Computación Distribuída

Proyecto MashUp

Manuel Candal Iglesias

14/12/2009

Objetivo

Creación de un Mashup que visualice en Google Maps locales de ocio introducidos por usuarios registrados.

El usuario validado podrá indicar en el mapa de Google Maps diversas localizaciones y consultarlas posteriormente.

A través de un formulario cubrir el nombre de la localización, comentarios sobre el lugar e incluso una imagen asociada.

Materiales

1. Eclipse 3.51 for Java Developers

   Plugins y librerías:

   - Eclipse Java EE Developer Tools 3.1.1
   - Eclipse Web Developer Tools 3.1.1
   - Google App Engine Java SDK 1.2.6
   - JSON Simple 1.1
   - Commons-fileupload 1.2.1

2. Librerías Javascript
   a. Google Maps Api v2
   b. Extensión LabeledMarker
   c. ExtJs 3.0.3
   d. Extensión Ext.ux.form.FileUploadField para ExtJs

3. Servidores web
   a. Apache – Tomcat v6.0 (pruebas locales)
   b. Google App Engine

4. Base de datos
   a. App Engine datastore con acceso JDO (Java Data Objects)

Procedimientos

El proyecto fue desarrollado utilizando al máximo los recursos gratuitos de la web. Haciendo diversos retoques para ajustarlos al objetivo del proyecto.

Servidor

Me decanté por utilizar Google App Engine (GAE), ya que permite colocar la web (y la base de datos asociada) en la infraestructura de Google[1].

Además me ahorré el desarrollo de un sistema de validación de usuarios. El GAE tiene clases Java para el control de usuarios utilizando cuentas de Google.

Para el acceso a base de datos utilicé la documentación suministrada por Google App Engine[2]

En el servidor desarrollé servlets para los diversos envíos de datos:

consultarlocalizacion

Este servlet está respondiendo por GET a peticiones del cliente. Permite un parámetro **id**, si está vacío devuelve por JSON todas las localizaciones guardadas en base de datos. En otro caso, devuelve la localización correspondiente al id indicado.

enviarlocalizacion

Sirve para guardar la localización de datos que llegan del cliente tanto por POST como por GET.

Devuelve por JSON el resultado de la inserción.

consultarimagen

Consulta la imagen en la base de datos según un parámetro **idimagen** pasado desde el cliente por GET.

enviarimagen

Los datos de imagen llegan por POST, la clase ServletFileUpload los procesa, se insertan y se devuelve el resultado al cliente utilizando JSON .

[1] Así, de forma simultánea, hago una prueba de "cloud computing" sobre la infraestructura de Google.
[2] http://code.google.com/appengine/docs/java/gettingstarted/

Cliente

En el momento que un usuario se valide en la pantalla inicial, aparecerá el mapa de Google Maps y se cargarán las localizaciones guardadas en la base de datos. Estas se colocan en el mapa utilizando un marcador extendido (LabeledMarker) que permite añadir al marcador estándar de Google (GMarker) propiedades adicionales. En este caso propiedades de localización.

Cuando el usuario hace clic en el mapa, se crea un marcador estándar de Google Maps (GMarker) y se abre una ventana de formulario[3] donde se cubrirán los datos relativos al lugar.

Una vez que se cubran correctamente los datos se convierte el marcador estándar a un marcador extendido (LabeledMarker)

JSON

En el cliente se utilizan mensajes JSON para la comunicación con el servidor, salvo para el envío de la imagen. Para ello utilizo una extensión de ExtJS para el envío de archivos mediante POST.

Archivos relevantes

- **js/formulario.js**: Desde donde se controla el envío de la imagen y de los datos de la localización.
- **js/map_functions.js:** Donde está el código de control especial sobre el mapa.

[3] Este formulario está creado utilizando la librería ExtJS.