**Introduction**

So far, we have looked at creating maps and adding markers and infowindows to maps in Google Maps. This week, we look at various types of overlay.

**A recap of earlier work**

In the first lecture on GoogleMaps, we started by looking at the history and general features of Google Maps. The service was introduced  in 2005, and has grown rapidly in popularity since then.  The basic map data, and the framework for placing markers on the maps, use a Mercator projection, based on the WGS84 datum, the same datum used by many applications with a worldwide scope, including the Global Positioning System.  GoogleMaps supports a number of map views, including Map (the default view) view, Satellite view, Terrain view, Street view and Traffic view. Traffic view is a derivative of the default Map view, in which a number of major roads have their shading and line-style varied according to real-time traffic flow data.  Street view is also derived from Map view: in selected cities some streets are highlighted - indicating that street view data are available - and clicking on such a street leads to a pop-up window being created that shows an eye-level view at that point, which is pan-able through 360°.

**Generating bespoke maps**

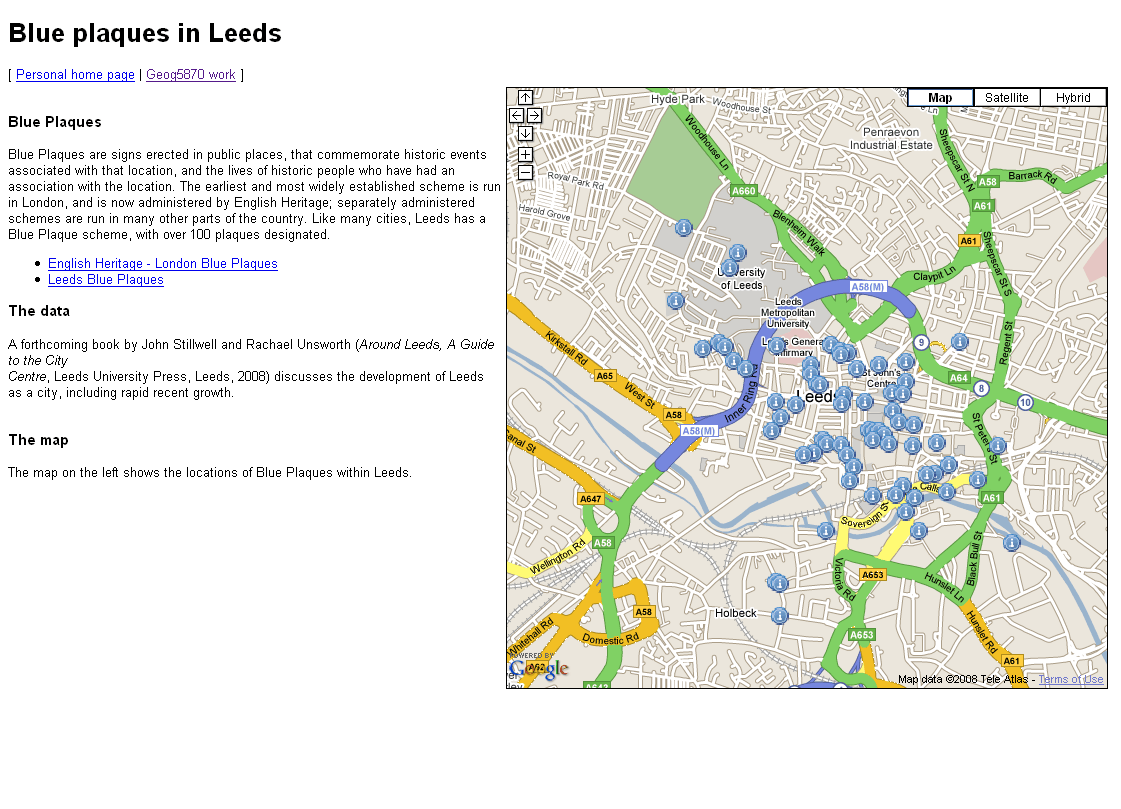
There are a number of ways of generating bespoke maps using Google Maps. A simple approach is to create a URL that calls a GoogleMaps server and includes a parameter that sets the map centre.  It is also possible to give a map a URL as a parameter. If that URL references a file containing KML format marker data, then those markers will be drawn on the map created by Google Maps. Both these approaches are fairly quick and easy, but are of limited use, although they may be an acceptable solution for illustrating the location of something in an ad hoc fashion, especially if one does not have access to a web server on which one can create and make available files.

A far more powerful and flexible way of creating bespoke maps is to use the GoogleMaps API. Google Maps is based around a Javascript interface that draws map data from a server, and presents it on screen with various control functions. By using the API, you can extend the Javascript map interface in many ways, adding your own data as overlays and modifying the appearance of the map.

In the second lecture, we looked at ways of extending the basic map in order to be able to present more useful information. This was demonstrated by adding some simple markers to the map, with pop-up information windows. At the same time, the growing size of the html page associated with this led to a need to clean up the code, and to separate out elements on a functional basis: individual files were created and used to contain the basic html code, the map loading logic, and a data file.

The practical session associated with the second lecture focussed on a more substantive task: creating a map with a set of markers and with pop-up information linked to each marker. The markers related to the locations of Blue Plaques in and around central Leeds. The task presented two major problems that are characteristic of similar real-world tasks: first, extracting the data from a spreadsheet in a usable format, and second, converting the co-ordinate data into the required format.

The resulting page should have looked something like the shown in Figure 1, although it is possible to have included much richer background information, and to have presented the map in a more appealing manner.

[](https://vlebb.leeds.ac.uk/bbcswebdav/users/geo6odw/geog58705871_2010-11/wk8/wk8_files/fig_1.gif)Figure 1: Presenting information about Blue Plaques in Leeds using Google Maps