

# Google Earth API

## Balloons

Instructor: Wei Ding  
CS 739 Spatial Data Mining

Reference:

<http://code.google.com/apis/earth/documentation/balloons.html>

1

## Introduction

- Balloons are information windows displayed in the Google Earth Plugin, optionally associated with a feature.
  - Their content can include HTML, CSS, or JavaScript.
- There are three types of Balloons
  - Feature balloons
  - HTML string balloons
  - HTML DIV balloons

2

# Feature balloons

- The basic balloon type in the Google Earth Plugin.
- Do not need to explicitly created.
- Feature balloons can also be created with `ge.createHtmlStringBalloon()`, which allows you to control certain aspects of how the balloon is displayed.

3

## Feature balloons: Example (1/4)

```
var balloon = ge.createHtmlStringBalloon("");  
//The feature with which they are associated  
balloon.setFeature(placemark);  
// Minimum and maximum height and width  
balloon.setMinWidth(400);  
balloon.setMaxHeight(400);  
// Whether or not to include a close button  
balloon.setCloseButtonEnabled(false);
```

*To be continued*

4

# Content scrubbing

- Feature balloons must be attached to a feature.
- Their contents are automatically set to be the scrubbed contents of the feature's <description>element.
- When a feature balloon's description contents are scrubbed, the following code is removed:
  - JavaScript
  - CSS
  - <iframe>tags
  - <embed>tags
  - <object> tags
- Scrubbing the contents is a security measure to prevent malicious code from executing in a balloon.
- However, if the content is known and trusted, there are a few methods available that will bring in all content without stripping any elements.

5

## Feature Balloons: Example (2/4)

- ```
var kmlString = " +
'<?xml version="1.0" encoding="UTF-8"?>' +
'<kml xmlns="http://www.opengis.net/kml/2.2">' +
' <Document>' +
'   <Style id="bstyle">' +
'     <BalloonStyle>' +
'       <text>' +
'         <![CDATA[' +
'           Species is ${species}<br>' +
'           Age is: ${age}' +
'         ]]>' +
'       </text>' +
'     </BalloonStyle>' +
'   </Style>' +
'   <Placemark id="example_placemark">' +
'     <styleUrl>#bstyle</styleUrl>' +
'     <name>Placemark 1</name>' +
'     <description>' +
'       <![CDATA[<a href="#" onclick="alert(\'Running some JavaScript!\');">Alert!</a><br>]]>' +
'     </description>' +
'     <ExtendedData>' +
'       <Data name="species">' +
'         <value>Badger</value>' +
'       </Data>' +
'       <Data name="age">' +
'         <value><![CDATA[<a href="#" onclick="alert(3);"> Click to see</a><br>]]</value>' +
'       </Data>' +
'     </ExtendedData>' +
'     <Point>' +
'       <coordinates>-122.4226,37.7792,0</coordinates>' +
'     </Point>' +
'   </Placemark>' +
' </Document>' +
'</kml>');
```

A CDATA section starts with "<![CDATA[" and ends with "]]>"  
The term CDATA is used about text data that should not be parsed by the KML parser

```
var kmlObject = ge.parseKml(kmlString);
ge.getFeatures().appendChild(kmlObject);
var placemark = ge.getElementById('example_placemark');
```

To be continued

## Feature Balloons: Example (3/4)

```
google.earth.addListener(placemark, 'click', function(event) {  
  // Prevent the default balloon from appearing.  
  event.preventDefault();  
  // Retrieve the feature's description content in its entirety  
  var content = placemark.getDescription();  
  // a new htmlStringBalloon  
  var balloon = ge.createHtmlStringBalloon("");  
  // attach to a feature  
  balloon.setFeature(placemark);  
  // feed it to this new htmlStringBalloon  
  balloon.setContentString(content);  
  ge.setBalloon(balloon);  
});
```

*To be continued*

7

## Feature Balloons: Example (4/4)

```
google.earth.addListener(placemark, 'click', function(event) {  
  // Prevent the default balloon from appearing.  
  event.preventDefault();  
  // use getBalloonHtml() to retrieve the contents of the  
<ExtendedData> element  
  // however, unsafe content will be stripped from the extended data.  
  In this case, the onclick javascript.  
  var content = placemark.getBalloonHtml();  
  // var content = placemark.getBalloonHtmlUnsafe(); will do the trick  
  var balloon = ge.createHtmlStringBalloon("");  
  balloon.setFeature(placemark);  
  balloon.setContentString(content);  
  ge.setBalloon(balloon);  
});
```

8

## In-Class Exercise

- Google Code Playground:  
[http://code.google.com/apis/ajax/playground/#extended\\_data\\_in\\_balloons](http://code.google.com/apis/ajax/playground/#extended_data_in_balloons)
- Task 1: try the 3 methods:
  - getDescription
  - getBalloonHtml
  - getBalloonHtmlUnsafe
- Task 2: Find the latitude and longitude of Boston, display “City is Boston and Age is 100” in two different ways. For example, an alert window and a text window.

9

## HTML string balloons

- Different from feature balloons that must be attached to a feature, HTML string balloons do not need to be attached to a feature.
- If no feature is specified with setFeature(), the balloon will be displayed in the center of the current view.
- HTML string balloons can contain any HTML or CSS markup, or JavaScript code.

10

# HTML string balloons: Example

- [http://code.google.com/apis/earth/documentation/samples/html\\_string\\_balloon\\_example.html](http://code.google.com/apis/earth/documentation/samples/html_string_balloon_example.html)
- ```
google.earth.addEventListener(placemark, 'click', function(event) {  
  // Prevent the default balloon from popping up.  
  event.preventDefault();  
  
  var balloon = ge.createHtmlStringBalloon("");  
  balloon.setFeature(placemark); // optional  
  balloon.setContentString(  
    '<object width="200" height="150"><param name="movie" '  
    + 'value="http://www.youtube.com/v/6mrG_bsqC6k&hl=en&fs=1"/>'  
    + '<param name="allowFullScreen" value="true"/>'  
    + '<embed src="http://www.youtube.com/v/6mrG_bsqC6k&hl=en&fs=1" '  
    + 'type="application/x-shockwave-flash" allowfullscreen="true" '  
    + 'width="200" height="150"></embed></object>');  
  
  ge.setBalloon(balloon);  
});
```

11

## In-Class Exercise

- Google Code Playground:  
[http://code.google.com/apis/ajax/playground/#html\\_string\\_balloons](http://code.google.com/apis/ajax/playground/#html_string_balloons)
- Task: Display a UMass Boston logo in the balloon.

12

# HTML DIV balloons

- HTML DIV balloons use a given <div> element as the content to show inside the balloon. The DIV is reparented to the plugin's internal DOM structure. All HTML, CSS markup, and JavaScript code is acceptable in an HTML DIV balloon.
- HTML DIV balloons must be explicitly created with createHtmlDivBalloon() and displayed with setBalloon(). Content must be added to a DIV element on the page, before being set with setContentDiv().

13

- A simple DIV balloon can be created as follows:

```
var balloon = ge.createHtmlDivBalloon("");
balloon.setFeature(placemark);
var div = document.createElement('DIV');
div.innerHTML = 'Any <em>HTML</em>, CSS, or JavaScript goes here.';
balloon.setContentDiv(div);
ge.setBalloon(balloon);
```

14

# HTML DIV balloons: Example

- The example copies the content from a specified element on the page into a new DIV, then displays that content in a balloon. This is the basis for dynamically generated pages, with placemarks displaying when page links are clicked. In the example below, the placemarks are hard-coded, but they could be generated from data in a database or flat file.
- **Source:** [http://code.google.com/apis/earth/documentation/samples/html div balloon example.html](http://code.google.com/apis/earth/documentation/samples/html%20div%20balloon%20example.html)

15

- Within the initialization callback function (initCB() in our examples):

```
// Create the placemarks and points.
window.placemark0 = ge.createPlacemark("");
var point0 = ge.createPoint("");
point0.setLatitude(12.342);
point0.setLongitude(54.321);
placemark0.setGeometry(point0);
ge.getFeatures().appendChild(placemark0);

// Add an event listener to each of the 'results' list items.
// If/else statement is a fix for IE behavior.
var divs = document.getElementById('results').getElementsByTagName('li');
for (var i = 0; i < divs.length; i++) {
  var singleDiv = divs[i];
  if (singleDiv.addEventListener)
    singleDiv.addEventListener('click', makeClickHandler(divs[i].id), false);
  else if (singleDiv.attachEvent)
    singleDiv.attachEvent('onclick', makeClickHandler(divs[i].id));
}
```

16

- Outside of initCB():

```
function makeClickHandler(id) {
  return function() {
    var balloon = ge.createHtmlDivBalloon("");
    // Attach this balloon to the placemark.
    balloon.setFeature(window['placemark' + id]);
    var div = document.createElement("DIV");
    // Populate the new div with the appropriate list item from the 'results' div.
    div.innerHTML =
document.getElementById('results').getElementsByTagName('li')[id].innerHTML;
    balloon.setContentDiv(div);
    ge.setBalloon(balloon);
  }
}
```

17

- Finally, within the body of the page:

```
<div id="results">
  <p><strong>Show:</strong></p>
  <ul>
    <li id="0">great swimming</li>
    <li id="1">sharks!</li>
    <li id="2">sea lions</li>
  </ul>
</div>
```

18

# In-Class Exercise

- Google Code Playground:  
[http://code.google.com/apis/ajax/playground/#html div balloons](http://code.google.com/apis/ajax/playground/#html%20div%20balloons)
- Task: Display a picture of your choice and the correspondent link in the balloon.