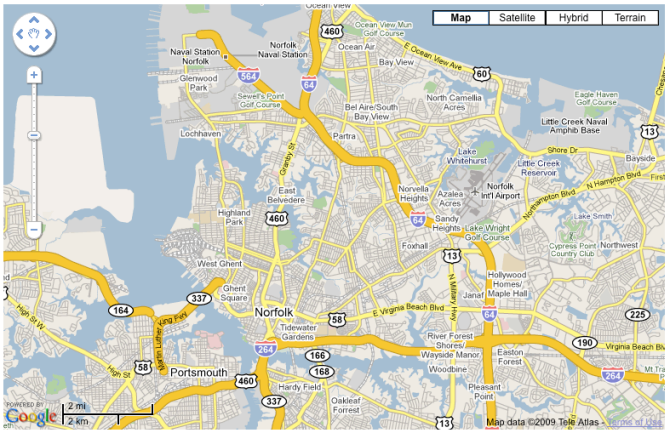
```
<?xml version="1.0" encoding="UTF-8"?>
<!--<!DOCTYPE html PUBLIC "-//W3C//DTD
XHTML 1.0 Transitional//EN" "http://www\
.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">-->
<html xmlns="http://www.w3.org/1999/xhtml">
```

If there's a problem with your code, you'll just see a blank page – no map will display.

# Google Maps API

## "Hello World" in Three Steps

- Load Google Maps API JavaScript
- Write initialize() JavaScript function
  - » all map setup goes here
- Create an HTML  element for the map to be placed in and display map on page load




http://code.google.com/apis/maps/documentation/v3/introduction.html



# "Hello World"

## Step 1 - Load Google Maps API JavaScript

*[ normal HTML heading stuff* ***but not the DOCTYPE line!****]*

```
<head>
<title>Google Maps Hello World</title>

<script type="text/javascript"
  src="http://maps.google.com/maps/api/js?sensor=false"> </script>
```

# "Hello World"

## Step 2 - JavaScript initialize() function

```
<script type="text/javascript">
  function initialize() {
    var latlng = new google.maps.LatLng(36.8854, -76.2581);

    var options = {
      zoom: 12,
      center: latlng,
      mapTypeId: google.maps.MapTypeId.ROADMAP
    };

    var map = new google.maps.Map(document.getElementById("map"),
                  options);
  }
</script>
```

create map coordinates for ODU

all objects are in google.maps namespace

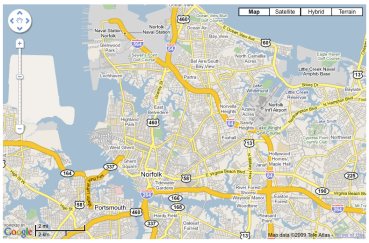
set zoom level, map center, and map type

create the map and display in id "map"



# "Hello World"

## Step 3 - Create  Element



load map

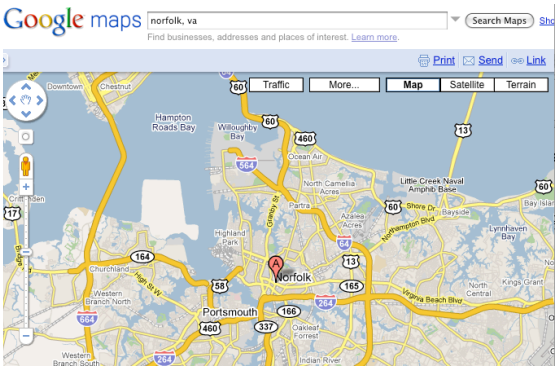
```
</head>

<body onload="initialize()">
  <div id="map" style="width: 800px; height: 500px"></div>
</body>

</html>
```

named element

http://www.cs.odu.edu/~mweigle/cs312/gmap/hello-world.html

# Google Maps API v3
## Outline

- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3



# Google Maps Basics
## Map Type - MapTypeId

- No default map type, must be set for all maps
- ROADMAP
  - » normal, default 2D tiles of Google Maps.
- SATELLITE
  - » photographic tiles.
- HYBRID
  - » mix of photographic tiles and a tile layer for prominent features (roads, city names).
- TERRAIN
  - » physical relief tiles for displaying elevation and water features (mountains, rivers, etc.).

Can set map type in options or using setMapTypeId()

# Google Maps Basics
## Zoom Level - zoom

- 0 - 19

- 0
  - » lowest zoom (whole world)

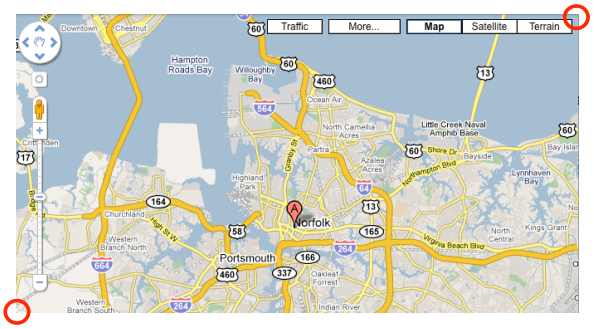
- 19
  - » highest zoom (individual buildings, if available)

- Retrieve current zoom level using getZoom()



# Google Maps Basics
## Latitudes and Longitudes

- A map displays a current "window" of the entire world within a *viewport*
  - » viewport can be defined by the rectangular points at its corners




- google.maps.LatLngBounds object
  - » defines a rectangular region using two LatLng objects representing the **southwest** and **northeast** corners of the viewport
  - » getBounds() - returns a LatLngBounds object corresponding to the current viewport

- We'll use LatLngBounds and LatLng to place objects on the map

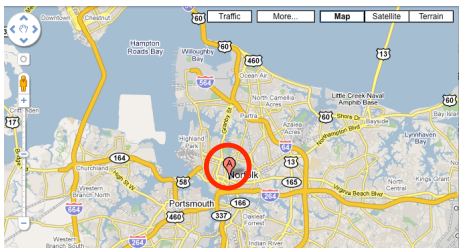
# Google Maps Basics
## Map Movement

- setCenter(*LatLng*)
  - » centers the map on the specified coordinates
- setZoom(*zoomLevel*)
  - » sets zoom to the specified level
- panTo(*LatLng*)
  - » changes the center point on the map
    - ❖ if new point is on map, uses smooth animation
    - ❖ if new point isn't on map, jumps to point



# Google Maps Basics
## Markers

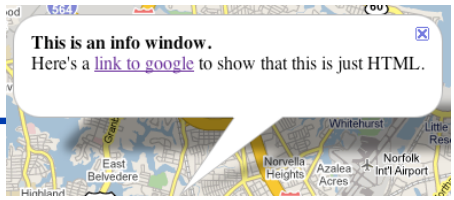
- google.maps.Marker object constructor takes a marker options object
  - » position – LatLng object
  - » map – specifies the map on which to place the marker
  - » title – tooltip text (optional)




```
var marker = new google.maps.Marker ( {
        position: latlngObject,
        map: mapObject,
        title: String
});
```

http://www.cs.odu.edu/~mweigle/cs312/gmap/hello-world-marker.html

# Google Maps Basics
## Info Windows




- Displays HTML content in a floating window
- google.maps.InfoWindow
  - » constructor takes InfoWindowOptions
  - » content: *String*
  - » position: *LatLng*
  - » maxWidth: *number*

```
infowindow.open(map);
```

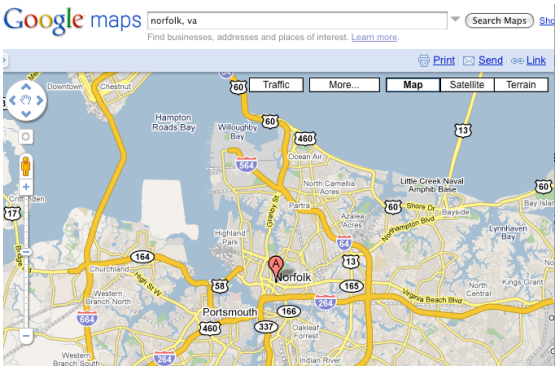
- Once the object is created, must call open() with the map as a parameter to show the window.

http://www.cs.odu.edu/~mweigle/cs312/gmap/info-window.html



# Google Maps API v3
## Outline

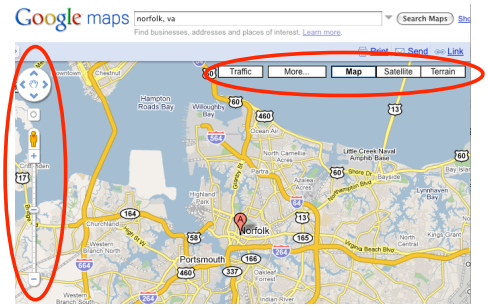
- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3

# Google Maps API
## Controls

- Navigation
  - » large pan/zoom control
  - » top-left corner
  - » can adjust the size
- Scale
  - » displays map scale
  - » not enabled by default
- MapType
  - » lets user toggle among different map types
  - » can adjust behavior of this control




http://code.google.com/apis/maps/documentation/v3/controls.html



# Google Maps API
## Controls

- There is a default set of controls.
- These can be disabled using map options
  - » disableDefaultUI: true
- Controls can also be added or modified using map options
  - » navigationControl: {*true, false*}
  - » mapTypeControl: *{true, false}*
  - » scaleControl: *{true, false}*

# Controls
## Navigation Control Options

- Modified with navigationControlOptions inside map options
- All constants have namespace of google.maps.NavigationControlStyle
  - » SMALL – mini zoom
  - » ZOOM_PAN – standard as in Google Maps
  - » ANDRIOD – small zoom as on Android phones
  - » DEFAULT – device dependent

```
var myOptions = {
  …,
  navigationControl: true,
  navigationControlOptions: {style: google.maps.NavigationStyle.SMALL},
}
```



# Controls
## Map Type Control Options

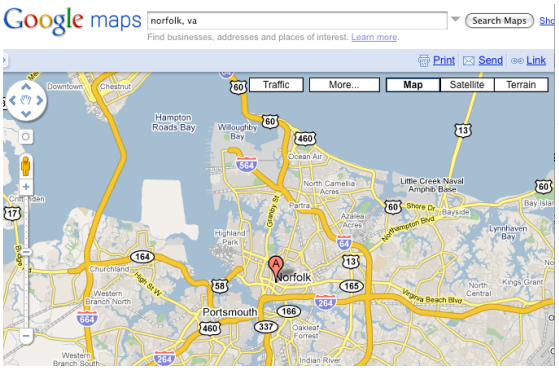
- Modified with mapTypeControlOptions inside map options
- All constants have namespace of google.maps.MapTypeControlStyle
  - » HORIZONTAL_BAR – as in Google Maps
  - » DROPDOWN_MENU – single button
  - » DEFAULT – device dependent

```
var myOptions = {
  …,
  mapTypeControl: true,
  mapTypeControlOptions: {style: google.maps.MapTypeControlStyle.DROPDOWN_MENU},
}
```

**Example:**
http://code.google.com/apis/maps/documentation/v3/examples/control-options.html

# Google Maps API v3
## Outline

- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3



# Google Maps API
## Events

- Just like regular JavaScript, we can interact with Google Maps using events
  - » user events (click, dblclick, mouseover, …)
  - » state events (zoom_changed, …)

- Register *event listeners* with the google.maps.event namespace
  - » google.maps.event.addListener (*object*, *event*, *handler*)

http://code.google.com/apis/maps/documentation/v3/events.html

# Events

## Example – hello-world-events.html

- Show info window when click on marker

```
// beginning of function initialize() is just like hello-world-marker.html

// create info window
var contentString = "<strong>You clicked!</strong>";
var infowindow = new google.maps.InfoWindow ({content: contentString});

google.maps.event.addListener (marker, "click", function() {
    infowindow.open (map, marker);
}); // end of function and addListener()

} // end initialize()
```

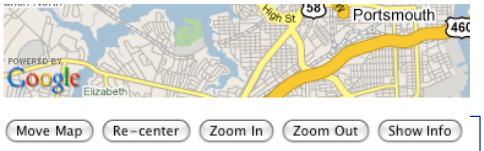
http://www.cs.odu.edu/~mweigle/cs312/gmap/hello-world-events.html



# Movement and Info Windows

## Example - moveinfo.html

- Create buttons below the map to move map, re-center, zoom in/out, and show the info window.
  - » each button calls a JavaScript function




```
<p>
<form action="" >
<input type="button" value="Move Map" onclick="animate();" />
<input type="button" value="Re-center" onclick="recenter();" />
<input type="button" value="Zoom In" onclick="zoomin();" />
<input type="button" value="Zoom Out" onclick="zoomout();" />
<input type="button" value="Show Info" onclick="showinfo();" />
</form>
</p>
```

http://www.cs.odu.edu/~mweigle/cs312/gmap/moveinfo.html

# Movement and Info Windows

## Example - moveinfo.html

- Make map and latlng global variables so they can be used in all of the functions.

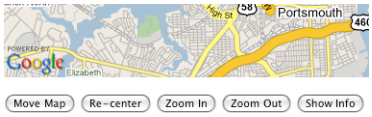
```
<script type="text/javascript">
  var map;
  var latlng;

  function initialize() {
    // just like hello-world.html except don't put var in front of
    // map or latlng (so that they stay global in scope)
  }
```

http://www.cs.odu.edu/~mweigle/cs312/gmap/moveinfo.html

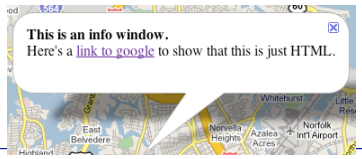


# Movement and Info Windows

## Example - moveinfo.html



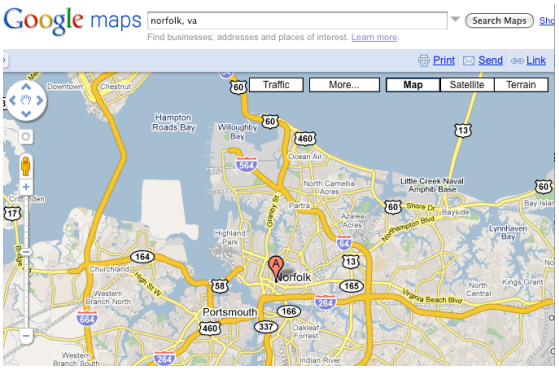

```
 function animate() { map.panTo (new google.maps.LatLng(36.9454,
-76.2581)); }

 function recenter () { map.setCenter(latlng); map.setZoom(12); }

 function zoomin () { map.setZoom (15); }

 function zoomout () { map.setZoom (6); }

 function showinfo () {
     // same as creating info window code in infowindow.html
}

</script>
```




http://www.cs.odu.edu/~mweigle/cs312/gmap/moveinfo.html

# Google Maps API v3
## Outline

- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3



# Google Maps API
## Overlays

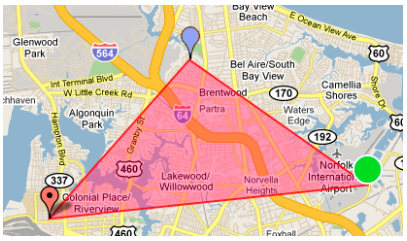
- Overlays are…
  - » objects on the map tied to latitude/longitude coordinates
  - » objects you add to the map to designate points, lines, areas

- Some types
  - » markers – points on the map
    - ❖ Marker (default marker) or Icon (customized marker)
  - » polylines – lines on the map
    - ❖ Polyline
  - » polygons - connected lines on the map
    - ❖ Polygon




# Overlays

## Example - overlays.html




- Insert blue icon
- Insert green circle icon

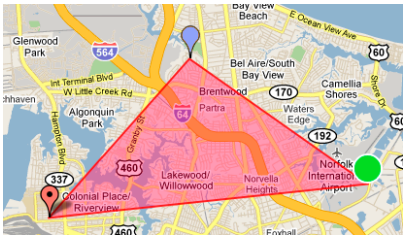
```
// beginning of function initialize() is just like hello-world-marker.html

var blueIcon = "http://gmaps-samples.googlecode.com/svn/trunk/markers/blue/blank.png";
var iconPoint = new google.maps.LatLng (36.9254, -76.2581);
var markerOptions = { position: iconPoint, map: map, icon: blueIcon };
var newMarker = new google.maps.Marker (markerOptions);

var circle = "http://gmaps-samples.googlecode.com/svn/trunk/markers/circular/greencirclemarker.png";
var circlePoint = new google.maps.LatLng(36.8900, -76.1950);
var circleOptions = { position: circlePoint, map: map, icon: circle };
var circleMarker = new google.maps.Marker (circleOptions);
```

http://www.cs.odu.edu/~mweigle/cs312/gmap/overlays.html



# Overlays

## Example - overlays.html




- Draw red polygon between icons

```
var coords = [
        new google.maps.LatLng(36.8804, -76.3081),
        new google.maps.LatLng(36.9254, -76.2581),
        new google.maps.LatLng(36.8900, -76.1950)];

var poly = new google.maps.Polygon ({
        paths: coords,
        strokeColor: "#FF0000",  strokeOpacity: 0.8, strokeWeight: 2,
        fillColor: "#FF0055",  fillOpacity: 0.35
  });

poly.setMap(map);
```

array of points

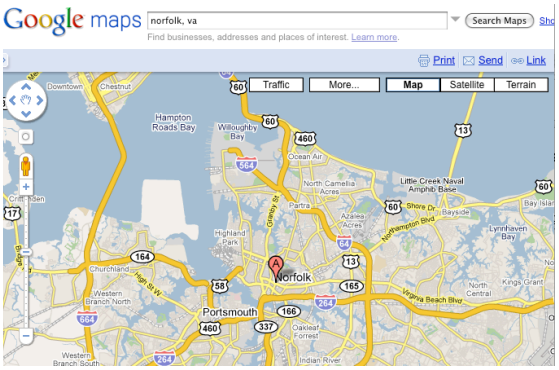
setup the polygon

put the polygon on the map

http://www.cs.odu.edu/~mweigle/cs312/gmap/overlays.html

# Google Maps API v3
## Outline

- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3



# Google Maps API
## Geocoding

- Geocoding
  - » process of converting an address to a latitude and longitude

- google.maps.Geocoder
  - » geocode() initiates a request to the geocoding service
    - ❖ object literal containing the input terms of the request
      - address string or LatLng (*reverse geocoding*)
    - ❖ callback to execute once the response has been received
      - receives GeocodeResults object array and status code as paramters

http://code.google.com/apis/maps/documentation/v3/services.html

# Geocoding
## GeocodeResults

```
results[]: {
   types[]: google.maps.GeocoderLocationType,
   formatted_address: String,
   address_components[]: {
        short_name: String,
        long_name: String,
        types[]: String
    },
    geometry: {
        location: LatLng,
        location_type: String,          // accuracy of the geocode
        viewport: LatLngBounds,   // recommended viewport
        bounds: LatLngBounds
    }
}
```

http://code.google.com/apis/maps/documentation/v3/services.html



# Geocoding
## Example - geocoding-simple.html

```
var geocoder;
var map;

function initialize() {
        // same as hello-world.html except don't put var in front of map

        geocoder = new google.maps.Geocoder();
}
```

# Geocoding
## Example - geocoding-simple.html

```
function codeAddress () {
    if (geocoder) {
      geocoder.geocode ({address: address}, function (results, status) {
        if (status == google.maps.GeocoderStatus.OK) {
                var position = results[0].geometry.location;
                map.setCenter (position);
                var marker = new google.maps.Marker ({
                        map: map, position: position});
        } else {
                alert ("Geocode was not successful because " + status);
        } // end if
       } // end function
      ); // geocode
    } // end if (geocoder)
  } // end function codeAddress
```



# Geocoding
## Example - geocoding-simple.html

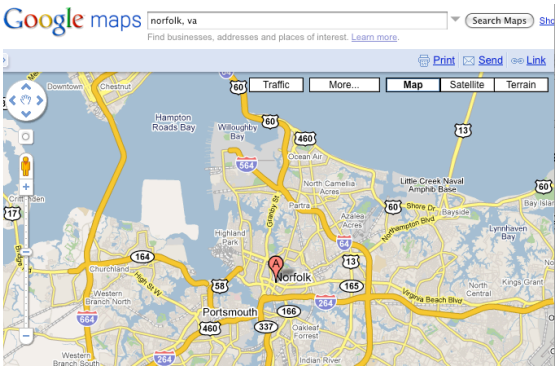
***New HTML Code***

```
<body onload="initialize()">
  <form action="">
    <p>
      <input type="text" size="60" id="address" value="4700 Elkhorn
Avenue, Norfolk, VA" />
      <input type="button" value="Go!" onclick="codeAddress()" />
    </p>
    <div id="map" style="width: 100%; height: 500px"></div>
  </form>
</body>
```

http://www.cs.odu.edu/~mweigle/cs312/gmap/geocoding.html

# Google Maps API v3
## Outline

- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3



# Google Maps API v3
## Directions

- Directions are calculated using the DirectionsService object
  - » receives direction requests and returns computed results
- Use DirectionsRenderer to render the results
  - » you can handle them yourself if you'd like
- Origins and destinations may be specified with LngLat or text strings
- Directions are displayed as a polyline drawn on the map and optionally as a series of turn-by-turn directions

http://code.google.com/apis/maps/documentation/v3/services.html

# Google Maps API v3
## Directions

- Like geocoding, directions is an asynchronous service
  - » will need a callback function
- Create a DirectionsService object
- Call route() method on that object with a DirectionsRequest object literal and a function as a parameter
  - » DirectionsRequest - origin, destination, other options



# Directions
## DirectionsRequest

```
{
 origin: LatLng | String,
 destination: LatLng | String,
 travelMode: DirectionsTravelMode,
 unitSystem: DirectionsUnitSystem,
 waypoints[]: DirectionsWaypoint,
 provideTripAlternatives: Boolean,
 region: String
}
```

http://code.google.com/apis/maps/documentation/v3/services.html

# Directions
## Displaying DirectionsResult

- Create a DirectionsRenderer object
- Call setMap() to bind it to the map
- If you want turn-by-turn directions, call setPanel() with the HTML element to display the results
- Call setDirections() passing it the DirectionsResults object.



# Directions
## Example - directions.html

```
var directionsDisplay;
var directionsService = new google.maps.DirectionsService();
var map;

 function initialize() {
        // same as hello-world.html except don't put var in front of map

        directionsDisplay = new google.maps.DirectionsRenderer();
        directionsDisplay.setMap (map);
        directionsDisplay.setPanel (document.getElementById("directions"));
}
```

# Directions

## Example - directions.html

```
function calcRoute () {
    var start = document.getElementById("start").value;
    var end = document.getElementById("end").value;
    var request = {
          origin: start,
          destination: end,
          travelMode: google.maps.DirectionsTravelMode.DRIVING
     };
     directionsService.route (request,  function (results, status) {
          if (status == google.maps.DirectionsStatus.OK) {
                directionsDisplay.setDirections(result);
          } else {
                alert ("Directions was not successful because " + status);
          } // end if
       } // end function
      ); // route
} // end function calcRoute
```



# Directions

## Example - directions.html

### HTML Code

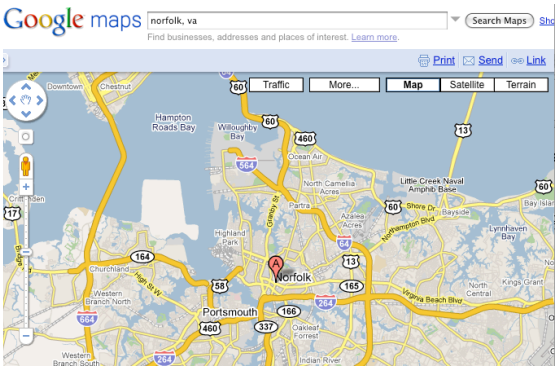
```
<body onload="initialize()">
  <form action="">
    <p>
      <label>Starting Address: </label> <input type="text" size="60"
id="start" value="4700 Elkhorn Avenue, Norfolk, VA" />
      <label>Destination Address: </label> <input type="text" size="60"
id="end" value="Chapel Hill, NC" />
      <input type="button" value="Go!" onclick="calcRoute()" />
    </p>
    <div id="map" style="width: 75%; height: 300px"></div>
    <div id="directions" style="width: 100%"></div>
  </form>
 </body>
```

http://www.cs.odu.edu/~mweigle/cs312/gmap/directions.html

# Google Maps API v3

## Outline

- "Hello World"
- Map Basics
- Controls
- Events
- Overlays
- Geocoding
- Directions




http://code.google.com/apis/maps/documentation/v3