GWT Architecture

Reid Holmes
Google Web Toolkit (GWT)
“Architectural Impact on Enterprise Web Application”
First Generation
First Generation

Shortcomings

• lack of a coherent architectural model
  – adhoc scripts development
  – no interaction between scripts

• evolution
  – how do you evolve scripts?
  – written in different languages

• data/information sharing
  – has to be via an external repository
  – difficult to control transactions
First Generation

Shortcomings

- **security**
  - CGI based scripts still a nightmare for admins
  - scripts are executed in native environment
    - server execution environment is directly exposed
    - vulnerabilities in server code are also exposed

- **throughput**
  - a script launches a process – *so what?*

- **tight coupling**
  - each view is coupled to its corresponding script
Second Generation

HTTP response
(HTML Document)

Presentation

JEE Container

Servlet
Services
JDBC

Data Tier
Second Generation

Shortcomings

• server focused
  – most improvements are realized on the server side
• client tier
  – still based on primitives
    • HTML, javascript, CSS etc.
    – not dynamic
• request-response cycle
  – worse than first generation ???
GWT

GWT is a lot more than that

- a paradigm shift away from
  - web application building blocks
  - synchronous communication
- built on standards
  - integrates well
    - with existing server applications, web services
  - composite building blocks
    - HTML, javascript etc are low-level primitives
- separation of concerns
GWT Components

Core
- Java-JS Compiler
- JRE Emulator
- JSNI
- GWT Servlet Ext

RPC

Browser Runtime Environment
- RPC manager
- Browser Integration
- widgets
- XML/JSON
- i18N
Java-JS Compiler

Converts Java to Javascript

- src-to-src compiler
- high level typed language to a script language
  - does this make sense???
- JS code optimization
  - browser engines
  - size
  - security / obfuscation
  - localization
JRE Emulator

Emulates

- core Java classes in Javascript

Composite building blocks

- allows for building composite building blocks
- client tier built on
  - composite building blocks
    - rather than low level primitives
Java Script Native Interface

- **wrapper**
  - for Javascript inside Java code

- **extension point**
  - for integration with non GWT client components

```java
private static native void jsString(String s) /*-{
    $wnd.alert("s is " + s);
}*/;
```
Remote Procedure Call

- replaces HTTP
  - for communication after app boot
- asynchronous – why?
  - breaks the request-response cycle
- supports various protocols
  - Ajax, JSON, GWT
Servlet Extension

Extension of JEE Servlet

• integration with older JEE application
  - get all the JEE benefits for free
• server component
  - facade for business functionality
• evolution
  - highly flexible
GWT Components

Core
- Java-JS Compiler
- JRE Emulator
- JSNI
- GWT Servlet Ext

Browser Runtime Environment
- RPC manager
- Widgets
- XML/JSON
- i18N
Impact

- Scalability
- Reusability
- Interoperability
- Design by contract
- Evolution
- Java based development
Scalability

Improvement in server performance

- (near) stateless servers
  - client tier components truly reside in client tier
  - previously
    - the state was maintained on the presentation tier
    - the view was rendered on the client tier

- optimized communication strategy
  - via aggregation of control/data exchange
  - decrease in server load
  - better bandwidth usage
Reusability

Application

• single code base to support
  – multiple browser engines
  – internationalization
    • i18N versions of the applications

• application broken over reusable modules

Design & development

• OOD
  – *what benefits do we get from OOD?*
Interoperability

Integration / Extension points

- **Javascript native interface (JSNI)**
  - a layer of abstraction for integrating Javascript
    - third party & legacy Javascript libraries
- **server side integration**
  - servlet extension
    - plugs into the JEE platform
    - also possible for other platforms
  - mashups
    - use of diverse web services
Design by Contract

Client tier standardization

• browser runtime environment (BRE)
  – client code has to abide by the BRE interface
  – same enhancement that JEE brought to server tier
  – *isn't that strong coupling between GWT and an application client code?*
    • preserves the architectural integrity
Evolution

Organic growth

- **OOA & OOD**
  - *what does this buy us?*
  - OO Javascript

Rich Internet Applications (RIA)

- HTML & HTTP as the basic building primitive
- prevails where others failed
  - Java applet, ActiveX, Adobe flex
Java based Development

Testing & Tools

- well established frameworks for
  - testing & profiling
- continuous integration
- well supported tools
  - IDEs, profilers etc.
- skills-set standardization
  - development teams