

# Formation GWT



# Sommaire Formation GWT

- **Introduction**
- **Qu'est ce qu'une application GWT ?**
- Méthodologie et étapes clés gestion projet GWT
- Interface utilisateur : widgets, panels, évènements
- GWT et les appels distants RPC
- Design patterns GWT
- Intégration applis coté serveur (Spring, Servlets...)
- Intégration GWT/Javascript : JSNI
- Modularisation d'applications GWT
- Génération de code

# Introduction

➤ Historique

➤ Architecture

➤ Services

➤ Exemples

# Historique

- 2005 : Ajax / web 2.0
- Frameworks Prototype, Script.aculo.us, DWR
  - Javascript pour des interfaces + riches
  - Rendre Ajax + facile !
- Oui mais...Comment gérer un projet javascript ?
  - Compatibilité navigateurs ?
  - Environnement de Dév (Débug, arrêt...) ?
  - Automatisation du build & travail en équipe ?
  - Tests unitaires ?
  - Internationalisation ?
  - Communication avec serveur ?
- Quelles compétences Javascript ?

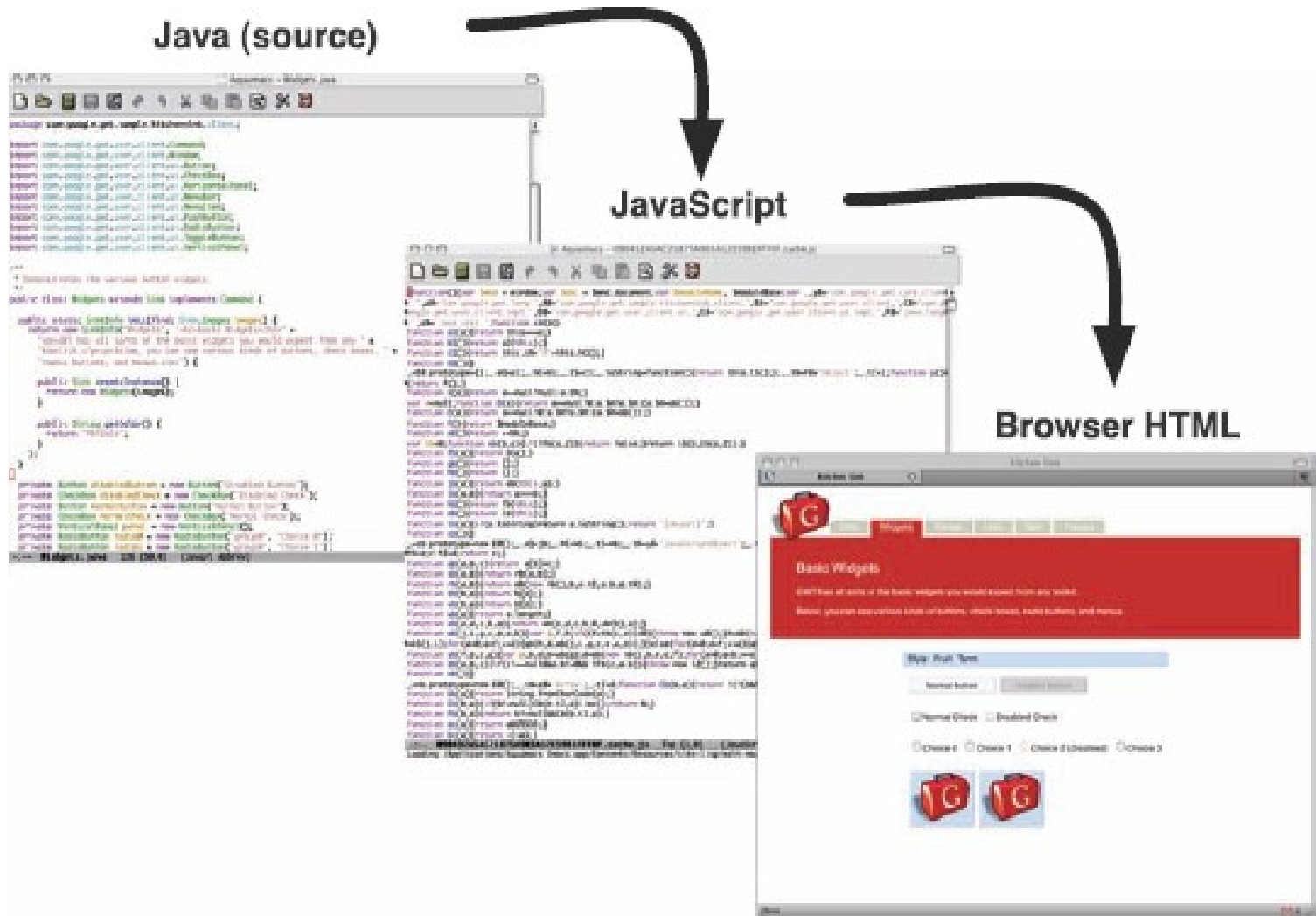
# Historique

- 2006 (JavaOne) : lancement Google Web Toolkit
- Utiliser Java pour générer Javascript !
- Bruce Johnson et Joel Webber



- Efficacité prouvée – communauté grandissante.

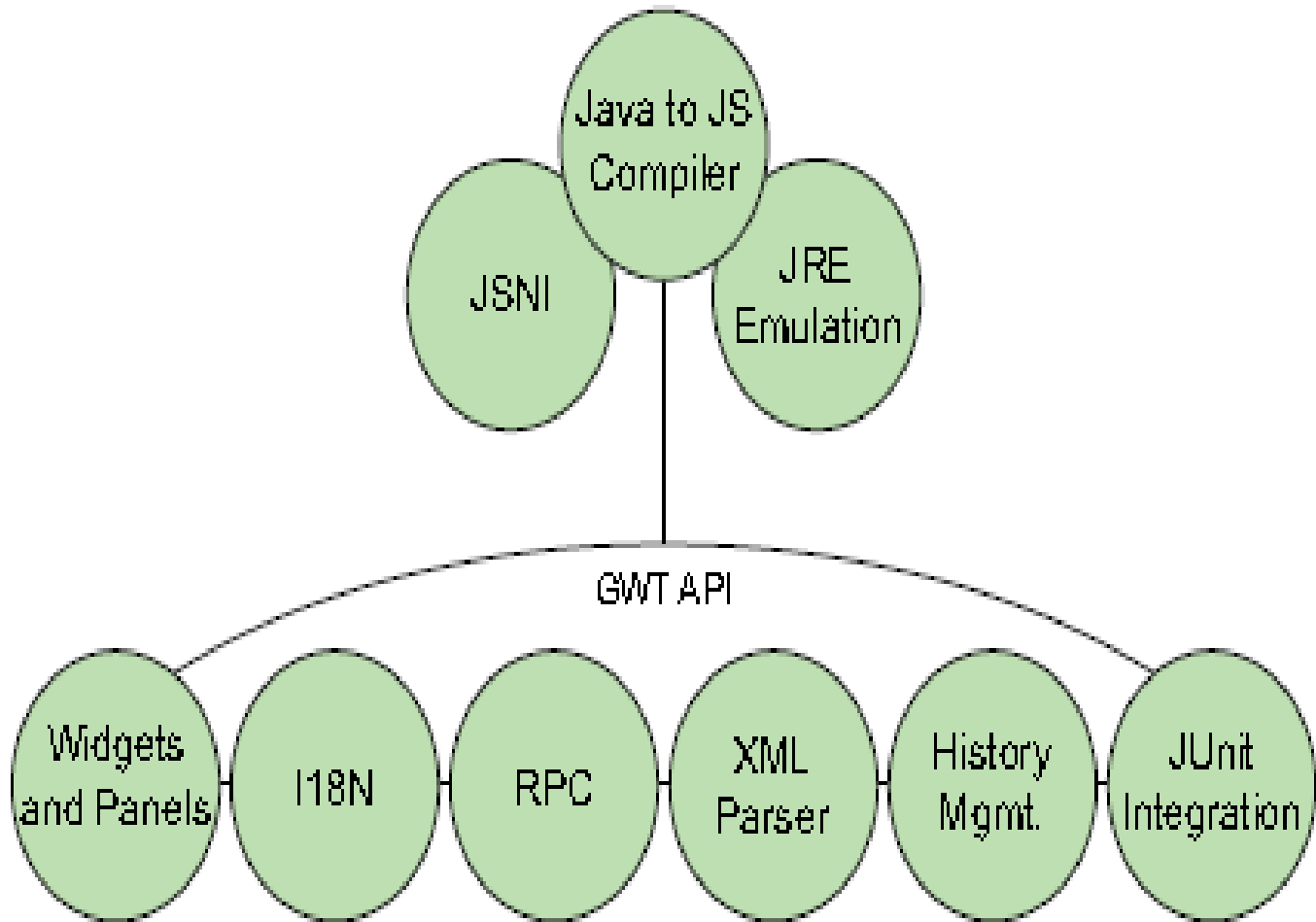
# GWT en action



## Qu'est ce que GWT ?

- GWT est un framework Ajax permettant d'écrire du code Java qui sera compilé en Javascript et exécuté coté client dans un navigateur.
- Vous pouvez utiliser vos outils Java habituels pour développer, débbugguer er tester vos applications GWT.
- GWT gère pour vous les différences de Javascript et HTML DOM entre navigateurs
- Le compilateur Java vers Javascript optimise au maximum le Javascript généré.
- Vous pouvez intégrer Gwt avec tout script Javascript et en particulier d'autres frameworks Ajax (Dojo, jQuery,...)

# Architecture GWT



# Outils fournis par GWT 1.5



gwt-windows-1.5.3.zip  
21 329 Ko

- ApplicationCreator
- BenchmarkViewer
- I18nCreter
- JunitCreator
- ProjectCreator

doc	
samples	
about.html	3 Ko
about.txt	1 Ko
applicationCreator.cmd	1 Ko
benchmarkViewer.cmd	1 Ko
COPYING	13 Ko
COPYING.html	16 Ko
gwt-benchmark-viewer.jar	3 475 Ko
gwt-dev-windows.jar	9 835 Ko
gwt-ll.dll	13 Ko
gwt-module.dtd	6 Ko
gwt-servlet.jar	703 Ko
gwt-user.jar	2 943 Ko
i18nCreator.cmd	1 Ko
index.html	6 Ko
junitCreator.cmd	1 Ko
projectCreator.cmd	1 Ko
release_notes.html	61 Ko
swt-win32-3235.dll	352 Ko

# Outils fournis par GWT 1.6



gwt-windows-1.6.4.zip  
22 949 Ko

- 
- BenchmarkViewer
- I18nCreator
- JunitCreator
- 
- WebAppCreator

doc	
samples	
about.html	4 Ko
about.txt	2 Ko
benchmarkViewer.cmd	1 Ko
COPYING	13 Ko
COPYING.html	16 Ko
gwt-api-checker.jar	61 Ko
gwt-benchmark-viewer.war	2 342 Ko
gwt-dev-windows.jar	10 959 Ko
gwt-ll.dll	13 Ko
gwt-module.dtd	6 Ko
gwt-servlet.jar	915 Ko
gwt-user.jar	3 489 Ko
i18nCreator.cmd	1 Ko
index.html	6 Ko
junitCreator.cmd	1 Ko
release_notes.html	62 Ko
swt-win32-3235.dll	352 Ko
webAppCreator.cmd	1 Ko

## Services GWT 1.6

- Génération du squelette de votre application
  - Outil : WebappCreator
- Gestion de l'internationalisation/fichiers propriétés
  - Outil : i18nCreator
- Tests Unitaires
  - Outil : JunitCreator
- Ecrire du code compatible navigateurs
  - IE, Firefox, Opera, Safari
- Faire communiquer code client / Server
  - Wrapper XMLHttpRequest, GWT-RPC

# EchoSystème GWT

- Le compilateur GWT est puissant mais...pas assez de Widgets + intégration serveur ?
- Projets complémentaires : Ext GWT et GWT-Ext
  - Objectifs : fournir Widgets qui manquent à GWT
  - <http://code.google.com/p/gwt-ext/>
  - <http://extjs.com/products/gxt/>
- Intégration GWT / Spring : GWT-Server Libraries
  - Rendre des beans Spring des services RPC
  - <http://gwt-widget.sourceforge.net/>
- Data binding : Gwittir (<http://code.google.com/p/gwittir/>)

# Quelles applis GWT en prod ?

- Lombardi Blueprint (Modélisation processus métier)
  - [http://www.youtube.com/watch?v=-f-YC\\_qaQ-8](http://www.youtube.com/watch?v=-f-YC_qaQ-8)
  
- GoGrid (Cloud computing / Infra. multiserveur)
  - <http://www.gogrid.com/how-it-works/gwt.php>
  
- Scenechronize (Animation de scène)
  - <http://www.youtube.com/watch?v=2gqDsi8zRt4>
  
- ...

## Nouveautés GWT 1.6

- Nouvelle structure de projet ('expanded war')
  - Code généré par le compilateur
  - Code supplémentaire appli web par le développeur
- Compilation java vers Javascript + rapide
- Nouveaux Widgets (DatePicker, LazyPanel)
  - Voir exemple : Showcase
- Nouvelle gestion événementielle (optimisée)
  - EventHandler au lieu de EventListener
  - Ex : ClickHandler (méthode onClick(ClickEvent))
- 120 bugs fix / améliorations.
- Plugin Eclipse !



# Plugin Eclipse Google



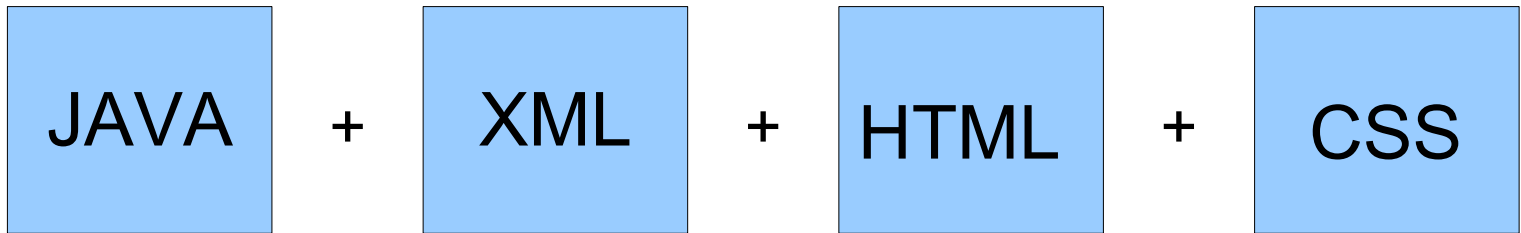
- La méthode la + rapide pour développer une appli GWT
- Le plugin installe GWT Toolkit + Google App Engine
- Google App Engine (serveur jee de google)
  - Déploiement facile d'applications jee (war)
  - Vérif code compatible App Engine
- Fonctionnalités GWT
  - Reconnaissance code JSNI (syntaxe, recherche...)
  - Interface config du compilateur GWT
  - Assistant pour création EntryPoint, modules, HTML
  - Support tests unitaires

# Qu'est ce qu'une application GWT (1.6 +)?

- JAVA (Entrypoint)
- XML
- HTML
- CSS
- WAR (mode compilé)

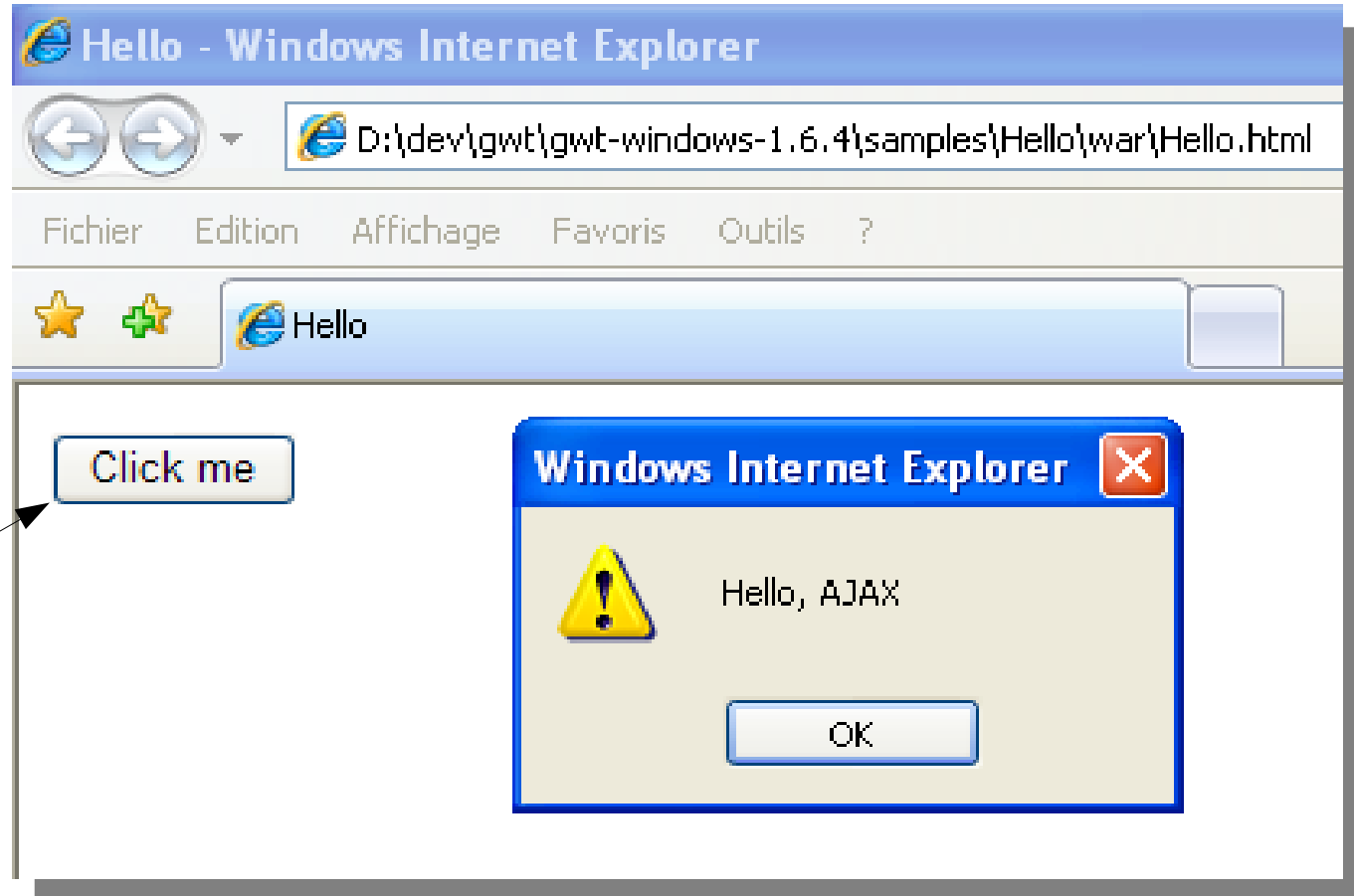
# Qu'est ce qu'une application GWT ?

- 1 application GWT = 1 module GWT (ou plusieurs) =



- JAVA = classe d'entrée obligatoire (interface **EntryPoint**)  
+ autres classes facultatives (Listeners, métier,...)
- La classe d'entrée redéfinie méthode **OnLoadModule()**
  - Objectifs : composition de la vue (widgets, panels, layouts...) **ET** gestion d'évènements utilisateurs (click...)

# Exemple d'exécution appli GWT Hello (livrée avec GWT)



Bouton créé  
dynamiquement

# Point d'entrée application GWT : Classe java (EntryPoint)

```
package com.google.gwt.sample.hello.client;

import com.google.gwt.core.client.EntryPoint;
import com.google.gwt.event.dom.client.ClickEvent;
import com.google.gwt.event.dom.client.ClickHandler;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.Button;
import com.google.gwt.user.client.ui.RootPanel;
```

```
/**
```

```
 * HelloWorld application.
```

```
 */
```

```
public class Hello implements EntryPoint {
```

2

```
public void onModuleLoad() {
```

```
    Button b = new Button("Click me", new ClickHandler() {
```

```
        public void onClick(ClickEvent event) {
```

```
            Window.alert("Hello, AJAX");
```

```
        }
```

```
    };
```

```
    RootPanel.get().add(b);
```

```
}
```

```
}
```

Interface à implémenter

1

Création bouton

3

Redéfinition

de OnModuleLoad

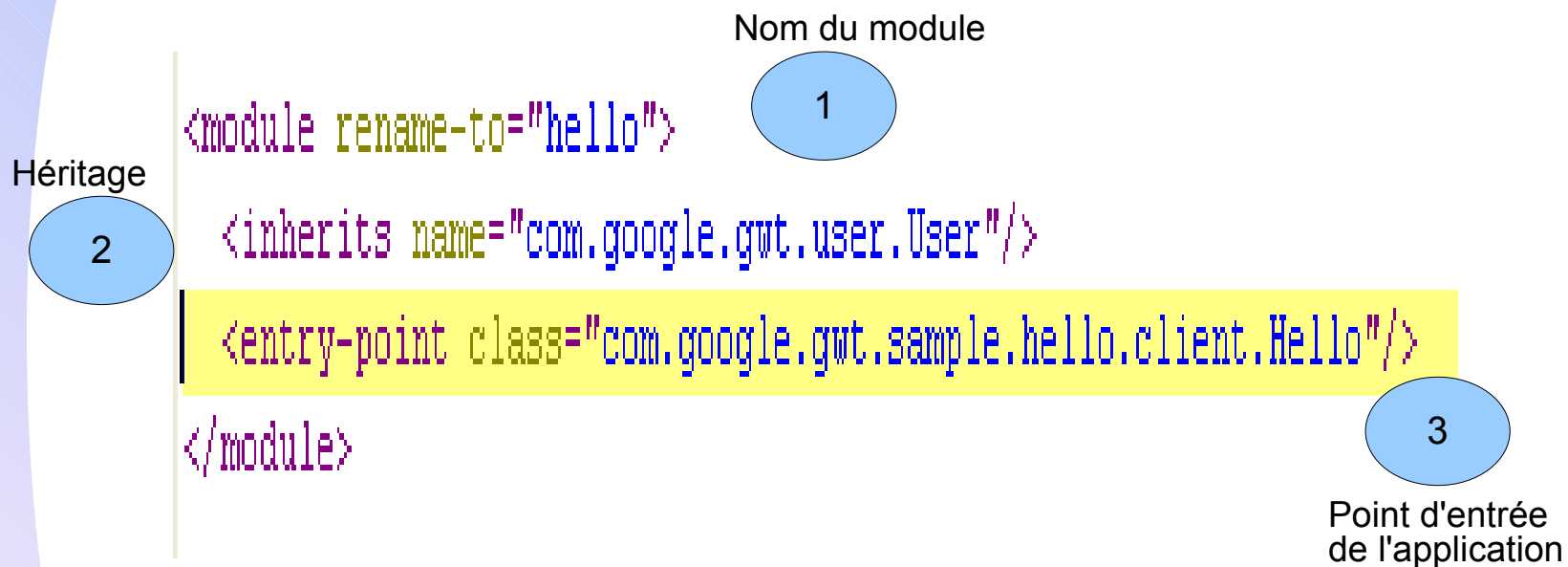
4

Gestion événement click souris

5

Ajout bouton dans fenêtre navigateur

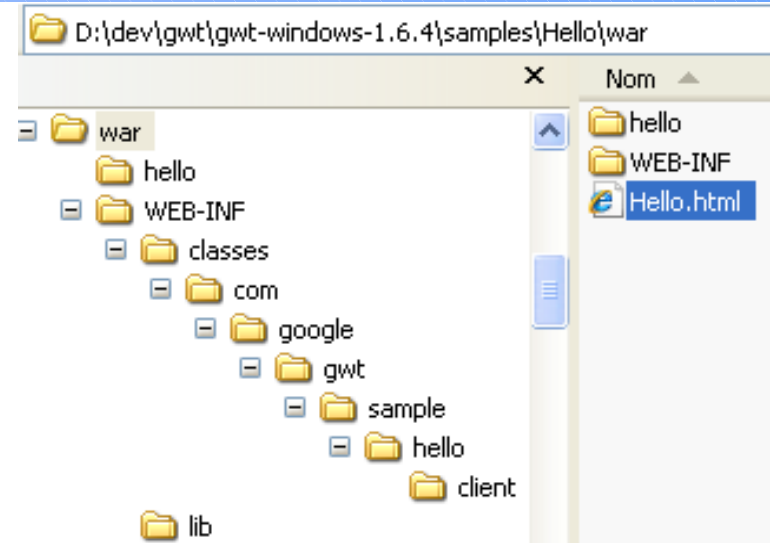
# Module XML (Hello.gwt.xml)



# Page HTML (Hello.html)

Page d'accueil de l'application WAR.  
Ici ne contient aucun élément  
Peux contenir des ancres <div id="">

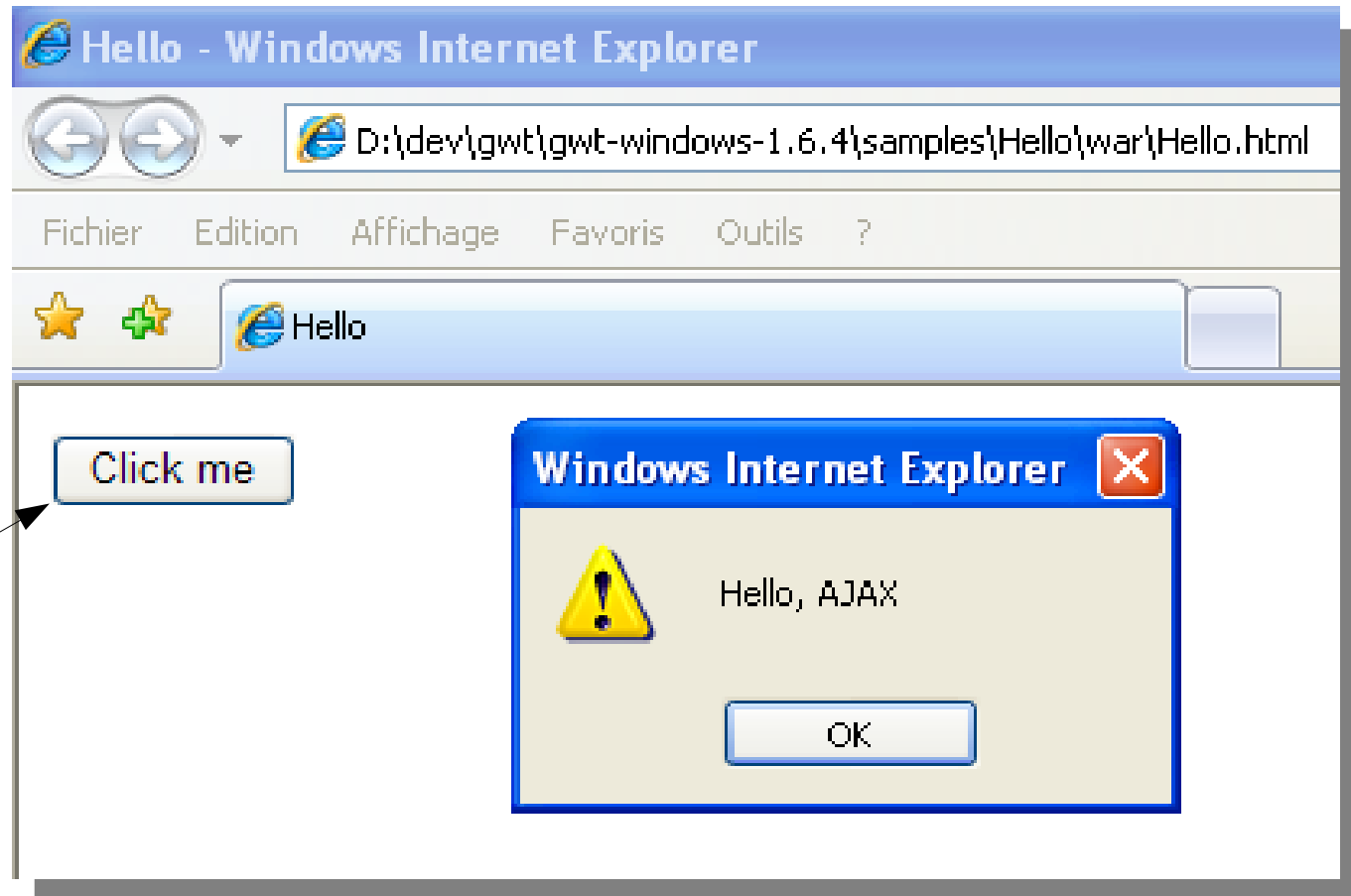
```
<html>  
  <head>  
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">  
    <title>Hello</title>  
  </head>  
  <body bgcolor="white">  
    <script type="text/javascript" language="javascript" src="hello/hello.nocache.js"></script>  
  </body>  
</html>
```



1

Permet d'identifier le type de navigateur utilisé  
et en déduire la page html à envoyer à l'utilisateur

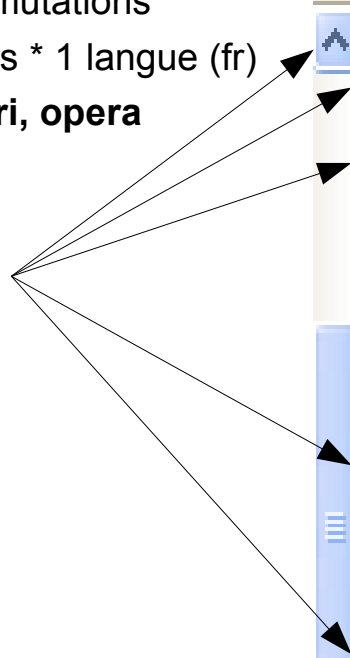
# Exécution



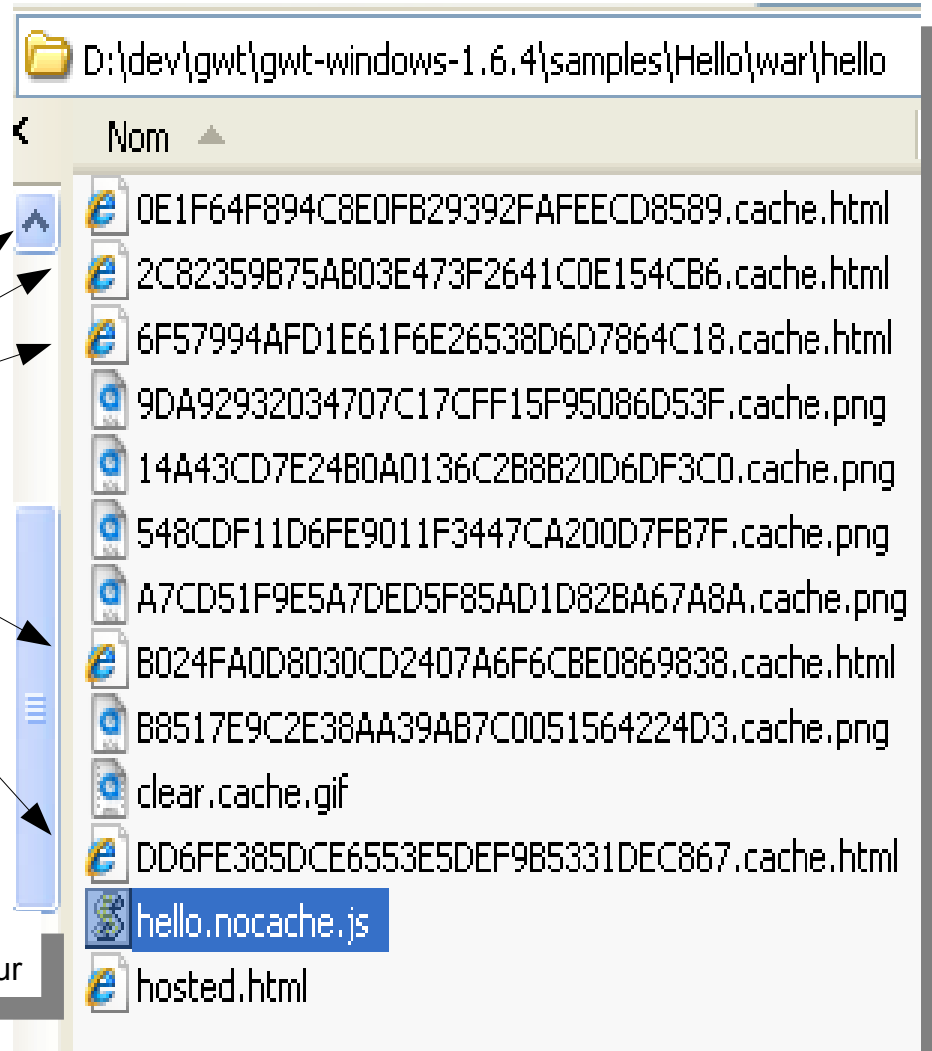
Bouton créé dynamiquement

# Fichiers générés par GWT & notion de permutation

5 fichiers html générés = 5 permutations  
 résultats de 5 navigateurs cibles \* 1 langue (fr)  
**ie6, gecko, gecko1\_8, safari, opera**



Permet d'identifier le type de navigateur utilisé et en déduire la page html à envoyer à l'utilisateur



## Contact : Douglas Mbiandou

- Ingénieur INSA Lyon (2000)
- 10 ans d'expériences projets SI
- Architecte / Formateur Java
- Responsable formation Objis
- 04 78 29 37 26 / 06 60 46 76 45
- [douglas.mbiandou@objis.com](mailto:douglas.mbiandou@objis.com)

