Corso di “Sviluppo di applicazioni Web”

Giovanni Grasso

GWT (Introduzione)

Come partire?

- Installare il toolkit link..
- Impostare il plugin di Eclipse
  - Select Window->Preferences->Cypal Studio and set GWT home to the directory where you have installed the Google Web Toolkit.

Struttura di progetto

- <nome modulo> (ad es. it/saw/prova) - Il package radice del progetto, contiene un file XML di descrizione del modulo
- <nome modulo>/client – File sorgente client-side
- <nome modulo>/server – Codice server-side code
- <nome modulo>/public – Risorse statiche che possono essere richieste all’applicazione Web (HTML, immagini, ecc.)

Applicazione GWT

- È necessario un punto di accesso (*main* in altri ambienti), definito “module entry point”

```
public interface EntryPoint
{
    void onModuleLoad();
}
```

- Cos’è un modulo?
- Definito da un file di configurazione XML
- Specifica un punto di accesso
  - Può trattarsi di una classe che genera l’HTML iniziale
- Indica un mapping delle servlet, da usare in Hosted Mode
- Può ereditare da altri moduli

Modalità operative

- GWT supporta due modalità operative:
  - **Hosted mode:**
    - utilizza un'istanza di Tomcat built-in come ambiente di esecuzione
    - L'applicazione gira come bytecode Java all'interno della JVM
    - È il più usato durante lo sviluppo, poiché, l'ambiente di esecuzione determina la disponibilità di facilities per lo sviluppo, senza abbandonare l'IDE (es. Eclipse)
  - **Web mode :**
    - L'applicazione è, in questo caso, composta solo da HTML e da codice Javascript compilato dal nostro codice Java originario, tramite il compilatore Java-to-JavaScript di GWT
    - Al momento del deploy su un ambiente di produzione, solo l'accoppiata JavaScript/HTML dovrà essere sottoposta ai web server
    - Gli utenti finali accedono solo alla versione "Web mode" dell'applicazione

Alcuni dei package più significativi:

- user.client.rpc – Implementazione client side delle classi per RPC (IsSerializable, AsyncCallback)
- user.client.ui –Widgets, Panels e altri classi di supporto alla GUI

Alcuni dei package più significativi:

- core.client:
  - GWT (uncaught exception handler)
  - JavascriptException
  - Interfaccia EntryPoint
- user.client – Browser history, manipolazione DOM, event handling ecc

Package user.client.ui

- Contiene alcuni elementi di base delle UI: TextBox, PasswordTextBox, Grid, Label, Listbox, MenuBar, MenuItem, Tree, HTMLTable
- Tutti gli elementi discendono da Widget
- Le astrazioni di Panel : Panel, VerticalPanel, HorizontalPanel, DeckPanel, DockPanel, RootPanel
- I panels sono oggetti compositi e supportano gerarchie guidate dal part-of

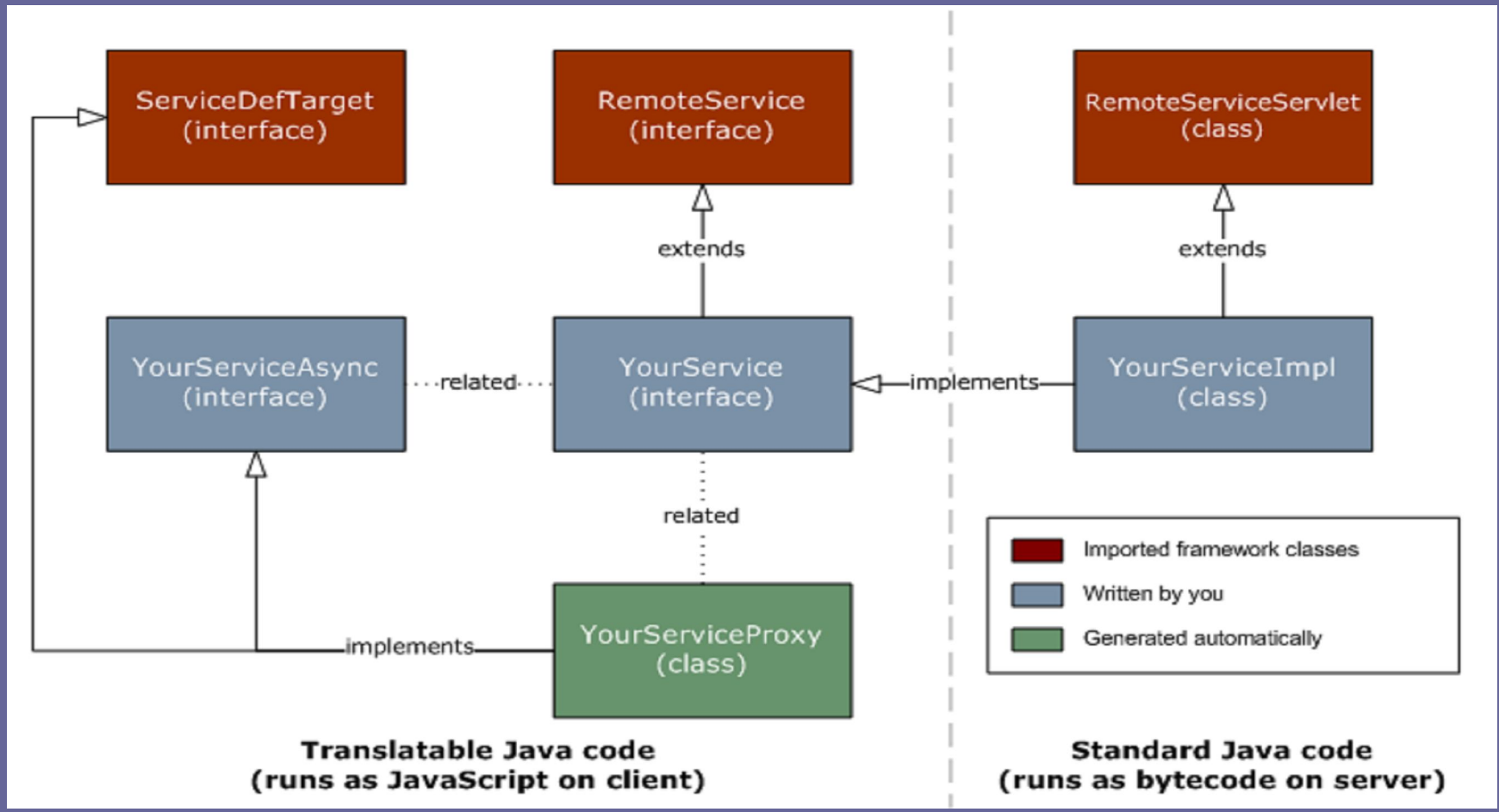
Gestione di eventi

- GWT supporta un’ampia scelta di interfacce per la gestione di eventi, tra cui: KeyboardListener, MouseListener, ClickListener, ChangeListener, FocusListener, ecc.
- Gli elementi dell’interfaccia utente hanno metodi per aggiungere e togliere listener per ognuno dei tipi supportati
  - Ad esempio, SourcesClickEvents contiene i metodi per la manipolazione dell’elenco di ClickListener

RPC in GWT

- Basato su un protocollo proprietario
- Utilizza AJAX
- Lato server, una Servlet è usata per ricevere le richieste
- Il procedimento di creazione è basato su un numero di passi ben definiti

Service “Plumbing” Diagram



Come si scrive un servizio in RPC?

- Creiamo un'interfaccia client-side che rappresenta il servizio
  - Deve estendere RemoteService di GWT, va annotata con @RemoteServiceRelativePath("<path della servlet>")
  - È necessario che tutti i parametri ed il tipo di ritorno estendano Serializable
- Lato server, scriviamo una Servlet, che estende RemoteServiceServlet di GWT ed implementa la prima interfaccia
  - In Hosted mode, <servlet path="*<path della servlet>*" class="*<classe della servlet>*" /> nel file .gwt.xml richiede l'istanziazione
- Creiamo un'interfaccia equivalente, ma asincrona; per ogni metodo dell'interfaccia originaria, ne creiamo uno equivalente, che:
  - Non restituisce nulla (è *void*)
  - Non lancia eccezioni
  - Ha un ulteriore parametro di tipo AsyncCallback<T> (T è il tipo dell'oggetto restituito dal metodo di partenza)

Effettuare la chiamata

- Creaimo un’istanza del proxy creato da GWT per il servizio, tramite GWT.create(<interfaccia del servizio>.class)
- L’oggetto che ci viene restituito è però del tipo della seconda interfaccia (quella asincrona)

Effettuare la chiamata

- Ogni metodo chiamato sull'oggetto restituito agisce sulla servlet, tramite HTTP
- L'oggetto AsyncCallback che bisogna fornire ha due metodi:
  - onSuccess(T), chiamato se la richiesta ha esito positivo
  - onFailure(Throwable), in caso di fallimento