

Google App Engine es una plataforma de desarrollo de aplicaciones Web que nace en Abril de 2008 y que en sus orígenes corría únicamente sobre Python.

En el siguiente enlace se puede ver un artículo que habla sobre las características de este servicio, cómo darse de alta y desarrollar una aplicación Hola Mundo en Python que corra en local y en el cloud Google.

http://observatorio.cnice.mec.es/index.php?option=com_content&view=article&id=735&catid=28%3Arecursos-online&Itemid=38&lang=es

Aproximadamente un año después de ver la luz, Google añade otro lenguaje de desarrollo, se trata de Java. Este artículo explicará como realizar la configuración necesaria para poder trabajar en local y en remoto con nuestras aplicaciones Java.

Además, este servicio puede desarrollarse con cualquier otro lenguaje que sea compatible con la Java Virtual Machine, como Ruby o JavaScript.

De momento PHP no está soportado, aunque se espera que sea el próximo paso en App Engine. Aún así podemos hacerlo correr a través de la implementación libre de PHP llamada Quercus. Existe más información disponible sobre este Open Source en <http://www.caucho.com/products/quercus/>

Se va a realizar este manual sobre una distribución Ubuntu, en concreto, sobre Max 5.0, de la Comunidad de Madrid con el kernel actualizado a 2.6.31-17-generic.

App Engine utiliza el standar Java Servlet para aplicaciones web. Los Servlet son objetos diseñados para correr en contenedores y son usuales en aplicaciones web dinámicas que hacen peticiones y demandan respuestas de los servidores a través de navegador.

App Engine ejecuta las clases de servlet de tu aplicación para procesar solicitudes y preparar respuestas en el entorno seguro que es la JVM (Java Virtual Machine).

El servlet, que no es otra cosa que un contenedor de clases java, se ejecuta en el servidor, a diferencia del conocido applet que lo hace en el navegador del usuario.

App Engine ejecuta el servlet con un objeto de solicitud y un objeto de respuesta y, a continuación, espera a que el servlet rellene el objeto de respuesta para enviarlo al usuario.

El ciclo de vida de un Servlet se divide en los siguientes puntos:

1.

El cliente solicita una petición a un servidor vía URL.

2.

El servidor recibe la petición.

1. Si es la primera, se utiliza el motor de Servlets para cargarlo y se llama al método `init()`.
2. Si ya está iniciado, cualquier petición se convierte en un nuevo hilo. Un Servlet puede manejar múltiples peticiones de clientes.
3. Se llama al método `service()` para procesar la petición devolviendo el resultado al cliente.
4. Cuando se apaga el motor de un Servlet se llama al método `destroy()`, que lo destruye y libera los recursos abiertos.

Por otro lado, cuando más adelante se esté ante el servlet picando código en modo consola en Eclipse se verá que aparece un fichero XML. App Engine necesita de este archivo de configuración adicional para poder desarrollar y ejecutar la aplicación. Este archivo se denomina `appengine-web.xml` y se ubica en `WEB-INF/` junto a `web.xml`. Incluye la ID registrada de la aplicación (Eclipse la crea a partir de una ID vacía que rellena el usuario), el número de versión de la aplicación y listas de archivos que se deben tratar como archivos estáticos (por

ejemplo, imágenes y CSS) y archivos de recursos (por ejemplo, JSP y otros datos de aplicación).

Empezando con Java.

Existen dos opciones para desarrollar aplicaciones Java en AppEngine:

- a) Descargar e instalar el SDK que nos ofrece Google, parecido al ya utilizado para Python en el artículo anterior de esta serie de AppEngine que ya hemos citado antes.
- b) Descargar e instalar un plugin del SDK que nos proporciona Google para Eclipse, el popular IDE (Integrated Development Environment) que utiliza gran parte de la comunidad de desarrolladores de Java.

El presente tutorial se centra en la segunda opción.

Para ello, deberemos comprobar que tenemos instalado el JDK (Java Development Kit) y JRE (Java Runtime Environment) ambos de Java.

Los que ya usen Eclipse para programar con otros lenguajes y no hayan especificado lo contrario en las opciones de instalación, ya tienen estos componentes.



Si no se tuvieran, se puede acudir a la web del proyecto libre OpenJDK <http://openjdk.java.net/> por lo que se escribe en un terminal

```
$ sudo apt-get install openjdk-6-jre openjdk-6-jdk
```

También se puede recurrir a otra opción propietaria como la que desarrolla Sun Microsystems, <http://java.sun.com/javase/downloads/index.jsp>

,

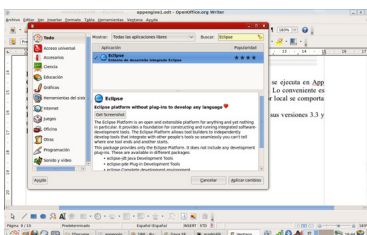
Donde se elegirá la plataforma, en este caso Linux.

Google App Engine es compatible con Java 5 y 6.

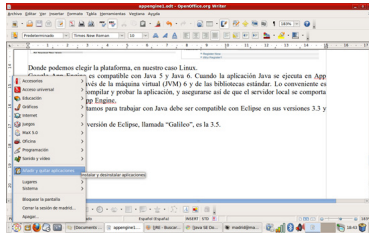
Cuando la aplicación Java se ejecuta en App Engine, lo hace a través de la máquina virtual (JVM) y de las bibliotecas estándar.

Lo conveniente es utilizar Java 6 para compilar y probar la aplicación, y asegurarse así de que el servidor local se comporta de forma similar a App Engine.

El plugin que necesitamos para trabajar con Java debe ser compatible con Eclipse en sus versiones 3.3 y superiores.



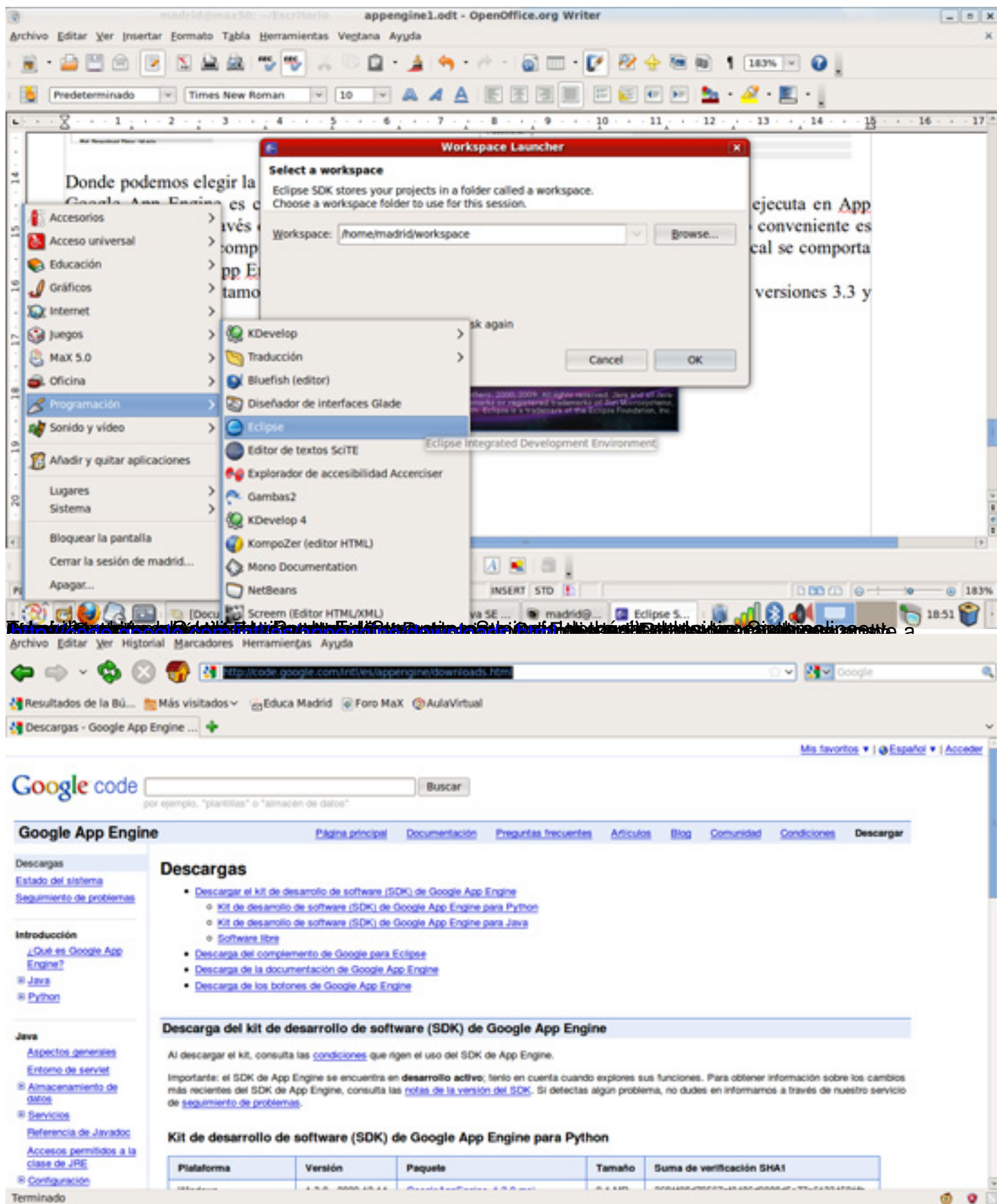
Instalamos la última versión de Eclipse, llamada “Galileo”, es la 3.5. Para ello vamos a Inicio, Añadir y quitar aplicaciones.



Y se introduce en la caja de búsqueda Eclipse, apareciendo como opción instalable. Aceptar e instalar.

AppEngine

Escrito por Carlos Artilles Fontales
Lunes, 14 de Junio de 2010 00:00

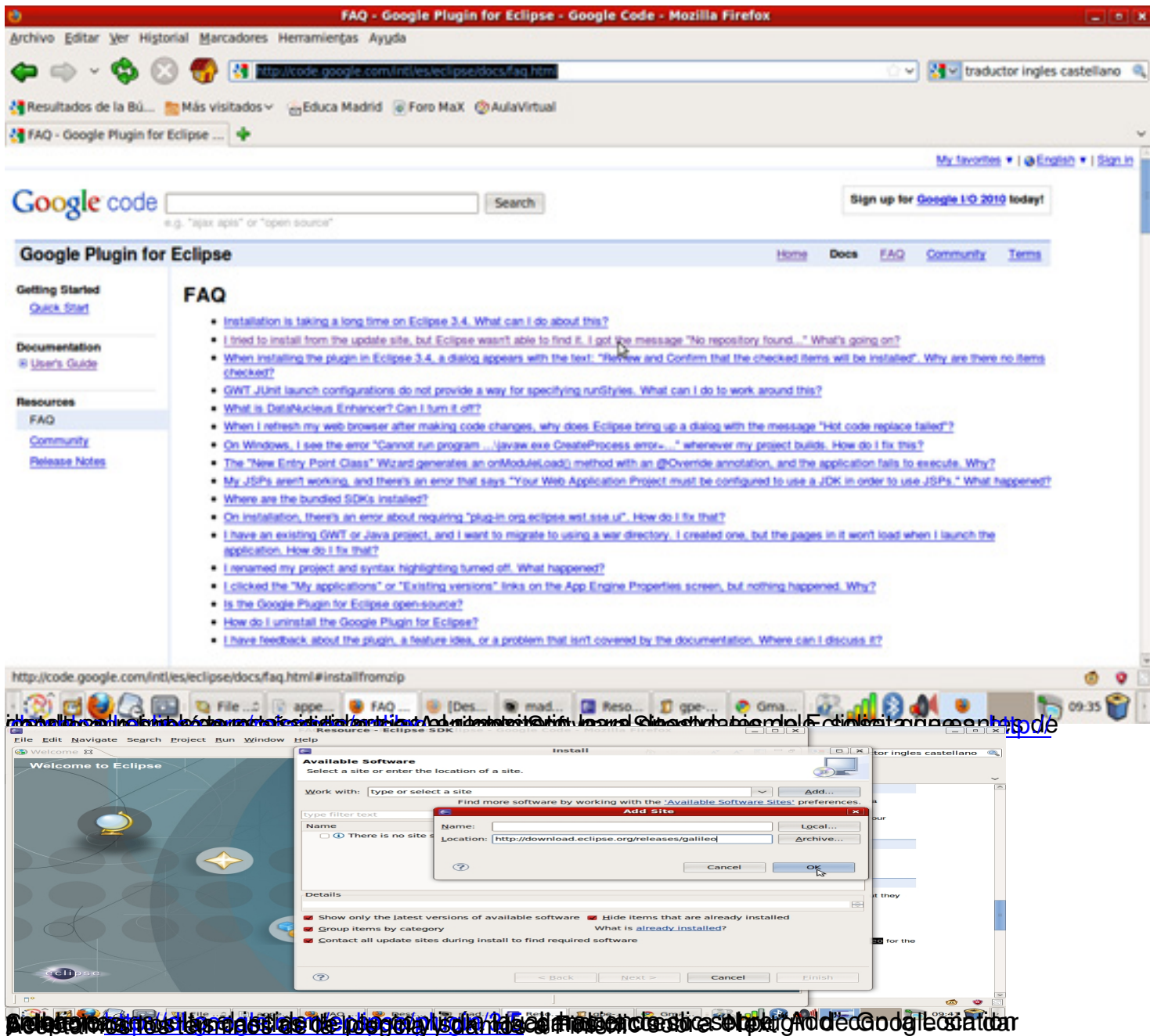




Terminado

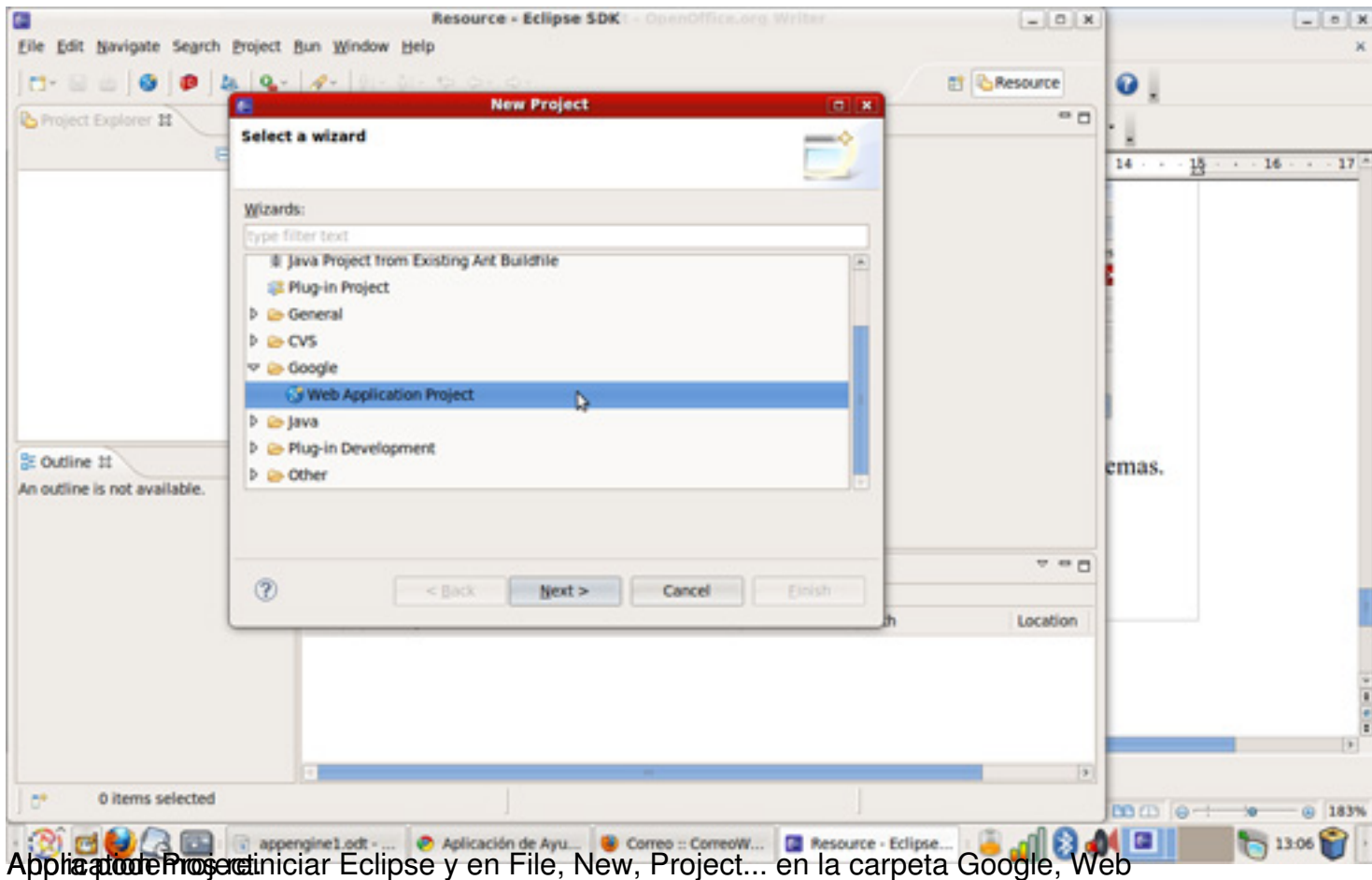
Se puede obtener información sobre el complemento de Google para Eclipse, de donde se puede descargar el SDK de Google App Engine, en <http://code.google.com/intl/es/appengine/downloads.html>.

Se puede obtener información sobre el complemento de Google para Eclipse, de donde se puede descargar el SDK de Google App Engine, en <http://code.google.com/intl/es/appengine/downloads.html>.



AppEngine

Escrito por Carlos Artilles Fontales
Lunes, 14 de Junio de 2010 00:00



Application Project. Iniciar Eclipse y en File, New, Project... en la carpeta Google, Web

Create a Web Application Project



Create a Web Application project in the workspace or in an external location

Project name:

ajalospajarosjava

Package: (e.g. com.example.myproject)

ajalospajaros

Location

☒ Create new project in workspace

☐ Create new project in:

Directory: /home/madrid/workspace/ajalospajarosjava

[Browse...](#)

Google SDKs

☐ Use Google Web Toolkit

☒ Use default SDK (GWT - 2.0.0)

[Configure SDKs...](#)

☐ Use specific SDK: GWT - 2.0.0

☒ Use Google App Engine

☒ Use default SDK (App Engine - 1.3.0)

[Configure SDKs...](#)

☐ Use specific SDK: App Engine - 1.3.0



< Back

Next >

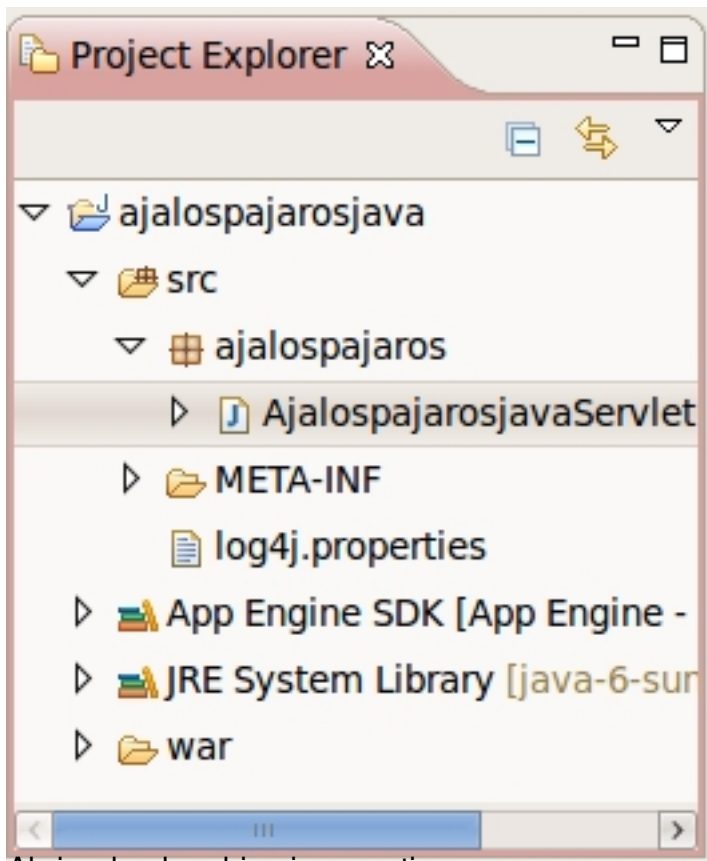
Cancel

Finish

Se han creado los proyectos en el IDE de Eclipse para Google SDKs Use Google Web Toolkit puesto

AppEngine

Escrito por Carlos Artiles Fontales
Lunes, 14 de Junio de 2010 00:00



Abriendo el archivo java se tiene:



Project Run

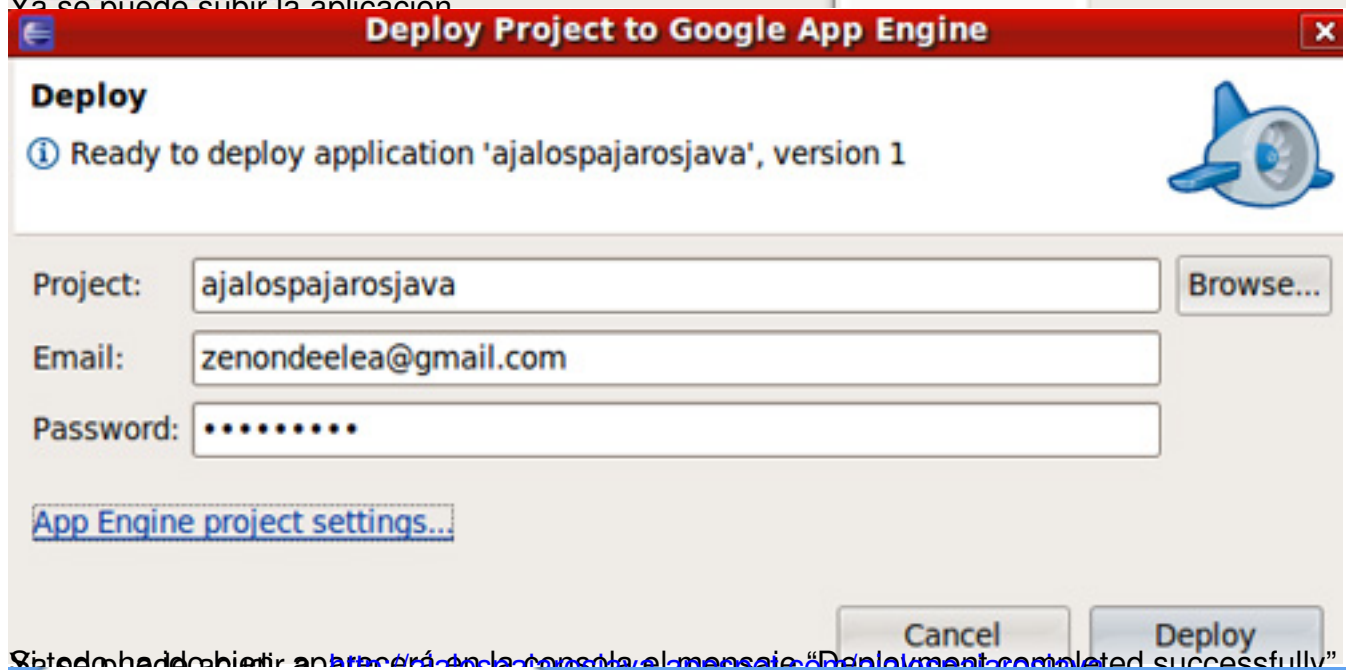
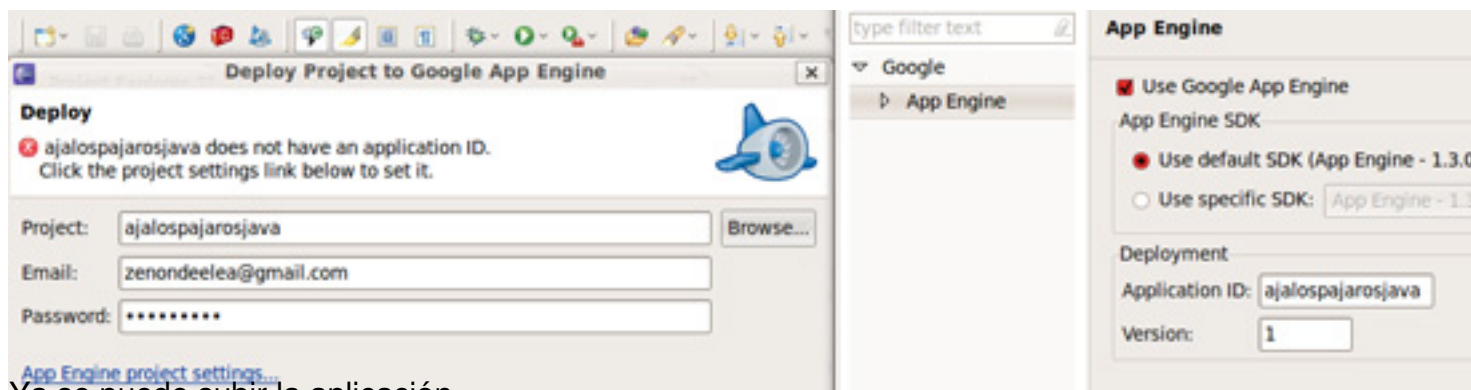


spajarosjavaSer

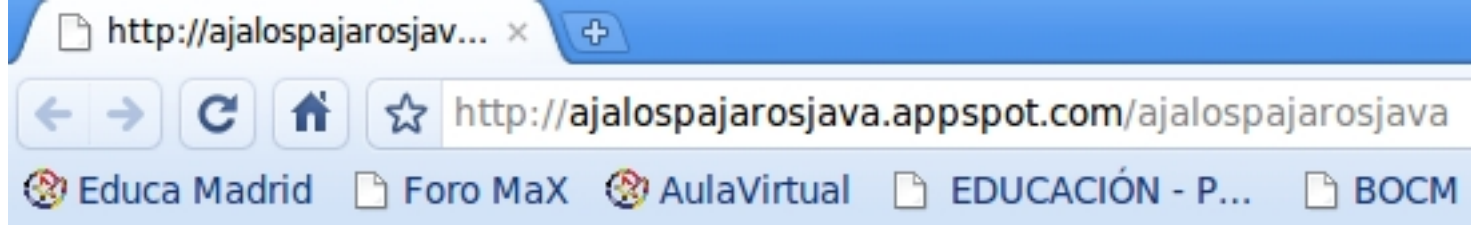
Open the file 'AjalospajarosjavaServlet.java' in the IDE. The code is as follows: package ajalospajaros; import java.io.IOException; @SuppressWarnings("serial") public class AjalospajarosjavaServlet extends HttpServlet { public void doGet(HttpServletRequest req, HttpServletResponse resp) throws IOException { resp.setContentType("text/plain"); resp.getWriter().println("Hello, world"); } }

AppEngine

Escrito por Carlos Artilles Fontales
Lunes, 14 de Junio de 2010 00:00



Si todo ha ido bien, en la consola de la interfaz de usuario se mostrará el mensaje "Deployment completed successfully".



Hola amigos de Java y AppEngine

Este artículo es parte de la serie de artículos sobre Google App Engine, que puedes encontrar en el siguiente enlace: <http://www.carlosartiles.com/2010/06/14/google-app-engine-1-3-0/>