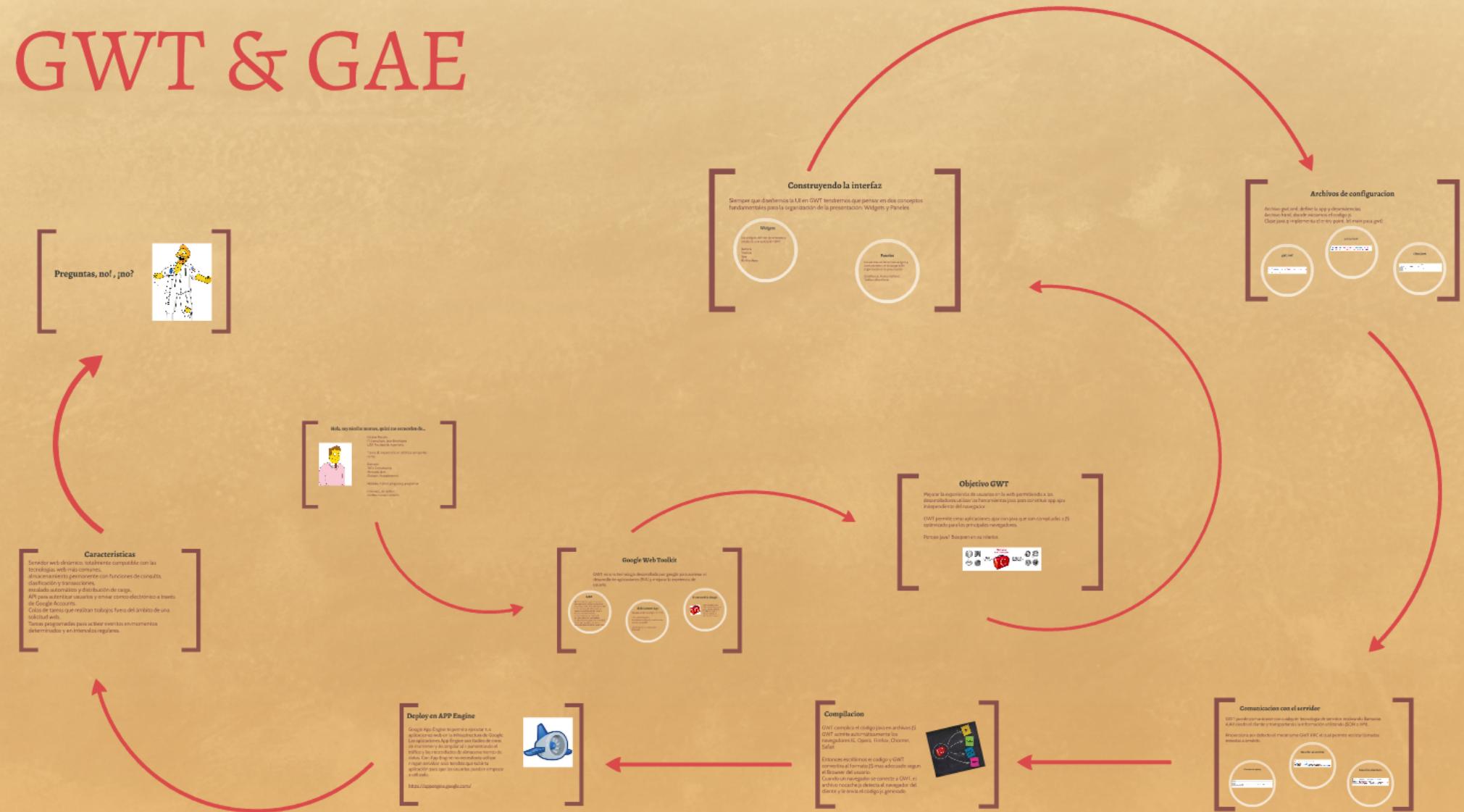
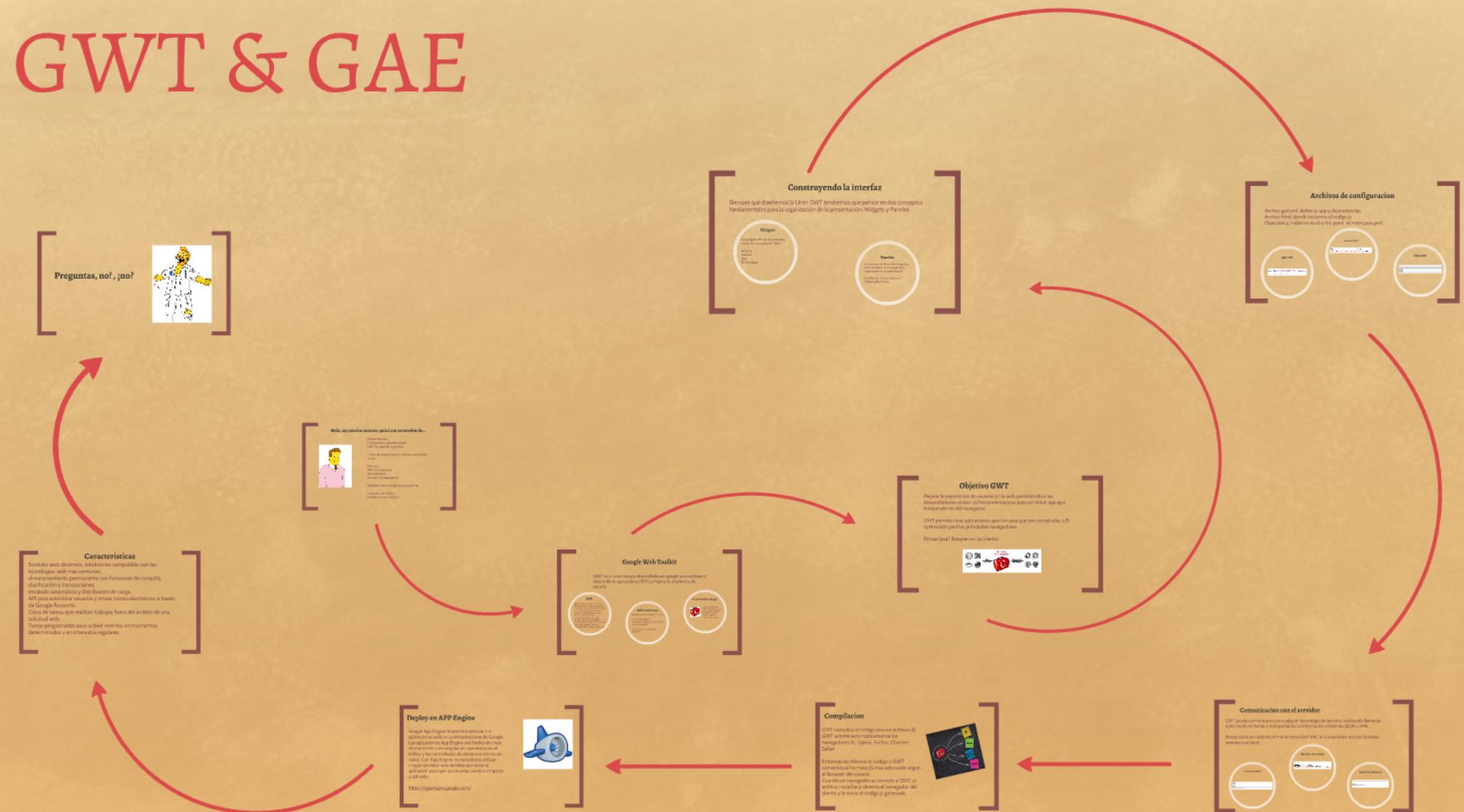


# GWT & GAE



# GWT & GAE



# Hola, soy nicolas moraes, quizá me recuerden de...



Nicolas Moraes.  
IT Consultant, Java Developer.  
UDE Facultad de Ingenieria.

7 años de experiencia en distintas compañías  
como:

Raccoon.  
TATA Consultancy.  
MercadoLibre.  
Globant. (Actualmente)

Hobbies: Futbol, ping pong, programar

nmoraes\_ en twitter  
nicolas moraes LinkedIn

# Google Web Toolkit

GWT: es una tecnologia desarrollada por google para acelerar el desarrollo de aplicaciones (RIA) y mejorar la experiencia de usuario.

## AJAX

AJAX a vuelo de pájaro: (asincronus java script and xml) es una técnica el cual nos permite crear aplicaciones RIA (rich internet application), estas se ejecutan en el browser del usuario mientras se mantiene una comunicación asíncrona con el servidor, esto sirve para realizar cambios sobre la pagina sin necesidad de recargar la pagina. (aumenta velocidad, interactividad y usabilidad).

## Rich Internet App

Ejemplos de RIA( silverlight, Fx, AJAX).

Desarrollar RIA, tiene desventajas( codigo JS, cross browser, conocer javascript).

Una Solucion, frameworks JS (JQUERY)

## Framework de Google



Aqui es donde google realiza un framework para acelerar el desarrollo de app RIAs basadas en AJAX, mejorando la experiencia del usuario.

# AJAX

AJAX a vuelo de pájaro: (asincronus java script and xml) es una técnica el cual nos permite crear aplicaciones RIA (rich internet application), estas se ejecutan en el browser del usuario mientras se mantiene una comunicación asíncrona con el servidor, esto sirve para realizar cambios sobre la pagina sin necesidad de recargar la pagina. (aumenta velocidad, interactividad y usabilidad).

# **Rich Internet App**

Ejemplos de RIA( silverlight, Fx, AJAX).

Desarrollar RIA, tiene  
desventajas( código JS, cross browser,  
conocer javascript).

Una Solucion, frameworks JS  
(JQUERY)

# Framework de Google



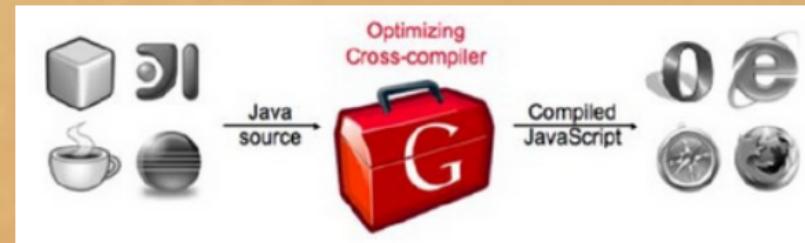
Aquí es donde google realiza un framework para acelerar el desarrollo de app RIAS basadas en AJAX, mejorando la experiencia del usuario.

# Objetivo GWT

Mejorar la experiencia de usuarios en la web permitiendo a los desarrolladores utilizar las herramientas java para constituir app ajax independiente del navegador.

GWT permite crear aplicaciones ajax con java que son compiladas a JS optimizado para los principales navegadores.

Porque java? Busquen en su interior.



# Construyendo la interfaz

Siempre que diseñemos la UI en GWT tendremos que pensar en dos conceptos fundamentales para la organización de la presentación: Widgets y Paneles

## Widgets

Los widgets definen las entradas y salidas de una aplicación GWT

Buttons  
Textbox  
Tree  
RichTextArea

## Paneles

Los paneles contienen los widgets y otros paneles y se encargan de la organización en la presentación

DockPanels, HorizontalPanel,  
TabPanel, RootPanel.

# Widgets

Los widgets definen las entradas y salidas de una aplicación GWT

Buttons

Textbox

Tree

RichTextArea

# Paneles

Los paneles contienen los widgets y otros paneles y se encargan de la organización en la presentación

DockPanels, HorizontalPanel,  
TabPanel, RootPanel.

# Archivos de configuracion

Archivo gwt.xml, define la app y dependencias.

Archivo html, donde iniciamos el codigo js.

Clase java q implementa el entry point. (el main para gwt)

gwt.xml

```
<!-- Specify the app entry point class.
<entry-point class='com.myPackage.client.DevFest2013' />
<!-- Specify the paths for translatable code -->
```

archivo html

```
<script type="text/javascript" language="javascript"
src="devFest2013/devFest2013.nocache.js"></script>
```

clase java

```
/* Entry point classes during development (for tests).
 */
public class DevFest2013 implements EntryPoint {
    ...
    // The message displayed to the user when the server cannot be reached or
    // returns an error.
    ...
}
```

# gwt.xml

```
<!-- Specify the app entry point class.  
<entry-point class='com.myPackage.client.DevFest2013'>  
  
<!-- Specify the paths for translatable code  
-->
```

## archivo html

```
<script type="text/javascript" language="javascript"  
src="devfest2013/devfest2013.nocache.js"></script>
```

# clase java

```
/**  
 * Entry point classes define <code>onModuleLoad()</code>.   
 */  
public class DevFest2013 implements EntryPoint {  
    /**  
     * The message displayed to the user when the server cannot be reached or  
     * returns an error.  
     */  
}
```

# Comunicacion con el servidor

GWT puede comunicarse con cualquier tecnología de servidor, realizando llamadas AJAX desde el cliente y transportando la información utilizando JSON o XML.

Proporciona por defecto el mecanismo GWT RPC el cual permite realizar llamadas remotas a servlets.

Creamos un proxy

```
/**  
 * Create a remote service proxy to talk to the server-side Greeting  
 * service.  
 */  
private final GreetingServiceAsync greetingService = GWT  
    .create(GreetingService.class);
```

Interfaz de servicio

```
/**  
 * The client side stub for the RPC service.  
 */  
@RemoteServiceRelativePath("greet")  
public interface GreetingService extends RemoteService {  
    String greetServer(String name) throws IllegalArgumentException;  
}
```

Interfaz asincrona

```
/**  
 * The async counterpart of <code>GreetingService</code>.   
 */  
public interface GreetingServiceAsync {  
    void greetServer(String input, AsyncCallback<String> callback)  
        throws IllegalArgumentException;  
}
```

# Creamos un proxy

```
/**  
 * Create a remote service proxy to talk to the server-side Greeting  
service.  
 */  
private final GreetingServiceAsync greetingService = GWT  
    .create(GreetingService.class);
```

# Interfaz de servicio

```
/**  
 * The client side stub for the RPC service.  
 */  
@RemoteServiceRelativePath("greet")  
public interface GreetingService extends RemoteService {  
    String greetServer(String name) throws IllegalArgumentException;  
}
```

# Interfaz asincrona

```
/**  
 * The async counterpart of <code>GreetingService</code>.   
 */  
public interface GreetingServiceAsync {  
    void greetServer(String input, AsyncCallback<String> callback)  
        throws IllegalArgumentException;  
}
```

# Compilacion

GWT complica el código java en archivos JS

GWT admite automáticamente los navegadores IE, Opera, Firefox, Chrome, Safari.

Entonces escribimos el código y GWT convertira al formato JS mas adecuado segun el Browser del usuario.

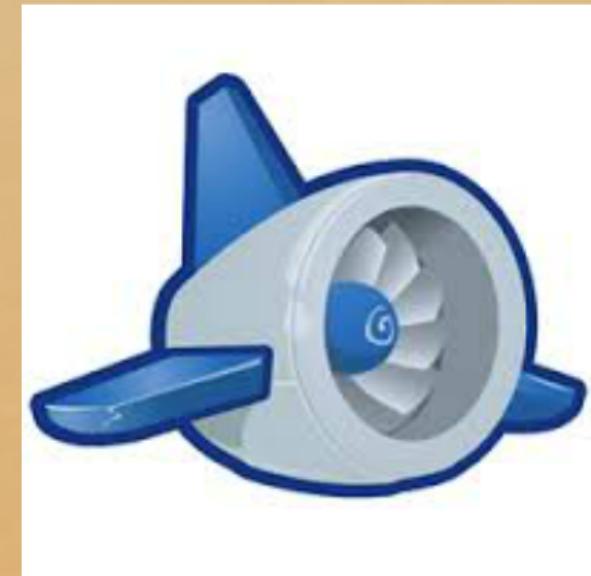
Cuando un navegador se conecte a GWT, el archivo nocache.js detecta el navegador del cliente y le envía el código js generado.



# Deploy en APP Engine

Google App Engine te permite ejecutar tus aplicaciones web en la infraestructura de Google. Las aplicaciones App Engine son fáciles de crear, de mantener y de ampliar al ir aumentando el tráfico y las necesidades de almacenamiento de datos. Con App Engine no necesitarás utilizar ningún servidor: solo tendrás que subir tu aplicación para que los usuarios puedan empezar a utilizarla.

<https://appengine.google.com/>



# Características

Servidor web dinámico, totalmente compatible con las tecnologías web más comunes, almacenamiento permanente con funciones de consulta, clasificación y transacciones, escalado automático y distribución de carga, API para autenticar usuarios y enviar correo electrónico a través de Google Accounts. Colas de tareas que realizan trabajos fuera del ámbito de una solicitud web, Tareas programadas para activar eventos en momentos determinados y en intervalos regulares.

# Preguntas, no! , ¡no?



# GWT & GAE

