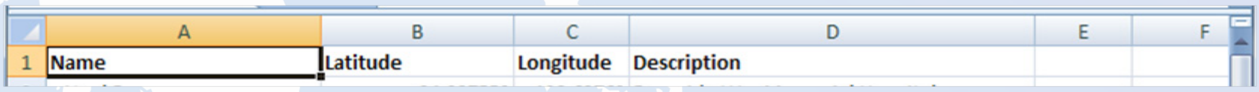
CS4HS workshop
Google maps

Developed by Claire Hocking

Class/Individual Project - Teacher Setup

Prior to setting the homework

Create 4 spreadsheets that are shared between all students in the class. Spreadsheets 1-3 should include each students name, a latitude coordinate column, a longitude coordinate column and a description column (seen below) and spreadsheet 4 should include each students name and their unique colour.



	A	B	C	D	E	F
1	Name	Latitude	Longitude	Description		

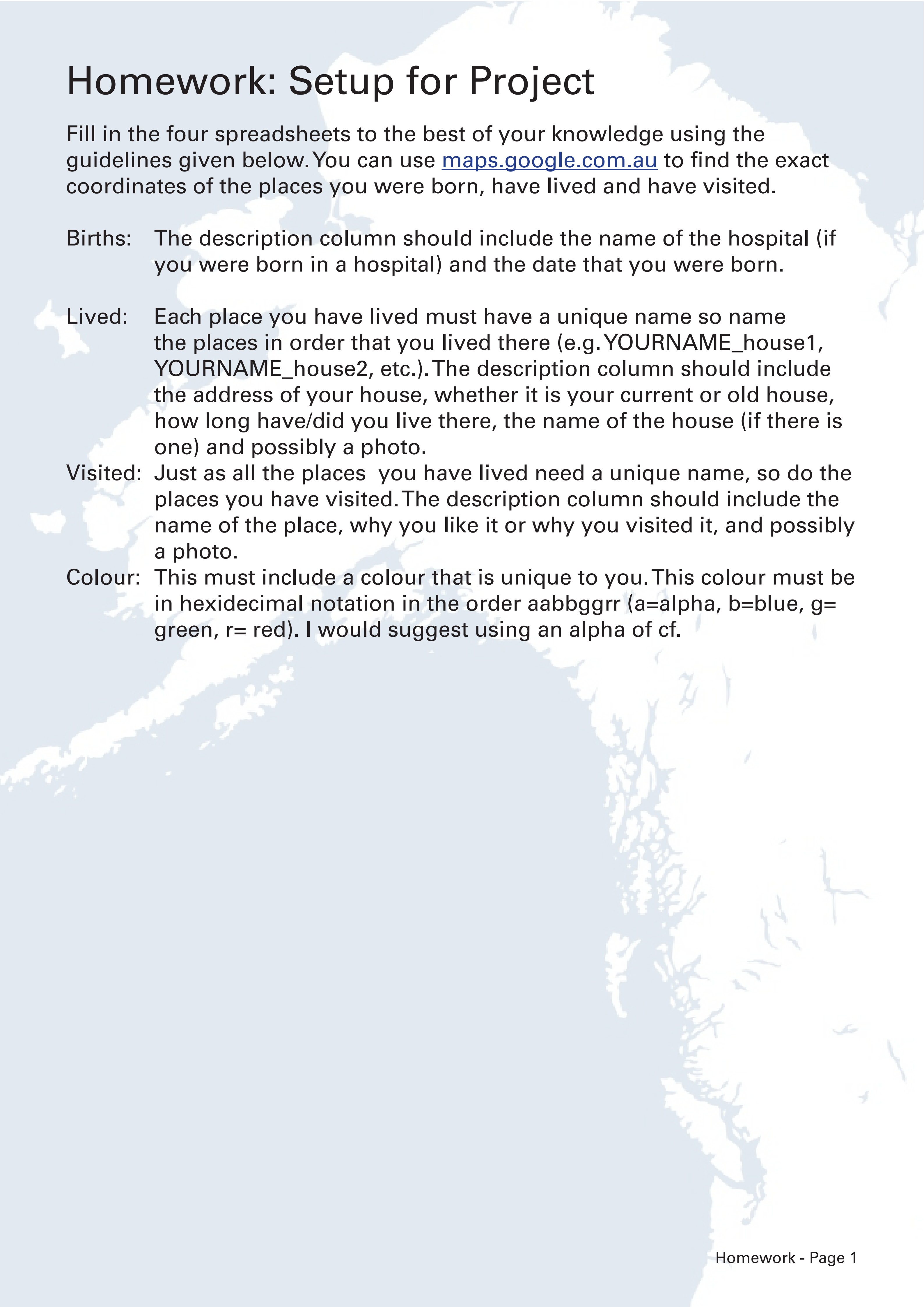
Your spreadsheets should be similar to the following:

1. Where were you born?
    a. The description should provide the name of the hospital, minimum.
2. Where have you lived?
    a. This could be more than one place.
    b. The description should include the address of your house, whether it is your current or old house, how long have/did you live there, the name of the house (if it has one) and possibly a photo.
3. What is your favourite place you have visited?
    a. The description should include the name of the place, why you like it and possibly a photo.
4. A unique colour that you want to be identified with.

I would recommend giving the homework to students at least a few days before you want to use the data in class. Also, if you are worried about students being able to find unique colours I have provided a spreadsheet with 40 colours and their corresponding hex codes in kml colour order.

Prior to commencing part 1

Create a kml file to identify the school which includes the name, latitude, longitude and description of the school. You may also want to include a link to the schools website and a photo of the school or your class in the description. However, be aware that to link in a photo it needs to be publicly accessible on the internet as does the kml file itself.



Homework: Setup for Project

Fill in the four spreadsheets to the best of your knowledge using the guidelines given below. You can use maps.google.com.au to find the exact coordinates of the places you were born, have lived and have visited.

Births:	The description column should include the name of the hospital (if you were born in a hospital) and the date that you were born.
Lived:	Each place you have lived must have a unique name so name the places in order that you lived there (e.g. YOURNAME_house1, YOURNAME_house2, etc.). The description column should include the address of your house, whether it is your current or old house, how long have/did you live there, the name of the house (if there is one) and possibly a photo.
Visited:	Just as all the places you have lived need a unique name, so do the places you have visited. The description column should include the name of the place, why you like it or why you visited it, and possibly a photo.
Colour:	This must include a colour that is unique to you. This colour must be in hexidecimal notation in the order aabbggrr (a=alpha, b=blue, g= green, r= red). I would suggest using an alpha of cf.

Worksheet 1: Basics of Google Maps

Complete the following questions while you work on the Google Maps Basic tutorial on http://www.w3schools.com/googleAPI/google_maps_basic.asp

1. How do you declare the location of the JavaScript file that loads the definitions and symbols you need to use the Google Maps API?

2. State the four map types.

3. How do you declare your unique API key?

4. What is the highest zoom level?

5a. Is it possible to load the Google Maps API on demand?

5b. Given the code below, what would you use to call this function to load the Google Maps API?

```
function loadScript()
{
var script = document.createElement("script");
script.src = "http://maps.googleapis.com/maps/api/js?callback=initialize";
document.body.appendChild(script);
}
```

5c. Why would you choose to load the API on demand?

6a. Sketch what the following code would produce. Assume that one centimeter is 100px.

```
<!DOCTYPE html>
<html>
<head>
<script scr="http://maps.googleapis.com/maps/api/js"></script>
<script>
function initialize() {
    var mapProp = {
          center: new google.maps.LatLng(51.508742,-0.120850),
          zoom:5,
          mapTypeId: google.maps.MaptypeId.ROADMAP
    };
    var map=new google.maps.Map(document.getElementById("googleMap"),mapProp);
}
google.maps.event.addDomListener(window, 'load', initialize);
</script>
</head>

<body>
<div id="googleMap" style="width:600px;height:450px;"></div>
</body>
</html>
```

6b. Label the map container and the map object on your diagram.

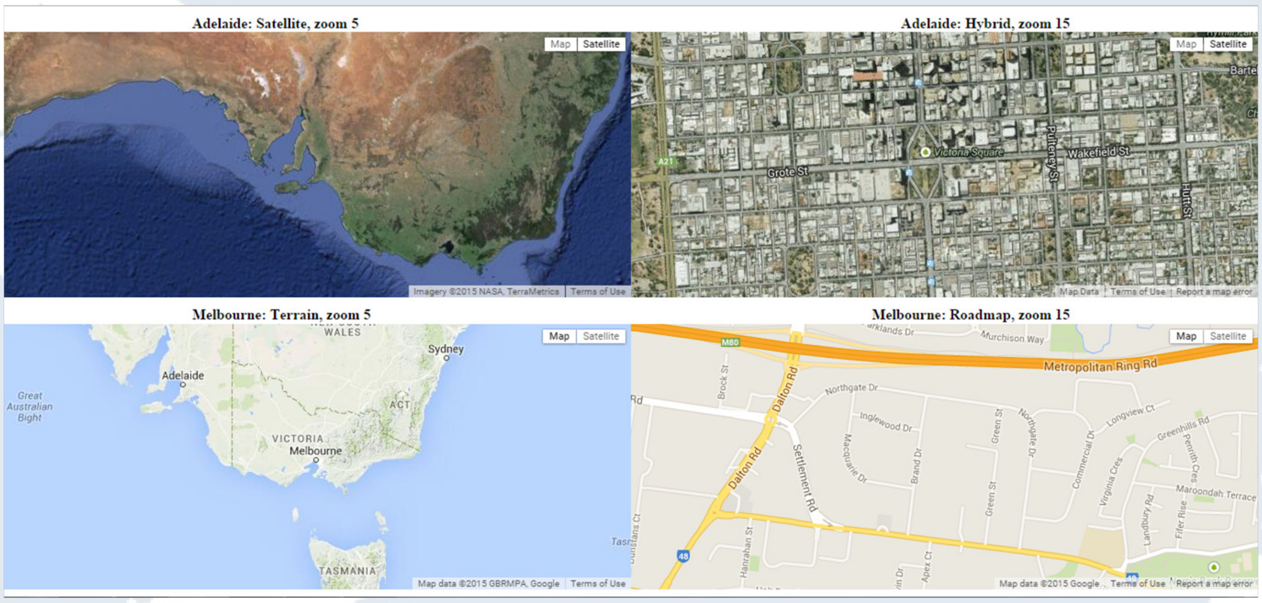
Activity 1: Types, Zoom and Centres

Aim: To show how Google Maps changes when you use different map types, zoom levels and centres.

Task: Create a webpage that shows the following.

- The four different types of map.
- Two different zoom levels.
- Two different map centre locations.

Caption each map so that users can identify the centre, zoom level and map type. Your solution should be similar to the one below but you can rearrange the maps and choose different centres.



Worksheet 2: Definitions

Fit the controls to their definitions

Rotate	Map Type	Scale	Street View
Overview Map	Pan	Zoom	

____________	: States the scale of the map. This control is not enabled by default.
____________	: Contains a small circular icon which allows you to rotate maps containing oblique (45°) imagery. This control appears by default in the top left corner of the map.
____________	: Can be dragged onto the map to enable Street View and by default appears in the top left corner of the map.
____________	: Displays a thumbnail overview map reflecting the current map viewpoint within a wider area. This control appears by default in the bottom right corner of the map, and is by default shown in its collapsed state.
____________	: Displays buttons for panning the map. This control appears by default in the top left corner of the map on non-touch devices. The Pan control also allows you to rotate 45° imagery, if available.
____________	: Displayed as a slider (for large maps) or small "+/-" buttons (for small maps) to control the zoom level of the map. This control appears by default in the left corner of the map on non-touch devices or in the bottom left corner on touch devices.
____________	: Lets the user toggle between map types and appears by default in the top right corner of the map.

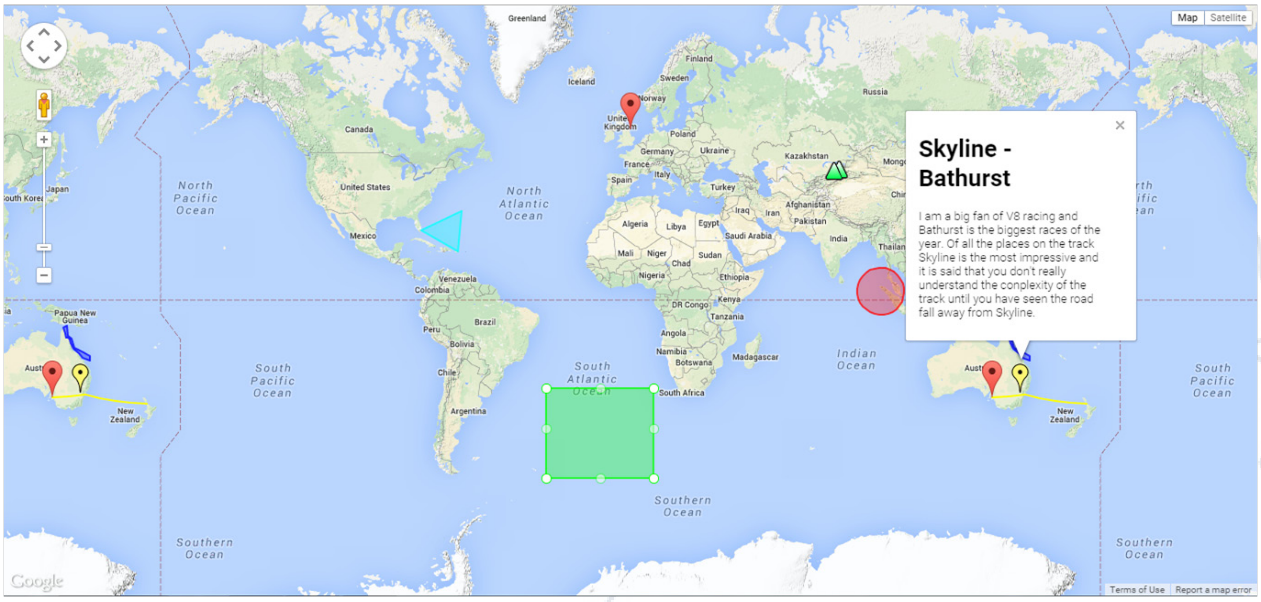
Activity 2: Map That!

Aim: To learn how to create a number of different overlays on a GoogleMap

Task: Using both the w3schools (http://www.w3schools.com/googleAPI/google_maps_overlays.asp) and Google developers (https://developers.google.com/maps/documentation/javascript/overlays) overlays tutorials create a map that displays the following.
*Note: Centre the map at (0,0)

1. A Marker indicating your house
2. An Icon showing Mt Everest
3. A polyline mapping the flight path ADL:SYD:AUK (Adelaide to Sydney, Sydney to Auckland, New Zealand)
4. An animated marker identifying Big Ben
5. A circle at the epicenter and approximate radius of the 2004 Boxing Day Tsunami
6. A triangle marking the Bermuda Triangle
7. A draggable rectangle over an ocean
8. A polygon showing the approximate area of the Great Barrier Reef
9. Info window over a place you want to visit. Include the name of the place, some information on it and why you want to visit.

Your completed map should look something like the following.



Skyline - Bathurst

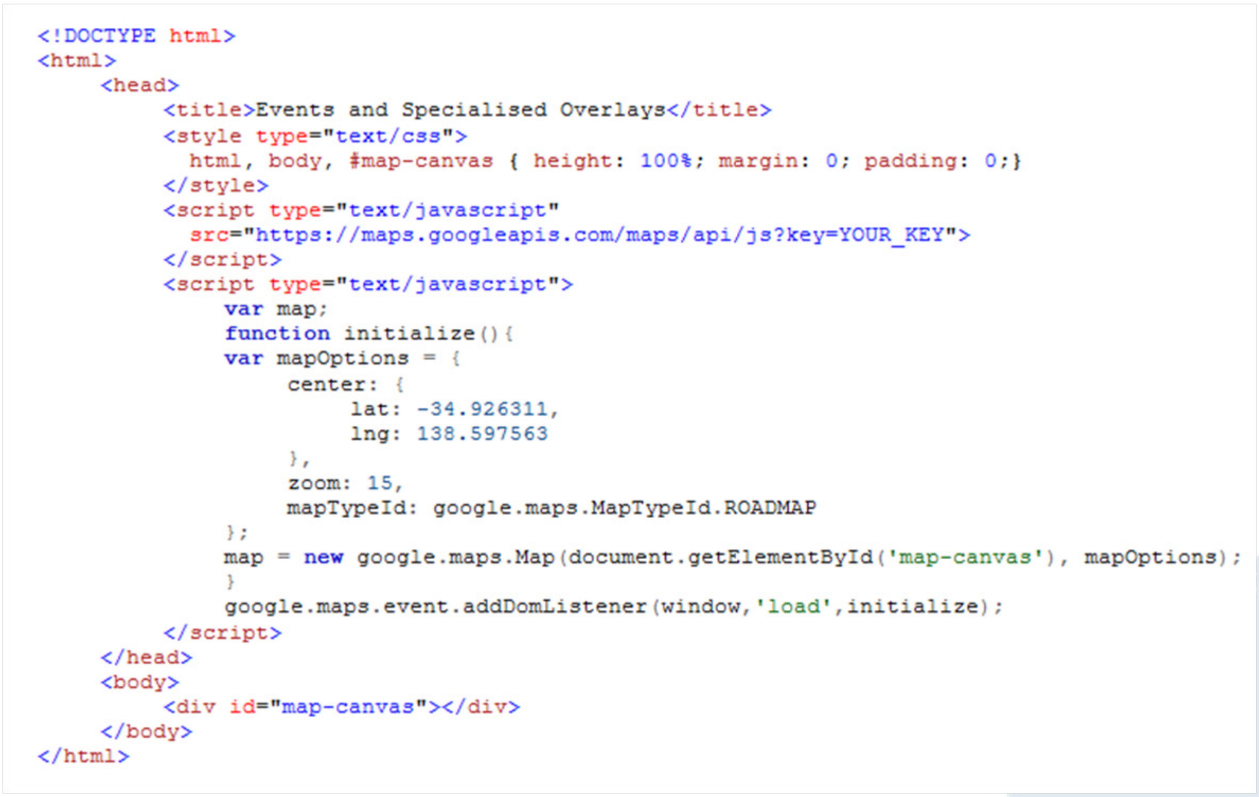
I am a big fan of V8 racing and Bathurst is the biggest races of the year. Of all the places on the track Skyline is the most impressive and it is said that you don't really understand the conplexity of the track until you have seen the road fall away from Skyline.

Activity 3: Events and Specialised Overlays

Aim: To learn how to create and manage events to show different overlays.

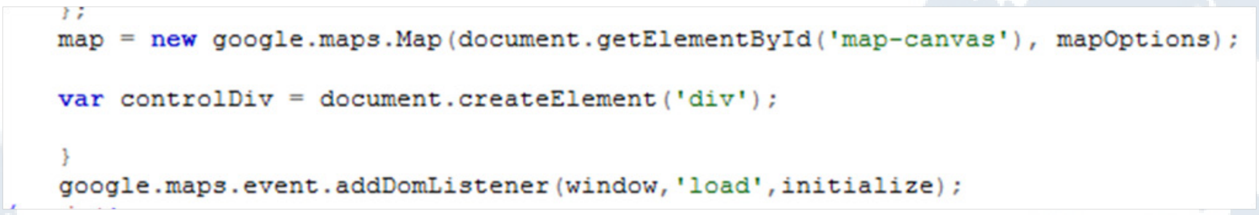
Task: Follow the steps below to create a map that allows the user to show different overlays (transit, traffic, biking) at the click of a button.

1. Create a basic map of Adelaide.



```
<!DOCTYPE html>
<html>
     <head>
          <title>Events and Specialised Overlays</title>
          <style type="text/css">
            html, body, #map-canvas { height: 100%; margin: 0; padding: 0;}
          </style>
          <script type="text/javascript"
            src="https://maps.googleapis.com/maps/api/js?key=YOUR_KEY">
          </script>
          <script type="text/javascript">
               var map;
               function initialize(){
               var mapOptions = {
                    center: {
                         lat: -34.926311,
                         lng: 138.597563
                    },
                    zoom: 15,
                    mapTypeId: google.maps.MapTypeId.ROADMAP
               };
               map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);
               }
               google.maps.event.addDomListener(window,'load',initialize);
          </script>
     </head>
     <body>
          <div id="map-canvas"></div>
     </body>
</html>
```

2. Add a button. This will be used to display the traffic overlay.
   a. At the end of the **`initialize()`** function, create a div to hold the buttons called **`ControlDiv`**



```
};
map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);

var controlDiv = document.createElement('div');

}
google.maps.event.addDomListener(window,'load',initialize);
```

b. Create a new function called **TrafficControl()**

```
function TrafficControl(controlDiv, map){
    var tControlUI = document.createElement('div');
    tControlUI.style.backgroundColor = '#FFAA00';
    tControlUI.style.border = '1px solid';
    tControlUI.style.cursor = 'pointer';
    tControlUI.style.textAlign = 'center';
    tControlUI.title = 'Show Traffic Overlay';
    controlDiv.appendChild(tControlUI);
    var tControlText = document.createElement('div');
    tControlText.style.fontFamily = 'Arial,sans-serif';
    tControlText.style.fontSize = '12px';
    tControlText.style.paddingLeft = '4px';
    tControlText.style.paddingRight = '4px';
    tControlText.style.paddingBottom = '1px';
    tControlText.style.paddingTop = '1px';
    tControlText.innerHTML = 'Traffic'
    tControlUI.appendChild(tControlText);
}
```

c. Create an instance of the traffic control button then push it to the map.

```
map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);

var controlDiv = document.createElement('div');
var tControl = new TrafficControl(controlDiv, map);
map.controls[google.maps.ControlPosition.TOP_RIGHT].push(controlDiv);
```

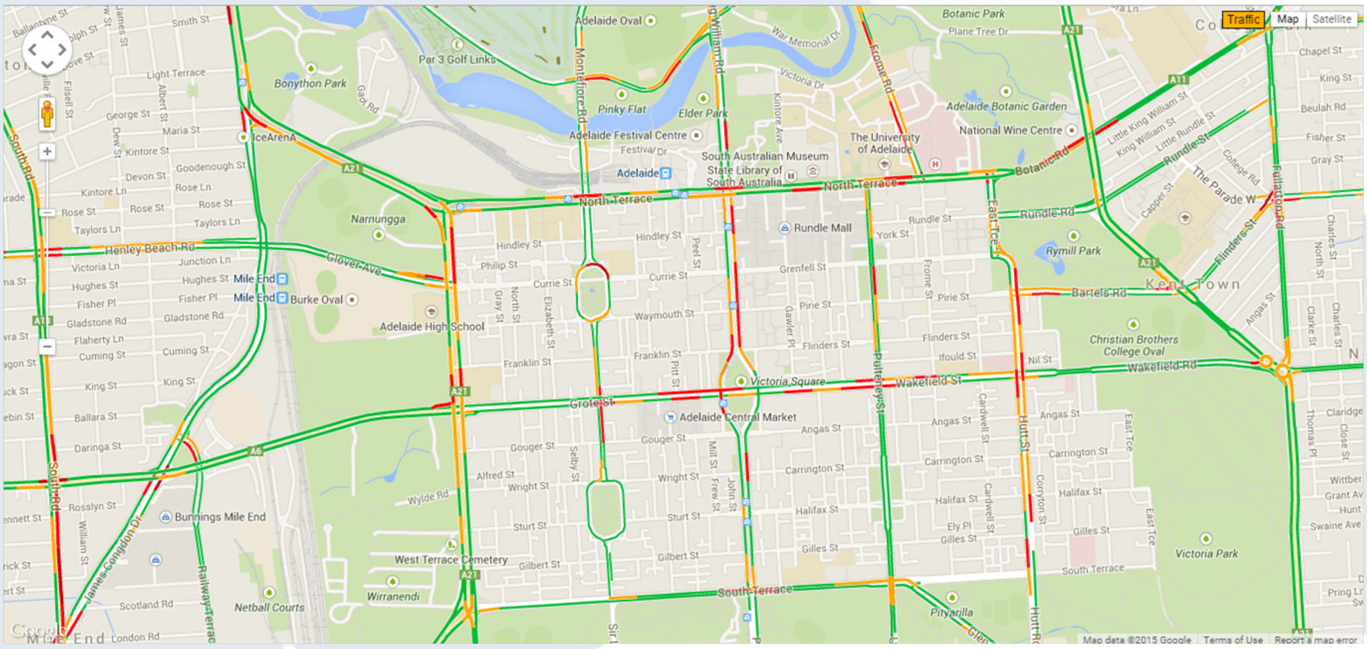
d. You will notice that the button does not currently do anything when it is clicked. We want this button to create a new traffic layer and display it when the button is clicked. To do this we must add an event listener to the **TrafficControl()** function.

```
var map;
var trafficLayer = new google.maps.TrafficLayer();
```

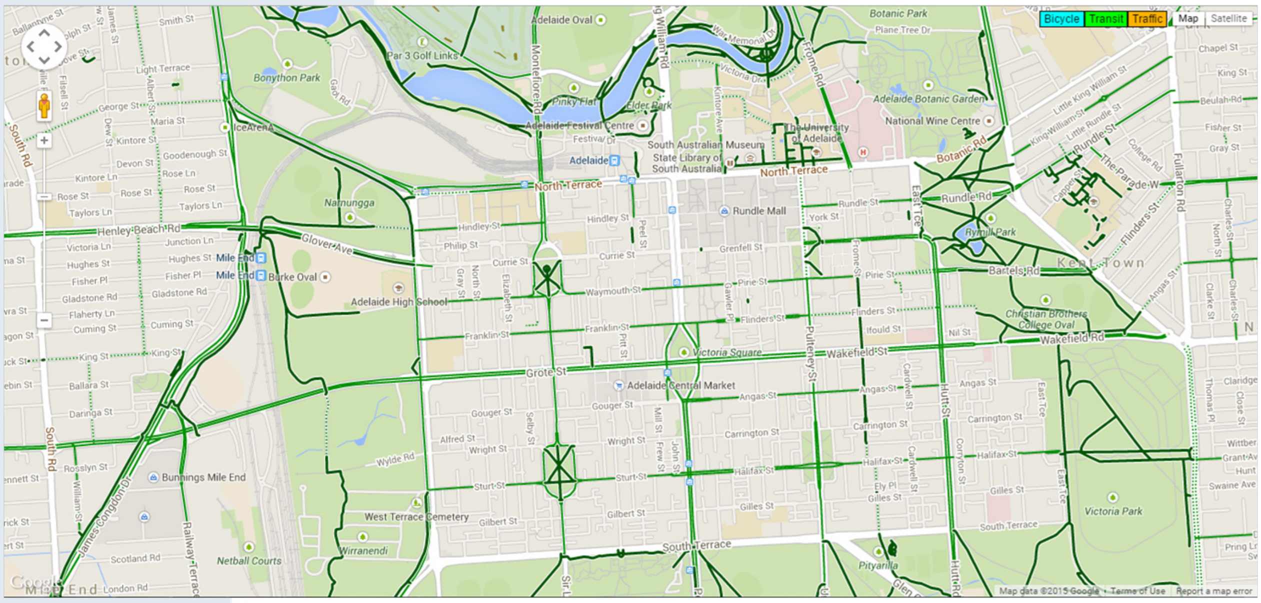
```
    tControlText.style.paddingTop = '1px';
    tControlText.innerHTML = 'Traffic'
    tControlUI.appendChild(tControlText);

    google.maps.event.addDomListener(tControlUI, 'click', function(){
        trafficLayer.setMap(map);
    });
}
```

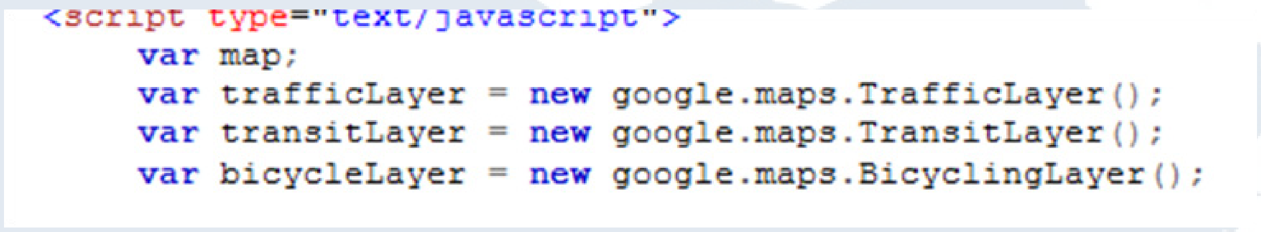
3. Check that your button shows the real time traffic information when the user clicks the traffic button. Your map should look similar to the map on the following page.



4. Repeat step 2 to create similar buttons which allow the user to show the Transit layer and the BicyclingLayer.
   *Note: If you want the buttons to appear in a row then you need to put them in separate divs.

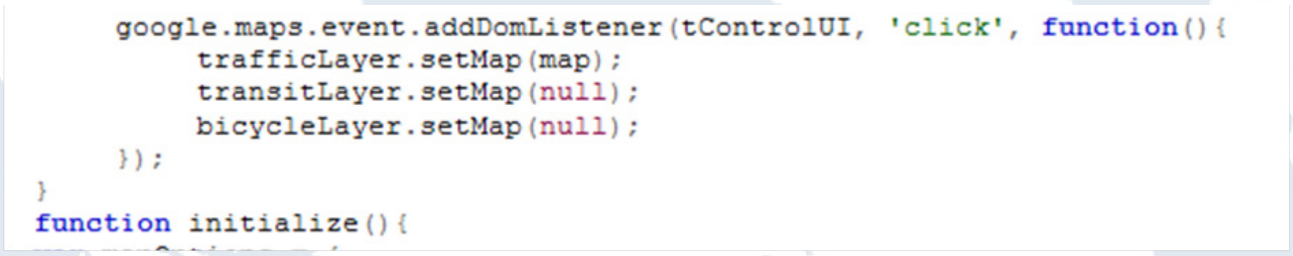


5. You will notice that when you click the buttons to show each layer it overlays the previous one. To stop this we are going to modify our code to hide all layers except for the one we want.
   a. First we want to move the variables for each of the layers to a place where all functions can access them.



```
<script type="text/javascript">
        var map;
        var trafficLayer = new google.maps.TrafficLayer();
        var transitLayer = new google.maps.TransitLayer();
        var bicycleLayer = new google.maps.BicyclingLayer();
```

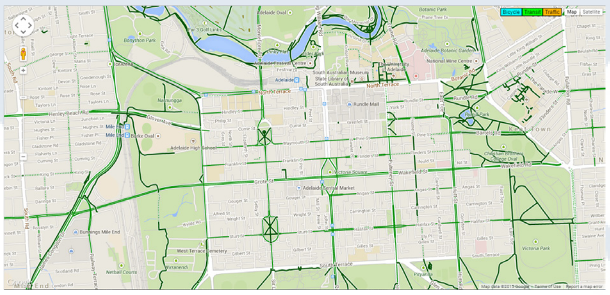
b. To remove a layer you need to set the layer's map to null. In TrafficControl() you would need the following code.

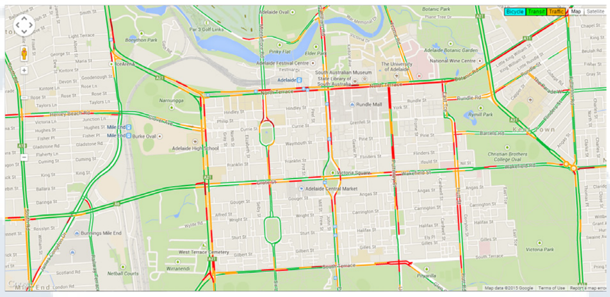


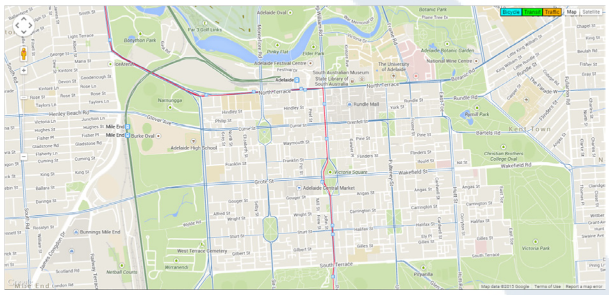
```
    google.maps.event.addDomListener(tControlUI, 'click', function(){
        trafficLayer.setMap(map);
        transitLayer.setMap(null);
        bicycleLayer.setMap(null);
    });
}
function initialize(){
```

c. Repeat the step above for both TransitControl() and BicycleControl()

You should now have a webpage that produces a map overlayed with each of the following


Bicycling


Traffic


Transit

Extension

1. Modify your functions or add a button that allows the user to remove all the layers.
2. Investigate how you could include a heatmap layer and what data you will need to do this.

Activity 4: Loading a KML file

Aim: To learn about how to load and display the data from a kml file.

Task: Follow the steps below to create a map that displays the approximate boundaries of each Australian state.

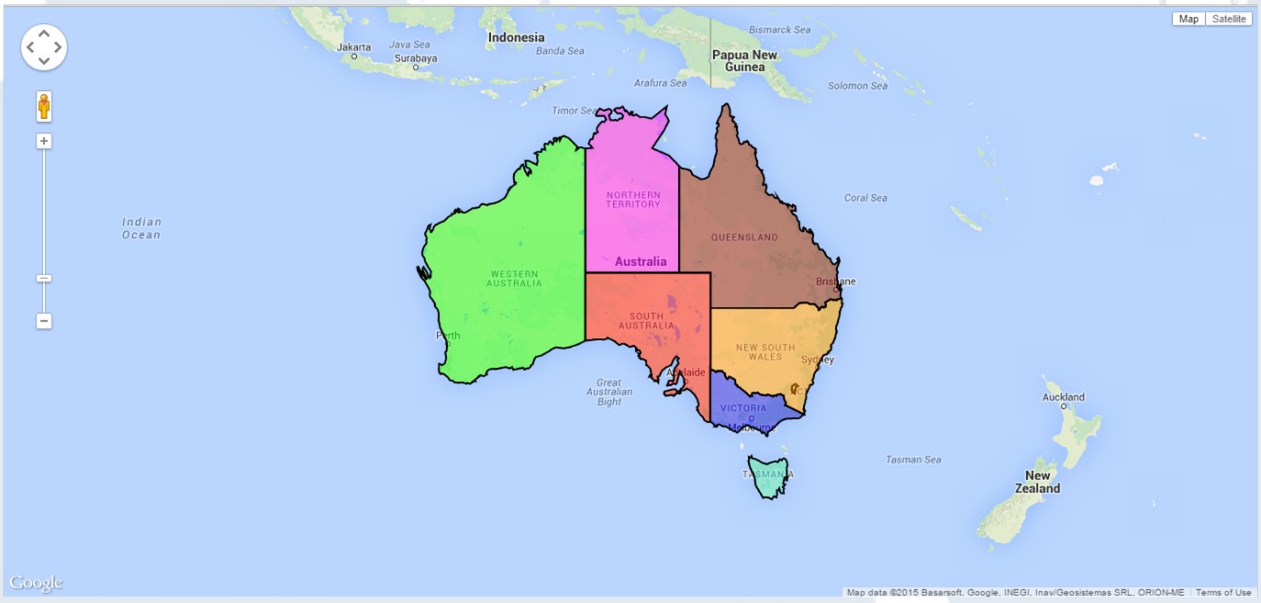
1. Create a basic map of Australia which is centered at lat:-28, lng: 137.883 with zoom 4.

2. In the **`initialize()`** function, add the code below to load the [states kml file]().
   *Note: To use a kml file it must be publically accessible on the internet. When you create your own kml files I suggest creating a GoogleSites page where you can upload your files and link to them.

```
        var georssLayer1 = new google.maps.KmlLayer({
            url: 'https://sites.google.com/site/clhkmlsites/KML-Files/ColouredStates.kml?
            attredirects=0&d=1//https://sites.google.com/site/clhkmlsites/KML-Files/
            states.kml?attredirects=0&d=1'
        });
        georssLayer1.setMap(map);

    }
    google.maps.event.addDomListener(window, 'load', initialize);
```

You should now have a map that looks like the one below.

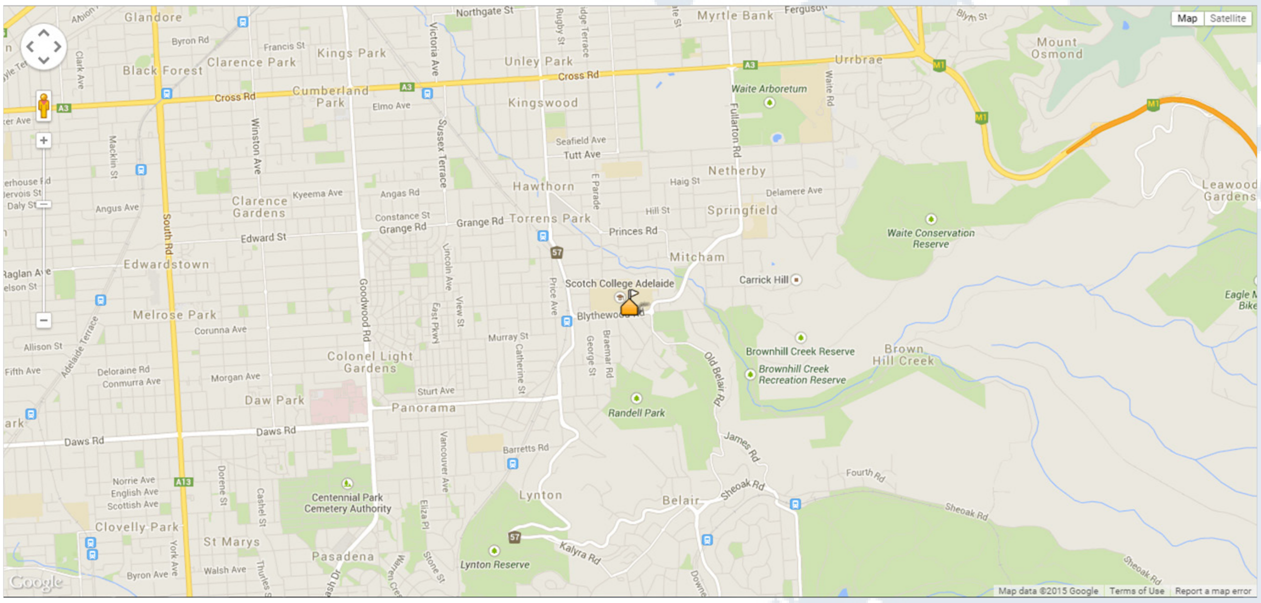


Class/Individual Project

Aim: To load the class data from kml files to show where we were born, have lived and have visited.

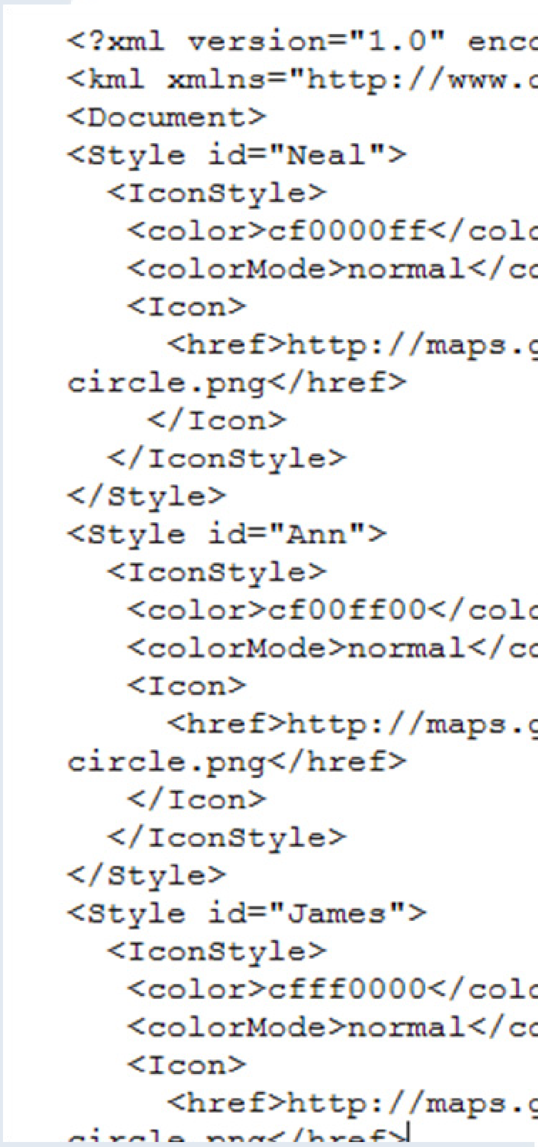
Task: Use your knowledge of loading kml files to display the data provided by the students in your class.

1. Create a basic html file that will load and show your school's location kml file. You should now see a school icon over the location of your school.



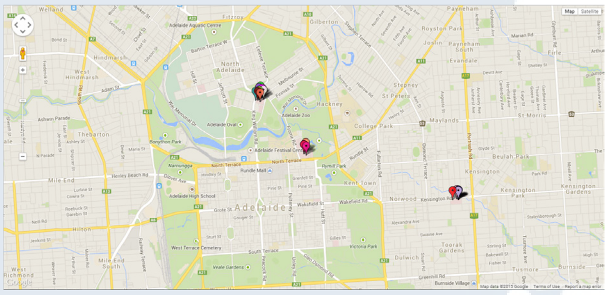
2. Using http://www.mapsdata.co.uk/online-file-converter/#kml-instructions convert the birthplaces spreadsheet to a kml file.

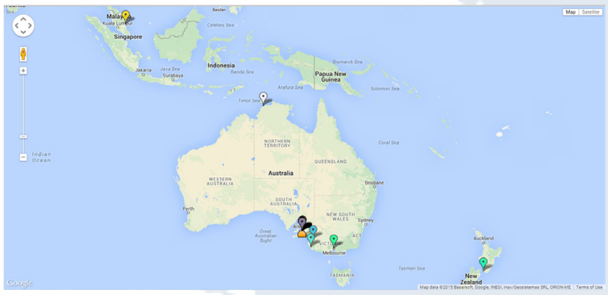
3. You need to edit this kml to include the icons colours to identify each student using the colours they have chosen in the colour spreadsheet. Each student will need to have their own icon colouring code similar to the code on the following page.
   *Note: To colour a marker through the kml file means that you overlay a colour over the base marker. Therefore, you need to use a white base marker so that the colours show correctly.
   *Note: If there are several students with the same name in the class then you may have to include their last name in the id as each id should be unique.



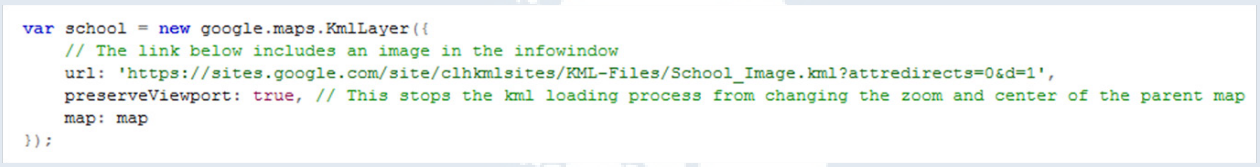
```
<?xml version="1.0" encoding="utf-8" ?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Style id="Neal">
  <IconStyle>
   <color>cf0000ff</color>
   <colorMode>normal</colorMode>
   <Icon>
     <href>http://maps.google.com/mapfiles/kml/paddle/wht-
circle.png</href>
    </Icon>
  </IconStyle>
</Style>
<Style id="Ann">
  <IconStyle>
   <color>cf00ff00</color>
   <colorMode>normal</colorMode>
   <Icon>
     <href>http://maps.google.com/mapfiles/kml/paddle/wht-
circle.png</href>
   </Icon>
  </IconStyle>
</Style>
<Style id="James">
  <IconStyle>
   <color>cfff0000</color>
   <colorMode>normal</colorMode>
   <Icon>
     <href>http://maps.google.com/mapfiles/kml/paddle/wht-
circle.png</href>
```

4. Load the birthplaces kml in the **`initialize()`** function of your html.





*Note: Loading a kml file can change the centre and zoom level of a map. If you want to keep the same centre and zoom, you need to add the code below to preserve the viewport.



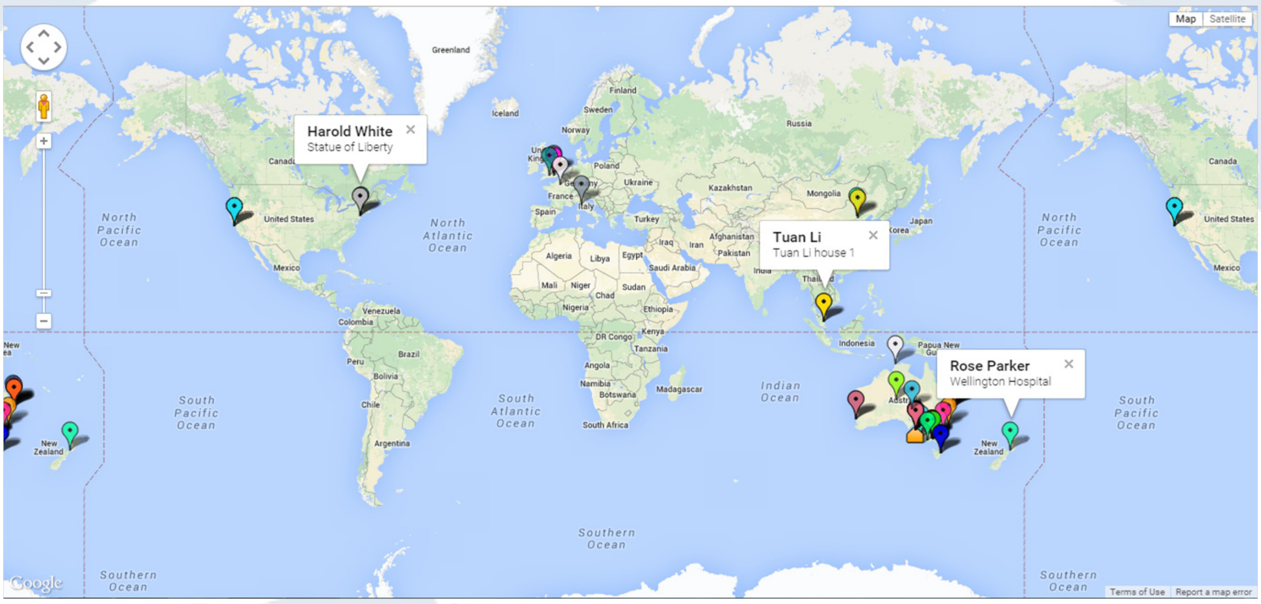
```
var school = new google.maps.KmlLayer({
    // The link below includes an image in the infowindow
    url: 'https://sites.google.com/site/clhkmlsites/KML-Files/School_Image.kml?attredirects=0&d=1',
    preserveViewport: true, // This stops the kml loading process from changing the zoom and center of the parent map
    map: map
});
```

*Note: There can be issues with local caches updating kml files, which makes it seem as though changes to your kml file have made no difference. Therefore, if you change the kml file give it a different name before loading to avoid unnecessary stress.

5. Using the online converter from step 2, convert both the lived and visited spreadsheets into kml files.

6. Just as you edited the birthplaces kml you also need to edit the lived and visited kmls to include the icon colours for each student.

7. Load both the lived and visited kmls in the **`initialize()`** function of your html.

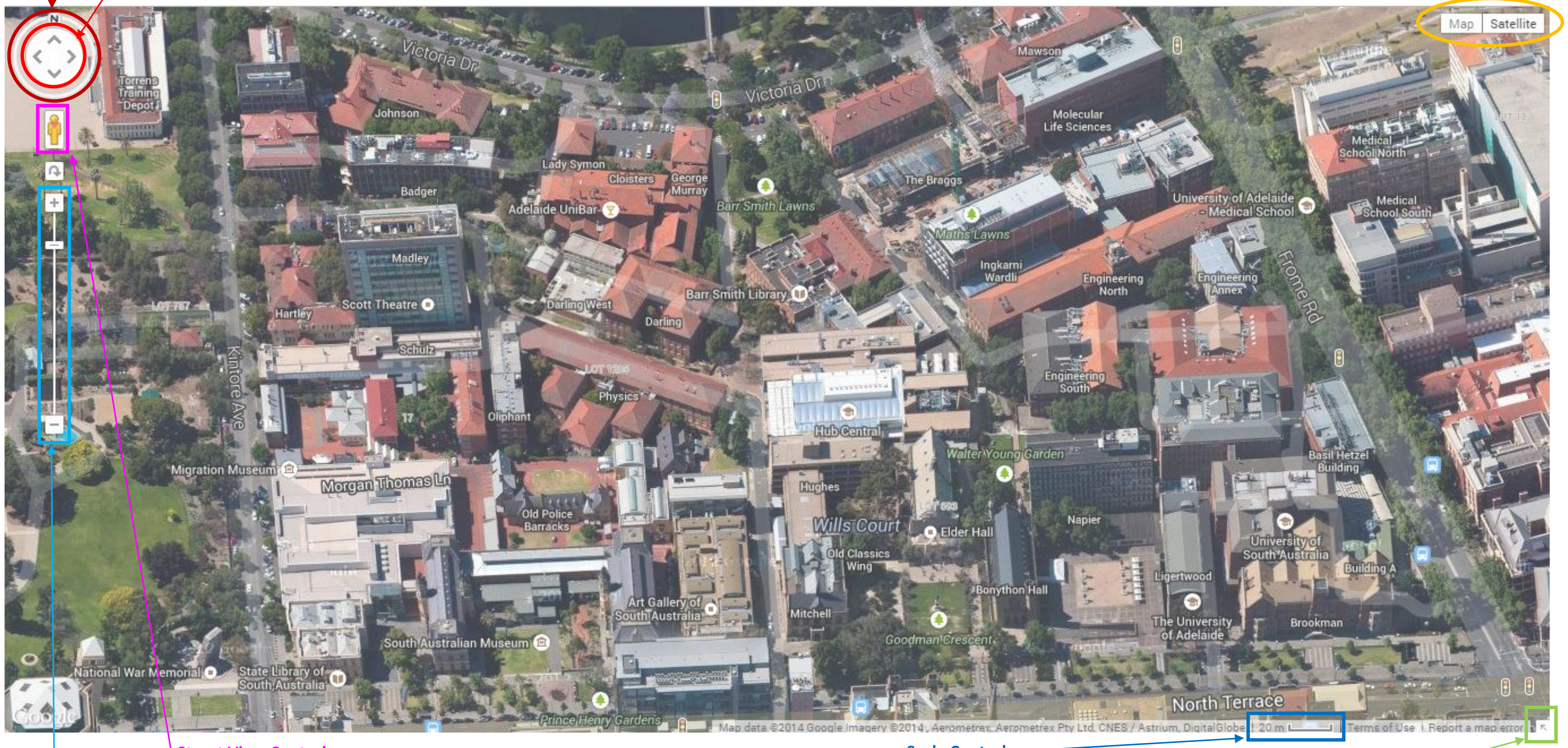


*Note: You should be able to click on any of the icons to bring up an infowindow which displays the student's name and the description you have included in the spreadsheet.

Rotate Control
Contains a small circular icon which allows you to rotate maps containing oblique (45°) imagery. This control appears by default in the top left corner of the map.

Pan Control
Displays buttons for panning the map. This control appears by default in the top left corner of the map on non-touch devices. The Pan control also allows you to rotate 45° imagery, if available.

Map Type Control
Lets the user toggle between map types and appears by default in the top right corner of the map.
HORIZONTAL_BAR: displays the array of controls as buttons in a horizontal bar as is shown below.
DROPDOWN_MENU: displays a single button control allowing you to select the map type via a dropdown menu.
DEFAULT: displays the "default" behaviour, which depends on screen size.


Map
Satellite
N
Victoria Dr
Kintore Ave
Frome Rd
North Terrace
Map data ©2014 Google Imagery ©2014 , Aerometrex, Aerometrex Pty Ltd, CNES / Astrium, DigitalGlobe
20 m
Terms of Use
Report a map error
Google

Street View Control
Can be dragged onto the map to enable Street View and by default appears in the top left corner of the map.

Scale Control
States the scale of the map. This control is not enabled by default.

Zoom Control
Displayed as a slider (for large maps) or small "+/-" buttons (for small maps) to control the zoom level of the map. This control appears by default in the left corner of the map on non-touch devices or in the bottom left corner on touch devices.
Options
SMALL: displays a mini-zoom control, consisting of only **+** and **-** buttons
LARGE: displays the standard zoom slider control
DEFAULT: picks an appropriate zoom control based on the map's size and the device on which the map is running.

Overview Map Control
Displays a thumbnail overview map reflecting the current map viewpoint within a wider area. This control appears by default in the bottom right corner of the map, and is by default shown in its collapsed state.

Student 1 660000	Student 2 0000FF	Student 3 FF9900	Student 4 FFFFFF	Student 5 663300
Student 6 CC3300	Student 7 9AFA00	Student 8 009900	Student 9 660066	Student 10 CC33FF
Student 11 6600FF	Student 12 330099	Student 13 FF00CC	Student 14 0066FF	Student 15 00FFFF
Student 16 00CC99	Student 17 66CCFF	Student 18 003300	Student 19 00FF00	Student 20 3C14FF
Student 21 CCFF00	Student 22 FF99CC	Student 23 FF9999	Student 24 9999FF	Student 25 99CC00
Student 26 808000	Student 27 FF0000	Student 28 006400	Student 29 FFFF00	Student 30 CD0000
Student 31 800080	Student 32 99FF99	Student 33 9314FF	Student 34 CC0066	Student 35 000080
Student 36 003366	Student 37 00D7FF	Student 38 000000	Student 39 CC6633	Student 40 808080

```
<?xml version="1.0" encoding="utf-8" ?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Style id="Neal">
  <IconStyle>
   <color>cf0000ff</color>
   <colorMode>normal</colorMode>
   <Icon>
     <href>http://maps.google.com/mapfiles/kml/paddle/wht-
circle.png</href>
    </Icon>
  </IconStyle>
</Style>
<Style id="Ann">
  <IconStyle>
   <color>cf00ff00</color>
   <colorMode>normal</colorMode>
   <Icon>
     <href>http://maps.google.com/mapfiles/kml/paddle/wht-
circle.png</href>
   </Icon>
  </IconStyle>
</Style>
⁞
More styles
⁞
<Folder>
  <Placemark>
      <name>Neal Bryan</name>
     <styleUrl>#Neal</styleUrl>
      <description>Burnside War Memorial Hospital</description>
      <Point>
        <coordinates>138.63763,-34.927559</coordinates>
     </Point>
  </Placemark>
  <Placemark>
      <name>Ann Clark</name>
     <styleUrl>#Ann</styleUrl>
      <description>Women's &amp; Children's Hospital</description>
      <Point>
        <coordinates>138.600113,-34.910868</coordinates>
      </Point>
  </Placemark>
  ⁞
  More Placemarks
  ⁞
</Folder>
</Document>
</kml>
```

Comment [C1]: Each student needs to have their own style. This allows them to have a uniquely identifiable pin on the map. This is the most time consuming part of the setup for this activity, especially if you let the students choose their own colours. Give the style an id that is unique to each student an that they can quickly identify, e.g. their first name or first_last name…

Comment [C2]: A student's unique colour.

Comment [C3]: The colour is only an overlay so you need to use a white pin as the base for the colours to show correctly.

Comment [C4]: Each student is given their own placemark. They will change the highlighted sections below.

```
<?xml version="1.0" encoding="utf-8" ?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Style id="School">
  <IconStyle>
    <Icon>
      <href>
        http://maps.google.com/mapfiles/kml/shapes/schools.png
      </href>
    </Icon>
  </IconStyle>
</Style>
<Folder>
  <Placemark id="Scotch College, Adelaide">
     <name>Scotch College, Adelaide</name>
     <styleUrl>#School</styleUrl>
     <description>Our School</description>
     <Point>
       <coordinates>138.617022,-34.982581</coordinates>
     </Point>
  </Placemark>
</Folder>
</Document>
</kml>
```

Comment [C1]: Link to the image used for the school icon. You don't need this style section but it just makes it easier to identify the school from the other pins later on.

Comment [C2]: Change the highlighted sections below to fit your school