Ingegneria del Software

Corso di Laurea in Informatica per il Management

GWT

Davide Rossi
Dipartimento di Informatica
Università di Bologna

So you want to create a web application?

Minimal list of technologies you need to master:

- HTML
- DOM
- ECMAScript (Javascript)
- CSS
- HTTP
- XML / JSON
- Server-side language (PHP, ASP, JSP, Javascript, ...)

GWT

GWT [/ˈgwɪt/] (originally Google Web Toolkit, but that's being obsoleted) is a set of tools aiming at simplifying (highly dynamic) web application development. A Java-centric approach trying to avoid the tech soup.

Client-side code is transcompiled from Java, content is handled by widgets and panel classes, server-side code is Java (simplified servlets).

You still need CSS, though.

Development toolkit

GWT is mainly a development toolkit, it includes useful tools meant to ease the life of web application programmers.

- High integration with Eclipse
- Powerful development mode
- UI designer

The client side

Java code is transcompiled to Javascript (Java 11 syntax is supported). There are limitations, though:

- No multithreading (and synchronization)
- (Almost) no reflection
- No serialization (but has its own version of it)
- Limited API (parts of `java.lang`, `java.lang.annotation`, `java.math`, `java.io`, `java.sql`, `java.util`, `java.util.logging` are avaliable)

The server side

Server side code is hosted in a Servlet container, no knowledge of Servets API is needed, communications with the client side code is handled by an integrated RPC mechanism. Most services extend the `RemoteServiceServlet` class that handles all the plumbing.

Client-server interaction

The provided RPC mechanism allow client code to invoke services running on the server side by creating a dynamic proxy that handles all the low-level details (marshaling, exceptions, ...). Beware: this is AJAX at heart so invocations are asynchronous: services have to expose an asynchronous interface (derived from the “regular” one) and clients refer to that interface.

Other mechanisms can be used for client-server communications (XML, JSON, ...), libraries are available but support is not integrated.

Host page and entry point

The host page is the entry point of the web application. Client code can add panels/widgets to specific areas of the host page or replace all its contents.

Once the host page is loaded the `onModuleLoad` method of the main class (that has to implement `EntryPoint`) is called.

Widgets, panels and events

UI elements on the web page can be dynamically created/modified using a set of UI classes containing panels (containers) and widget (UI elements).

Event handlers are used to react to user gestures just as in Swing.

If needed, an API is available to directly manipulate the host page's DOM.

GWT RPC

GWT RPC is a simplified method to let client code interact with server code. Server code is structured in services, Java classes that extend `RemoteServiceServlet` and implement an interface describing the operations available to the client. An asynchronous version of the interface has to be provided too.

Clients dynamically create a proxy for each service they want to use (using `GWT.create()`) and interact with them by using their asynchronous interface.

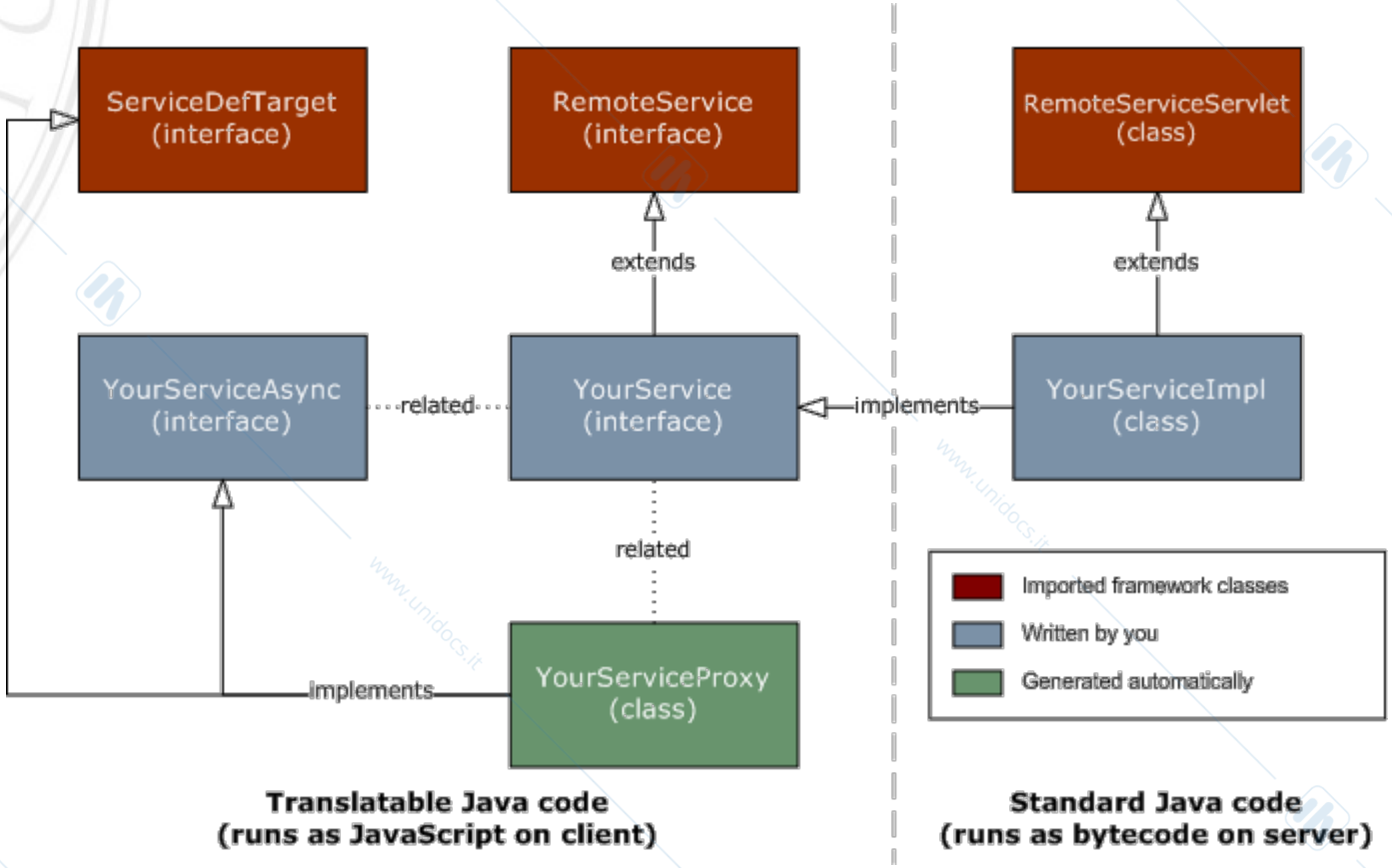
Services and interfaces

```
package com.example.foo.client;

import com.google.gwt.user.client.rpc.RemoteService;

public interface MyService extends RemoteService {
  public String myMethod(String s);
}
```

```
package com.example.foo.server;

import com.google.gwt.user.server.rpc.RemoteServiceServlet;
import com.example.client.MyService;


public class MyServiceImpl extends RemoteServiceServlet implements
    MyService {

  public String myMethod(String s) {
    // Do something interesting with 's' here on the server.
    return s;
  }
}
```

```
package com.example.foo.client;

interface MyServiceAsync {
  public void myMethod(String s, AsyncCallback<String> callback);
}
```

GWT RPC



Sessions

As with all web applications, the server-side code has no knowledge about who is making invocations.

When creating a multi-page GWT application (but remember: GWT applications can be hosted in a single page) sessions can be used to associate information to the requests (such as: the identity of the user making the call). `getThreadLocalRequest().getSession().setAttribute(name, value)` and `getThreadLocalRequest().getSession().getAttribute(name)` can be used to access session data.