Departamento de Lenguajes y Sistemas Informáticos

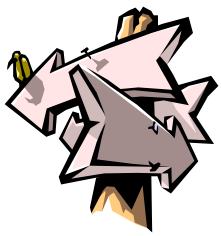
Desarrollo de Aplicaciones con GWT (I)

Práctica 2

Arquitectura e Integración del Software

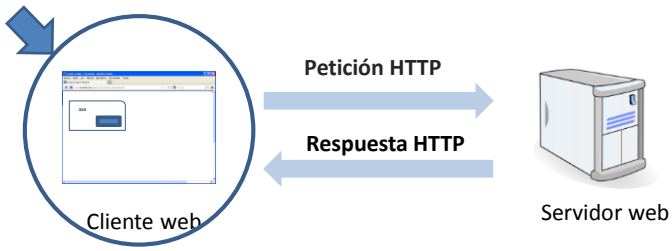
Curso 2012/2013

Índice

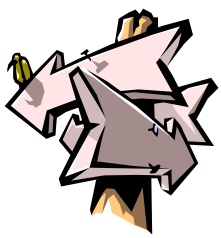


Introducción

- En las aplicaciones web, también las desarrolladas con GWT, tenemos que tener en cuenta la arquitectura cliente-servidor.
- GWT nos permite desarrollar ambas capas utilizando lenguaje Java.
- **En la primera parte de esta práctica, aprenderemos a desarrollar la interfaz de usuario con GWT.**



Índice



Módulo GWT

- Una aplicación GWT se llama **módulo**.
- Un módulo tiene un fichero de configuración *nombredemodulo.gwt.xml*
- En el fichero de configuración se indican cosas como la plantilla de los controles GWT por defecto o la clase de entrada a nuestra aplicación.

Módulo GWT

AISSPracticaAgenda.gwt.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  When updating your version of GWT, you should also update this DTD reference,
  so that your app can take advantage of the latest GWT module capabilities.
-->
<!DOCTYPE module PUBLIC "-//Google Inc.//DTD Google Web Toolkit 2.5.0//EN"
  "http://google-web-toolkit.googlecode.com/svn/tags/2.5.0/distro-source/core/src/gwt-module.dtd">
<module rename-to='aisspracticaagenda'>
  <!-- Inherit the core Web Toolkit stuff.                        -->
  <inherits name='com.google.gwt.user.User'/>

  <!-- Inherit the default GWT style sheet.  You can change       -->
  <!-- the theme of your GWT application by uncommenting          -->
  <!-- any one of the following lines.                            -->
  <inherits name='com.google.gwt.user.theme.clean.Clean'/>
  <!-- <inherits name='com.google.gwt.user.theme.standard.Standard'/> -->
  <!-- <inherits name='com.google.gwt.user.theme.chrome.Chrome'/> -->
  <!-- <inherits name='com.google.gwt.user.theme.dark.Dark'/>     -->

  <!-- Other module inherits                                      -->

  <!-- Specify the app entry point class.                         -->
  <entry-point class='es.us.eii.client.AISSPracticaAgenda'/>

  <!-- Specify the paths for translatable code                    -->
  <source path='client'/>
  <source path='shared'/>

</module>
```

Estilo visual de la aplicación por defecto.

Dependencias con otros módulos.

Clase de entrada (entrypoint) de nuestra aplicación.

Módulo GWT

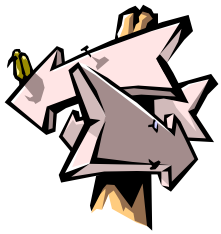
- Las aplicaciones GWT tienen tres paquetes principales:
  - **Client:** En este paquete desarrollamos todo el código con el que GWT genera Javascript y HTML que se procesa en el navegador.
  - **Server:** En este paquete desarrollamos el código del lado de servidor (backend).
  - **Shared:** Para implementar clases cuyos objetos se usarán tanto en cliente como en servidor (hay algunas restricciones para esto).

Módulo GWT

- Asociado a nuestro módulo tenemos un fichero HTML *`Nombredemodulo.html`*
- También llamado fichero host, es el html en el que se ejecuta todo el código que GWT genera desde nuestro código JAVA en el paquete cliente de GWT.
- Todo el código HTML que se incluye aquí aparecerá en toda nuestra aplicación.
- Aquí podremos asociar nuestras hojas de estilos (css) o añadir código HTML estático (ej: cabecera o pie de página).

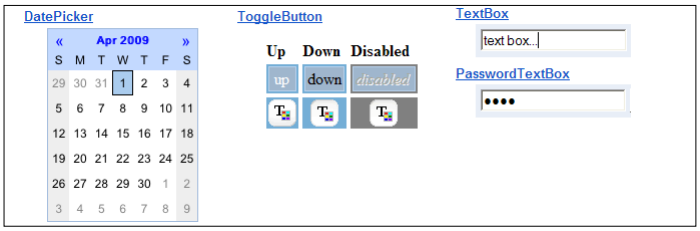
Índice

- Introducción
- Módulo GWT
- **Diseño de Interfaz de Usuario**
- Gestión de Eventos
- Enlaces



Diseño de Interfaz de Usuario

- Las clases del paquete *client* de nuestra aplicación corresponde a la ejecución en cliente (navegador).
- El diseño de la interfaz de usuario web en GWT está basado en widgets.
- Entre los widgets tenemos contenedores (paneles), y controles (botones, etiquetas, etc…).



Diseño de Interfaz de Usuario

- Vistazo general a los widgets de GWT:

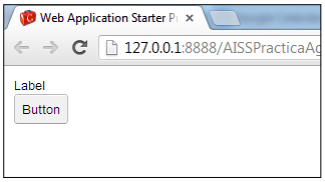
  https://developers.google.com/web-toolkit/doc/1.6/RefWidgetGallery

  http://gwt.googleusercontent.com/samples/Showcase/Showcase.html

Diseño de Interfaz de Usuario

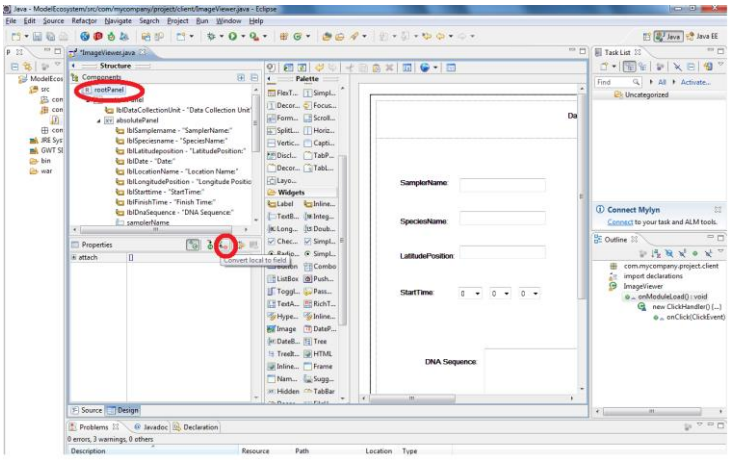
- Todos los widgets (contenedores y controles) se van definiendo de forma jerárquica a partir de un panel llamado **RootPanel**.
- Para obtener este panel usamos el método estático ***get()*** de la clase RootPanel, que se corresponde con el elemento ***body*** del HTML principal.
- El punto de entrada a la aplicación se hace en el método **onModuleLoad().**

```
public void onModuleLoad() {
  Panel p = RootPanel.get();
  VerticalPanel a = new VerticalPanel();
  a.add(new Label("Label"));
  a.add(new Button("Button"));
  p.add(a);
}
```



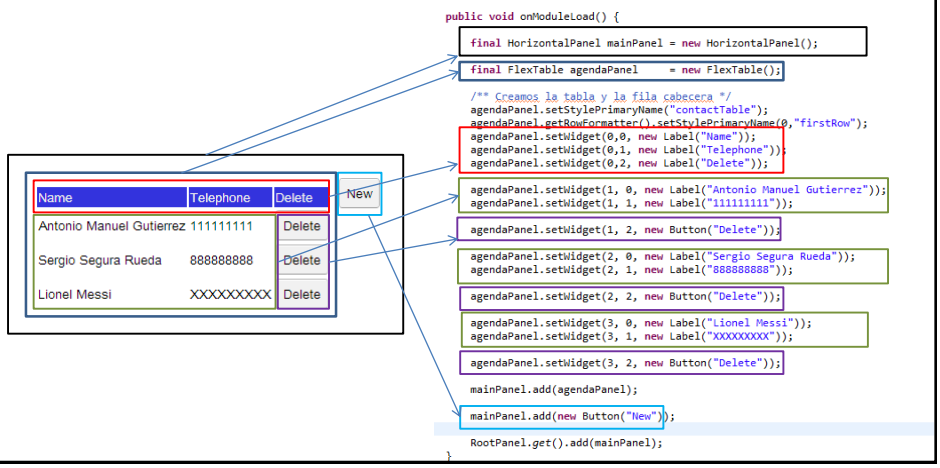
Diseño de Interfaz de Usuario

GWT Designer



Diseño de Interfaz de Usuario

- Antes del desarrollo de la interfaz es conveniente analizar la información que se va a mostrar al usuario y que éste va a proporcionar, para hacer un prototipo de interfaz de usuario (**mockup**) para decidir los widgets que necesitamos.



Diseño de Interfaz de Usuario
Estilo

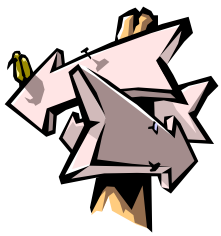
- GWT provee varias plantillas predefinidas para el estilo de los widgets. En el fichero de configuración de GWT se indica la plantilla a utilizar.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  When updating your version of GWT, you should also update this DTD reference,
  so that your app can take advantage of the latest GWT module capabilities.
-->
<!DOCTYPE module PUBLIC "-//Google Inc.//DTD Google Web Toolkit 2.5.0//EN"
  "http://google-web-toolkit.googlecode.com/svn/tags/2.5.0/distro-source/core/src/gwt-module.dtd">
<module rename-to='aisspracticaagenda'>
  <!-- Inherit the core Web Toolkit stuff.                        -->
  <inherits name='com.google.gwt.user.User'/>

  <!-- Inherit the default GWT style sheet.  You can change       -->
  <!-- the theme of your GWT application by uncommenting          -->
  <!-- any one of the following lines.                            -->
  <inherits name='com.google.gwt.user.theme.clean.Clean'/>
  <!-- <inherits name='com.google.gwt.user.theme.standard.Standard'/> -->
  <!-- <inherits name='com.google.gwt.user.theme.chrome.Chrome'/> -->
  <!-- <inherits name='com.google.gwt.user.theme.dark.Dark'/>      -->
```

- En general, los estilos se definen en una hoja de estilos CSS aparte.

Índice

- Introducción
- Módulo GWT
- Diseño de Interfaz de Usuario
- **Gestión de Eventos y Navegación**
- Enlaces



Gestión de Eventos y Navegación
Eventos

- GWT provee de un sistema de eventos para responder a las acciones del usuario y el control de la información en la interfaz de usuario.
- Para manejar o reaccionar a un evento, se indica el objeto que va a manejar dicho evento. Estos objetos tienen que implementar la interfaz de manejador de dicho evento (Evento Click=>Interfaz ClickHandler)
- Existen eventos predefinidos asociados a los widgets de GWT (y interfaces predefinidas para dichos eventos), como pulsar botón, cambiar el valor de un campo de texto, etc.
- También es posible definir nuestros propios eventos.

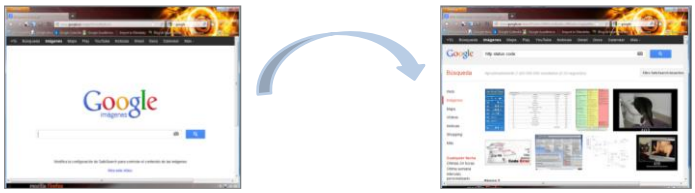
Gestión de Eventos y Navegación
Eventos

- Un ejemplo: Indicar el comportamiento asociado al pulsar un botón.

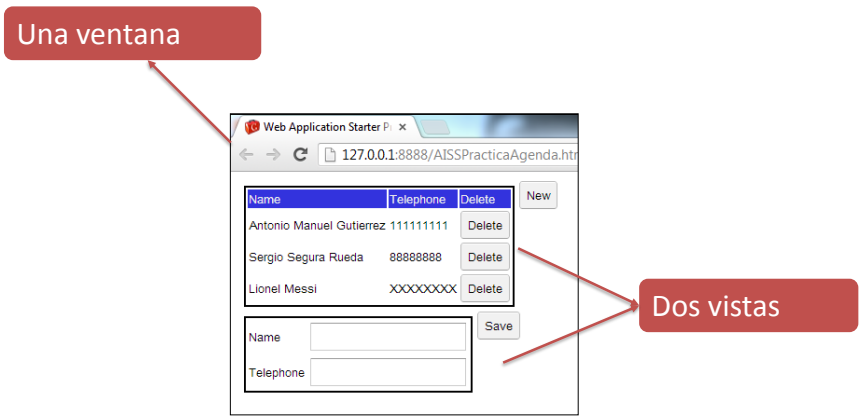
```
Button pressButton = new Button("Press");
final TextBox nameBox = new TextBox();
pressButton.addClickHandler(new ClickHandler() {
      public void onClick(ClickEvent event) {
          nameBox.setValue("Has Pulsado el Boton");
      }
    });
```

Gestión de Eventos y Navegación
Vistas y navegación

- En GWT siempre se trabaja con la **misma ventana**.
- Cualquier aplicación trabaja con gran cantidad de **vistas** de diferente funcionalidad
- Es conveniente separar e independizar las vistas en código.
- También aplicamos este criterio cuando en la misma ventana queremos mostrar diálogos diferentes.
- Necesitamos gestionar las transiciones entre vistas de una manera flexible.



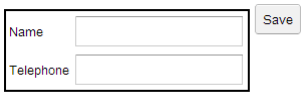
Gestión de Eventos y Navegación
Vistas y navegación



Gestión de Eventos y Navegación
Vistas y navegación

- Para la implementación de varias vistas seguiremos los siguientes pasos:
  - Para cada vista, crearemos una clase que extienda a la clase *Composite*.
    public class EditSongView extends Composite {…}
  - El constructor de la clase debe recibir un mapa en el que la vista podrá recibir parámetros de otras vistas.
  - En el constructor de la clase crearemos un panel (ej. HorizontalPanel) que será el contenedor principal de la vista.
    HorizontalPanel mainPanel = new HorizontalPanel();
  - Una vez creado el panel principal debemos invocar al método intiWidget con dicho panel.
    initWidget(mainPanel);

Gestión de Eventos y Navegación
Vistas y navegación




```
public class ViewCreate extends Composite{
  public ViewCreate(Map<String, String> params) {
      HorizontalPanel mainPanel = new HorizontalPanel();
      initWidget(mainPanel);
      final FlexTable agendaPanel = new FlexTable();
      agendaPanel.setWidget(0,0, new Label("Name"));
      …
      …
      mainPanel.add(agendaPanel);
      Button saveButton = new Button("Save");
      mainPanel.add(saveButton);

      saveButton.addClickHandler(new ClickHandler() {
          public void onClick(ClickEvent event) {
             FlowController.go("list",
                new HashMap<String,String>());
          }
      });
      }
}
```

```
public class ViewList extends Composite {
  public ViewList(Map<String, String> params) {
      HorizontalPanel mainPanel = new
HorizontalPanel();
      initWidget(mainPanel);
      FlexTable agendaPanel = new FlexTable();
      agendaPanel.setWidget(0,0, new Label("Name"));
      …
      …
      Button addButton = new Button("New");
      addButton.addClickHandler(new ClickHandler() {
         public void onClick(ClickEvent event) {
            FlowController.go("create",
               new HashMap<String,String>());
         }
      });
      mainPanel.add(addButton);
  }
}
```

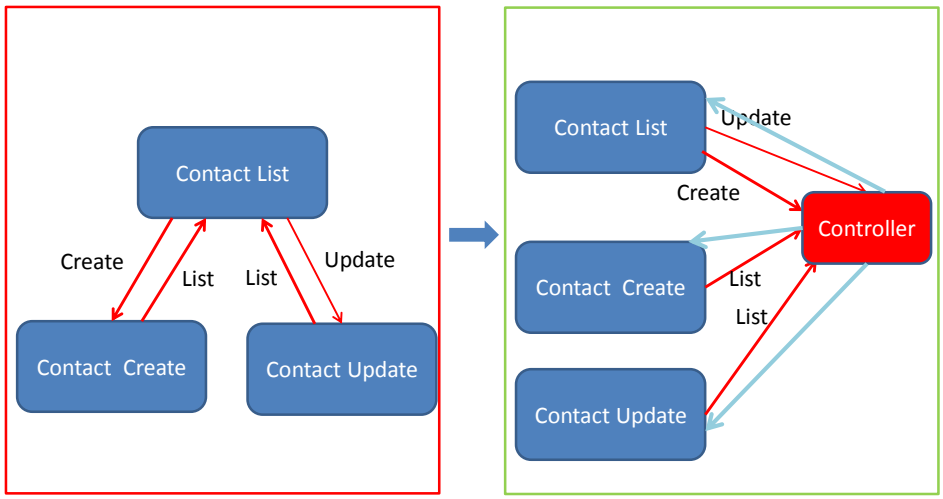
Gestión de Eventos y Navegación
Vistas y navegación

- Para la gestión de las navegación seguiremos los siguientes pasos:
  - Crear una clase adicional (clase controlador) con un método estático para gestionar la navegación.
  - El método debe recibir como parámetros una cadena (token) que identifique la vista a activar y un mapa con los parámetros que se deseen pasar a la vista.
  - En el controlador decidimos la jerarquía de vistas utilizando un panel padre para las vistas. Como las vistas extienden Composite, se pueden utilizar como paneles.

```
public class FlowController {

   public static void go(String token, Map<String,String> params){
     Panel p = RootPanel.get();
     if (token=="list" || token=="init" ){
        p.clear();
        p.add(new ViewList(params));
     }else if (token=="create" ){
        p.add(new ViewCreate(params)); //Se añade al contenido que existiese
     }
}
```

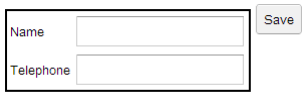
Gestión de Eventos y Navegación
Vistas y navegación

- Las transiciones entre ventanas las delegamos en un solo punto, que tiene la responsabilidad de decidir que vistas activar.



Gestión de Eventos y Navegación
Vistas y navegación

- Las vistas usarán la clase controlador para decir la vista que quieren activar y los parámetros que quieren pasarle.




```
public class ViewCreate extends Composite{
  public ViewCreate(Map<String, String> params) {
      HorizontalPanel mainPanel = new HorizontalPanel();
      initWidget(mainPanel);
      final FlexTable agendaPanel = new FlexTable();
      agendaPanel.setWidget(0,0, new Label("Name"));
      …
      …
      mainPanel.add(agendaPanel);
      Button saveButton = new Button("Save");
      mainPanel.add(saveButton);

      saveButton.addClickHandler(new ClickHandler() {
          public void onClick(ClickEvent event) {
             FlowController.go("list",
                new HashMap<String,String>());
          }
      });
      }
}
```

```
public class ViewList extends Composite {
  public ViewList(Map<String, String> params) {
     HorizontalPanel mainPanel = new
HorizontalPanel();
      initWidget(mainPanel);
      FlexTable agendaPanel = new FlexTable();
      agendaPanel.setWidget(0,0, new Label("Name"));
      …
      …
      Button addButton = new Button("New");
      addButton.addClickHandler(new ClickHandler() {
         public void onClick(ClickEvent event) {
            FlowController.go("create",
               new HashMap<String,String>());
         }
      });
      mainPanel.add(addButton);
  }
}
```

Gestión de Eventos y Navegación
Vistas y navegación

- El esquema de vistas empleado permite mostrar varias vistas simultánes en la misma ventana o sólo una vista a la vez.

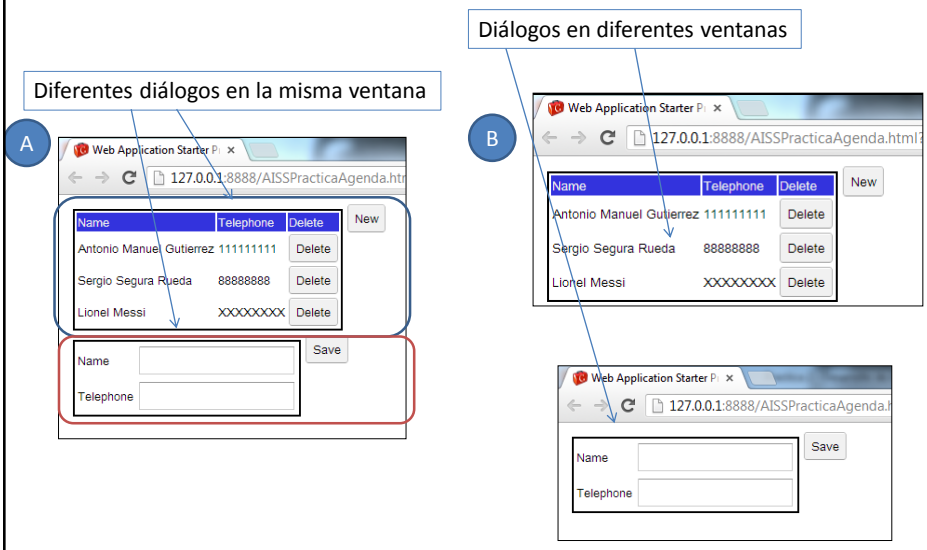
A

```
public class FlowController { //Vistas simultáneas

   public static void go(String token, Map<String,String> params){
     Panel p = RootPanel.get();
     if (token=="list" || token=="init" ){
        p.clear();
        p.add(new ViewList(params));
     }else if (token=="create" ){
        p.add(new ViewCreate(params)); //Se añade al contenido que existiese
     }
}
```

B

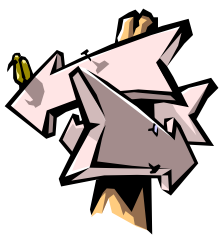
```
public class FlowController {//Solo una vista a la vez

   public static void go(String token, Map<String,String> params){
     Panel p = RootPanel.get();
     if (token=="list" || token=="init" ){
        p.clear();
        p.add(new ViewList(params));
     }else if (token=="create" ){
        p.clear(); //Limpiamos la ventana
        p.add(new ViewCreate(params));
     }
}
```

Gestión de Eventos y Navegación
Vistas y navegación



Índice

- Introducción
- Módulo GWT
- Diseño de Interfaz de Usuario
- Gestión de Eventos
- **Enlaces**



Enlaces

- **Tutoriales (MUY RECOMENDADOS)**

Tutorial de iniciación a GWT. Stockwatcher
https://developers.google.com/web-toolkit/doc/latest/tutorial/gettingstarted

Tutorial de iniciación a GWT. Stockwatcher con GWT Designer
https://developers.google.com/webtoolkit/tools/gwtdesigner/tutorials/stockwatcher

Tutorial de uso del patrón MVP en GWT (¡Para nota!)
https://developers.google.com/web-toolkit/articles/mvp-architecture?hl=es

Enlaces

- Widgets de GWT

https://developers.google.com/web-toolkit/doc/1.6/RefWidgetGallery
http://gwt.googleusercontent.com/samples/Showcase/Showcase.html

- Edición de Mockups (necesario para el primer entregable)

http://www.balsamiq.com
http://pencil.evolus.vn/

Disclaimer and Terms of Use

All material displayed on this presentation is for teaching and personal use only.

Many of the images that have been used in the presentation are Royalty Free images taken from http://www.everystockphoto.com/. Other images have been sourced directly from the Public domain, from where in most cases it is unclear whether copyright has been explicitly claimed. Our intention is not to infringe any artist's copyright, whether written or visual. We do not claim ownership of any image that has been freely obtained from the public domain. In the event that we have freely obtained an image or quotation that has been placed in the public domain and in doing so have inadvertently used a copyrighted image without the copyright holder's express permission we ask that the copyright holder writes to us directly, upon which we will contact the copyright holder to request full written permission to use the quote or images.