# GOOGLEMAP2

The GOOGLEMAP2 control is used to provide for Google Maps support within Application Designer pages. The control internally makes use of the Google Maps API. In order to use the control on your site, you need to sign up for a Google Maps API key at *http://code.google.com/apis/maps/signup.html*. Make sure that you agree with the Google Maps API Terms of Use (*http://code.google.com/apis/maps/terms.html*).

The following topics are covered below:

- Before You Start
- Example
- Typical Problems
- Properties

---

## Before You Start

In order to use the GOOGLEMAP2 control, you need to sign up for a Google Maps API key. A key is valid for a single "directory" on your web server only, i.e. you sign up for a URL like *http://www.mysite.com/mywebapp/myproject*. With a standard installation of Application Designer on localhost, you may sign up for the URL *http://localhost:8080/mywebapp/myproject*. Typically, you develop your Application Designer web application not on the site on which you run it later in productive mode. Therefore, you may sign up for two different sites (development and production site).

### Required Steps

1. Choose the project directory that keeps the layouts using the GOOGLEMAP2 control.
2. Sign up for a Google Maps API key at *http://code.google.com/apis/maps/signup.html* for this project directory (e.g. *http://localhost:8080/mywebapp/myproject*).
3. Create the API key page. Store the key page in the registered project directory. You are free in naming the file (the file extension must be "html"). The GOOGLEMAP2 control embeds your API key as a subpage. The subpage must have the following minimum structure:

```
<html>
  <head>
    <script src=" http://maps.google.com/maps?file=api&amp;v=2.x&amp;key=YOUR_API_KEY"></script>
        <script src="../HTMLBasedGUI/general/googlemapsscript.js"></script>
  </head>
  <body>
    <div id="map" style="position:absolute; top0; left:0;"></div>
  </body>
</html>
```

   You see that the page includes two JavaScript libraries. The first line refers to the Google Maps API. Replace the placeholder "YOUR_API_KEY" with your Google Maps API key. With the second line, the page includes the control's scripting (calls from Application Designer to the Google Maps). The page body is quite simple: it contains a single `div` tag with the ID "map". This `div` is used as an anchor to insert Google Maps controls dynamically.

# Example

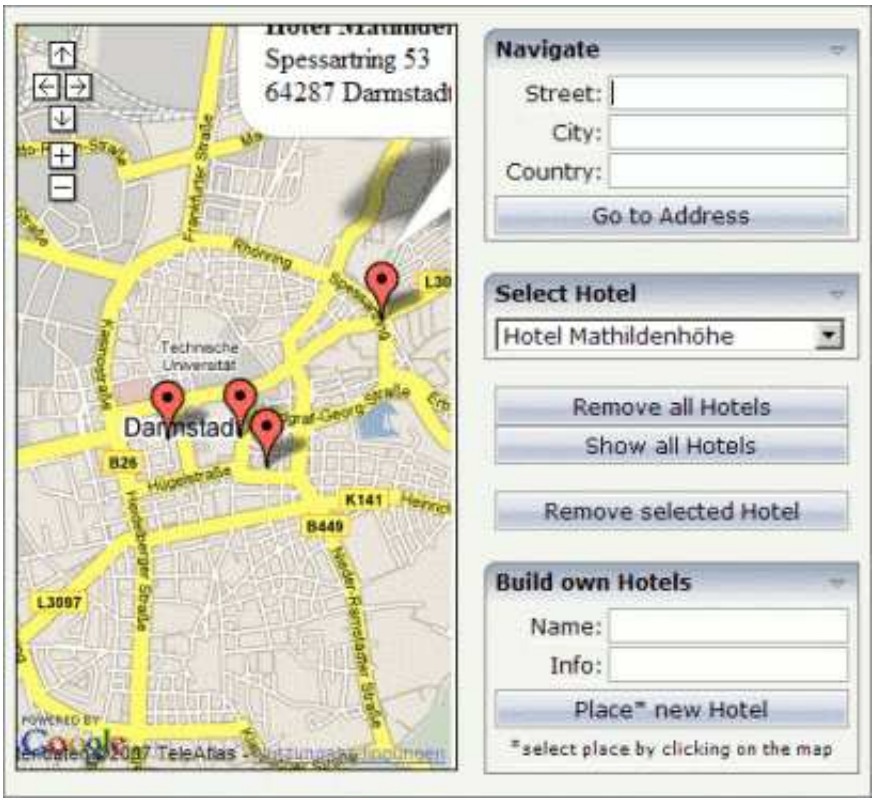
The following topics are covered below:

- General Usage
- Marker Management

## General Usage

The map options are taken from the property `infoprop`. On this object, you may set the address (or latitude and longitude), the zoom level and the map size as well as the map type.

**Note:**
The usage of address or longitude/latitude is mutually exclusive.




The above map is controlled by the following adapter code:

```
public class GoogleMap2Adapter extends Adapter
{
  // property >gm2Info<
  GOOGLEMAP2Info m_gm2Info = new GOOGLEMAP2Info(
                               GOOGLEMAP2Info.NO_MAPTYPE_CONTROL,
                               GOOGLEMAP2Info.SMALL_MAP);
  public GOOGLEMAP2Info getGm2Info(){ return m_gm2Info; }
  public void setGm2Info(GOOGLEMAP2Info value){ m_gm2Info = value; }
  // Marker items
  private class HotelMarker extends GOOGLEMAP2Item
  {
    // [see section "Marker Management"]
```

```
  ....
}

private Hashtable hotels = new Hashtable();

/** initialisation - called when creating this instance */
public void init()
{
  m_gm2Info.setAddress("Darmstadt, Germany");
  m_gm2Info.setZoomlevel("13");
  setupHotels();
}

// property >hotelSelection<
String m_hotelSelection = "";
public String getHotelSelection(){ return m_hotelSelection; }
public void setHotelSelection(String value){ m_hotelSelection = value; }

// property >validHotSel<
COMBODYNValidValues m_validHotSel = new COMBODYNValidValues();
public COMBODYNValidValues getValidHotSel(){ return m_validHotSel; }

/** */
public void onSelect()
{
  HotelMarker hotel = (HotelMarker) hotels.get(m_hotelSelection);
  m_gm2Info.centerMarker(hotel);
}

// property >hotelDesc<
String m_hotelDesc = "";
public String getHotelDesc(){ return m_hotelDesc; }
public void setHotelDesc(String value){ m_hotelDesc = value; }

// property >hotelName<
String m_hotelName = "";
public String getHotelName(){ return m_hotelName; }
public void setHotelName(String value) { m_hotelName = value; }

/** */
public void onPlaceOwn()
{
  if (m_hotelName.equals(""))
  {
    outputMessage(MT_ERROR, "Please specify a name.");
    return;
  }

  HotelMarker MyHotel = new HotelMarker(m_hotelName);
  MyHotel.setInfoText("<b>" + m_hotelName + "</b>\n" + m_hotelDesc);

  m_gm2Info.addMarkerToLastSelectedPoint(MyHotel);
}

/** */
public void onRemove()
{
  m_gm2Info.removeLastSelectedMarker();
}

/** */
public void onRemoveAll()
```

```
{
  m_gm2Info.clear();
  hotels.clear();
  m_validHotSel.clear();
  m_hotelSelection = "";
}

/** */
public void onShowAll()
{
  onRemoveAll();
  setupHotels();
}

private void setupHotels()
{
  setupHotel("Bestwestern, Parkhaus-Hotel",
      "Grafenstraße 31, 64283 Darmstadt");
  setupHotel(" ....
  // deactivate last added marker
  m_gm2Info.setSelectedMarker(null);
}

private void setupHotel(String name, String address)
{
  HotelMarker hotel = new HotelMarker(name, address);
  hotel.setInfoText("<b>" + name + "</b>\n" + address.replaceAll(", ", "\n"));
  m_gm2Info.addMarker(hotel, false);
  if (name.length() > 23)
    name = name.substring(0, 23) + "...";
  m_validHotSel.addValidValue(String.valueOf(hotel.getId()), name);
  hotels.put(String.valueOf(String.valueOf(hotel.getId())), hotel);
}

// property >naviCity<
String m_naviCity;
public String getNaviCity(){ return m_naviCity; }
public void setNaviCity(String value){ m_naviCity = value; }

// property >naviCountry<
String m_naviCountry;
public String getNaviCountry(){ return m_naviCountry; }
public void setNaviCountry(String value){ m_naviCountry = value; }

// property >naviStreet<
String m_naviStreet;
public String getNaviStreet(){ return m_naviStreet; }
public void setNaviStreet(String value){ m_naviStreet = value; }

/** */
public void onNavigate()
{
  String address = "";

  if (!m_naviStreet.equals(""))
  {
    address += m_naviStreet + ", ";
  }
  if (!m_naviCity.equals(""))
  {
    address += m_naviCity + ", ";
  }
```

```
    if (!m_naviCountry.equals(""))
    {
      address += m_naviCountry;
    }

    m_gm2Info.setAddress(address);
  }

  ....
}
```

The above map is initialized with the instantiation of the `GOOGLEMAP2Info` object and just a few simple lines of code in the `init()` method.

The constructor of the `GOOGLEMAP2Info` class takes the following arguments:

- **Map Type Control Setting**
  Using the constant "MAPTYPE_CONTROL" (instead of "NO_MAPTYPE_CONTROL" which is used in the above example) would result in three buttons in the upper right corner of the map, which enable the user to change the map view between "Map", "Satellite" and "Hybrid" mode.

  **Note:**
  The range of zoom levels may differ for different map types in the same region.

- **Map Size Setting**
  For the above map the map size property is set to the constant "SMALL_MAP" which results in the four navigation arrows and the zoom buttons in the upper left corner. The constant "LARGE_MAP" would alternatively provide more precise navigation controls allocating more of the map area in exchange.

The `GOOGLEMAP2Info` class provides for a second constructor without any arguments. Using this constructor is equal to the usage of the described constructor with the constants "NO_MAPTYPE_CONTROL" and "SMALL_MAP".

In the `init()` method, the map view is positioned via the `setAddress` method. The same result would be achieved using the `setLatLng` method with the argument "49,879046" (for latitude) and "8,670112" (for longitude). It is obligatory to set the map view using one of these variants or using a marker (see *Marker Management* for further information). Otherwise the map will not be displayed.

The range of values for the `zoomlevel` property may vary according to the map region. The value "4" is used by default if `zoomlevel` is not set explicitly.

The GOOGLEMAP2 control listens to changes on the address (or latitude/longitude) and the `zoomlevel` property.

## Marker Management

To use the marker management of the GOOGLEMAP2 control, you need an implementation of the `GOOGLEMAP2Item` class. For the above example, the following code was used:

```
private class HotelMarker extends GOOGLEMAP2Item
{
  private String m_name;

  public HotelMarker(String name)
  {
```

```
    super(true);
    m_name = name;
  }

  public HotelMarker(String name, String address)
  {
    super(address, true);
    m_name = name;
  }

  public void reactOnSelect()
  {
    outputMessage(MT_SUCCESS, "Hotel '" + m_name + "' selected.");
  }

  public void reactOnDrag()
  {
    outputMessage(MT_SUCCESS, "Hotel '" + m_name + "' has moved.");
  }

  public void reactOnDeactivate()
  {
    outputMessage(MT_SUCCESS, "Hotel '" + m_name + "' deselected.");
  }
}
```

The methods `reactOnSelect()`, `reactOnDrag()` and `reactOnDeactivate()` have to be implemented in order to define the behavior for the following events:

- **Select**
  The user clicks on the corresponding marker on the map.

  If the user clicks the button, for example, five times, this event is fired five times, even if the button remains active.

- **Drag**
  The user drags the marker to a different position on the map and drops it.

  It is possible to switch dragging on and off for each marker using the marker's `setDraggable(boolean)` method. By default, all markers are draggable.

- **Deactivate**
  The user clicks a different marker or somewhere else on the map when the marker is active.

  A marker is considered active from selection until deactivation.

Markers may be added to specific positions or to the position the user has clicked last on the map, removed, activated, deactivated or centered using the `infoprop` property. As mentioned above in the section *General Usage*, a marker may be used to set the map's view if it is told to center on the marker.

Each marker may have an `infoText` that is shown within the pop-up when the marker is selected by the user. Changes to this text will be updated on the client side. If no text is set, a pop-up will not appear. Since the `infoText` is treated as HTML code, it may be formatted like HTML. Only breaks will automatically be replaced.

The GOOGLEMAP2 control listens to changes on markers, address (or latitude/longitude) and `infoText` property.

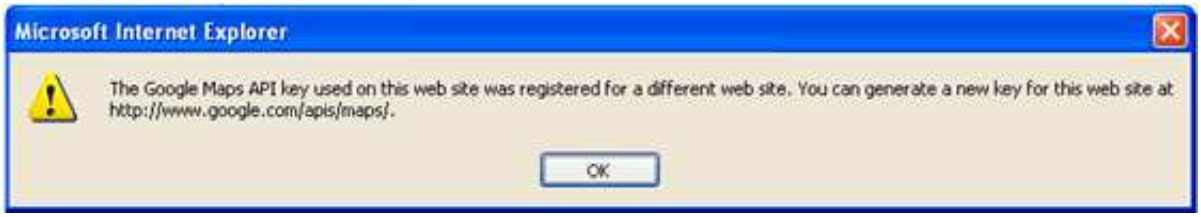
# Typical Problems

The following topics are covered below:

- Google Map API Key
- Map Remains Gray

## Google Map API Key

Your Google Maps API key is bound to a directory on a certain web server (i.e. you sign up for the URL *http://mycomputer.mydomain.com:8080/mywebapp/myproject*). If you use your key for another URL, Google shows an error message:




Reasons that cause the error:

- You have registered your computer using the computer's name (e.g. *http://mycomputer...*). But the Application Designer development workplace is started using the URL *http://localhost....*

  Solution: start the Application Designer workplace with *http://mycomputer....*

- The registered directory (e.g. *.../mywebapp/myproject*) does not match your installation (either a mistake in writing when signing up for the key or you have renamed the web application or project after registration).

  Solution: rename your web application or project to match the registered names. Or sign up for a new key and insert the new key into the API key page. In the latter case, delete the content of the browser's cache. Otherwise, the browser will use the former API key page (and thus the old key).

## Map Remains Gray

If you use longitude and latitude for placing the marker on the map, their values may exceed the map top (or bottom) border. If you are able to find the map by scrolling down (or up), then this is the case. Check the values for longitude and latitude in this case.

# Properties

| Basic |
|---|

| | | | |
|---|---|---|---|
| infoprop | Name of adapter property representing the control on server side.<br><br>The property must be of type GOOGLEMAPInfo. Read further information inside the Java API Documentation. | Obligatory | |
| apikeypagename | Name of the Maps API Key page. Example: mygooglemapsapikey.html. Keep this file within the project directory (directory within the CIS HTML pages are kept). The GOOGLEMAP-control expects this file within certain Javascript includes and content. Have look into chapter "Google Map - Before You Start" within the Developers Guide | Obligatory | |
| width | Width of the control.<br><br>There are three possibilities to define the width:<br><br>(A) You do not define a width at all. In this case the width of the control will either be a default width or - in case of container controls - it will follow the width that is occupied by its content.<br><br>(B) Pixel sizing: just input a number value (e.g. "100").<br><br>(C) Percentage sizing: input a percantage value (e.g. "50%"). Pay attention: percentage sizing will only bring up correct results if the parent element of the control properly defines a width this control can reference. If you specify this control to have a width of 50% then the parent element (e.g. an ITR-row) may itself define a width of "100%". If the parent element does not specify a width then the rendering result may not represent what you expect. | Optional | 100<br><br>120<br><br>140<br><br>160<br><br>180<br><br>200<br><br>50%<br><br>100% |
| height | Height of the control.<br><br>There are three possibilities to define the height:<br><br>(A) You do not define a height at all. As consequence the control will be rendered with its default height. If the control is a container control (containing) other controls then the height of the control will follow the height of its content.<br><br>(B) Pixel sizing: just input a number value (e.g. "20").<br><br>(C) Percentage sizing: input a percantage value (e.g. "50%"). Pay attention: percentage sizing will only bring up correct results if the parent element of the control properly defines a height this control can reference. If you specify this control to have a height of 50% then the parent element (e.g. an ITR-row) may itself define a height of "100%". If the parent element does not specify a width then the rendering result may not represent what you expect. | Optional | 100<br><br>150<br><br>200<br><br>250<br><br>300<br><br>250<br><br>400<br><br>50%<br><br>100% |

| | | | |
|---|---|---|---|
| comment | Comment without any effect on rendering and behaviour. The comment is shown in the layout editor’s tree view. | Optional | |
| Appearance | | | |
| pagestyle | CSS style definition that is directly passed into this control.<br><br>With the style you can individually influence the rendering of the control. You can specify any style sheet expressions. Examples are:<br><br>border: 1px solid #FF0000<br><br>background-color: #808080<br><br>You can combine expressions by appending and separating them with a semicolon.<br><br>Sometimes it is useful to have a look into the generated HTML code in order to know where direct style definitions are applied. Press right mouse-button in your browser and select the "View source" or "View frame’s source" function. | Optional | |
| rowspan | Row spanning of control.<br><br>If you use TR table rows then you may sometimes want to control the number of rows your control occupies. By default it is "1" - but you may want to define the control two span over more than one columns.<br><br>The property only makes sense in table rows that are snychronized within one container (i.e. TR, STR table rows). It does not make sense in ITR rows, because these rows are explicitly not synched. | Optional | 1<br><br>2<br><br>3<br><br>4<br><br>5<br><br>50<br><br>int-value |
| colspan | Column spanning of control.<br><br>If you use TR table rows then you may sometimes want to control the number of columns your control occupies. By default it is "1" - but you may want to define the control to span over more than one columns.<br><br>The property only makes sense in table rows that are snychronized within one container (i.e. TR, STR table rows). It does not make sense in ITR rows, because these rows are explicitly not synched. | Optional | 1<br><br>2<br><br>3<br><br>4<br><br>5<br><br>50<br><br>int-value |