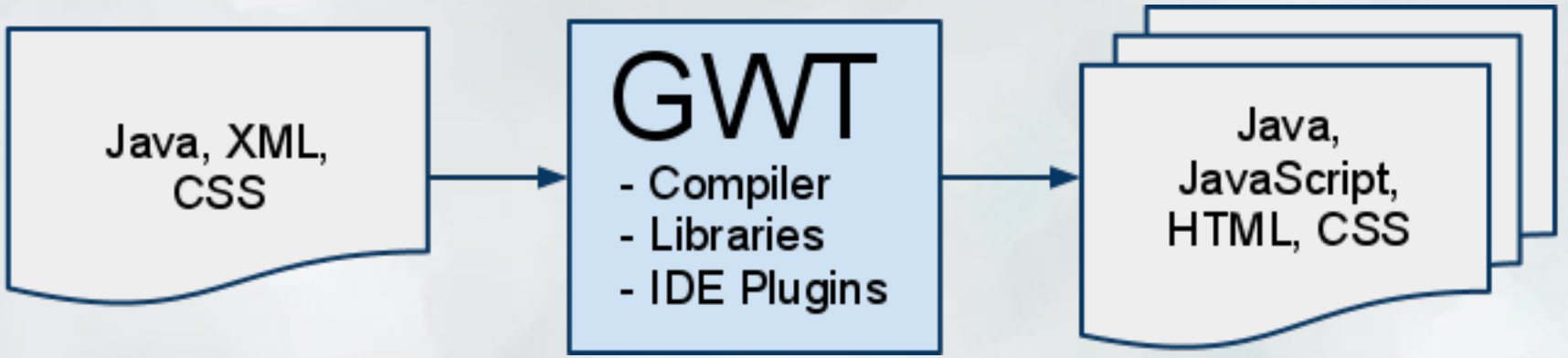
Google Web Toolkit

CWRU Hacker(['][s]['][]) Society

What is GWT?

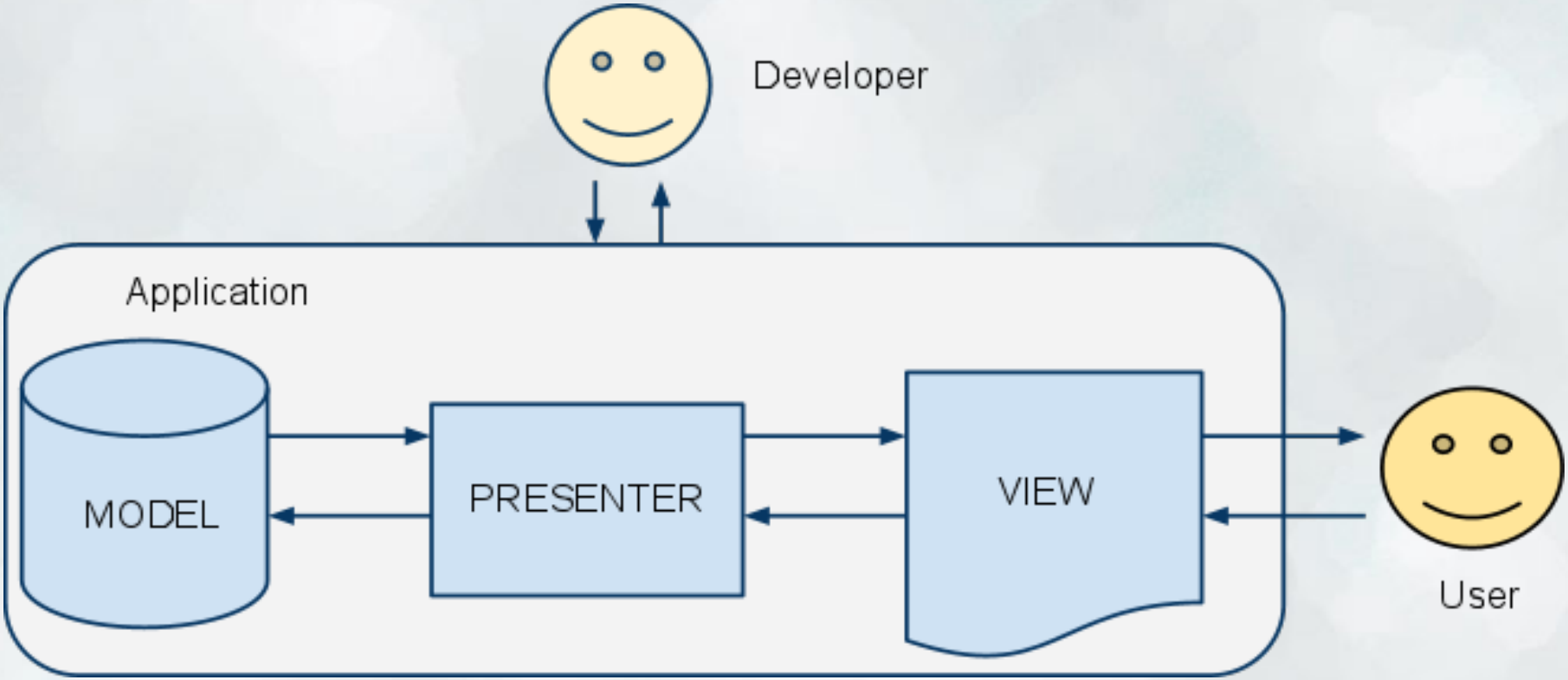


(By the way, it's free and open source under Apache License v2.0.)

Questions to be Answered:

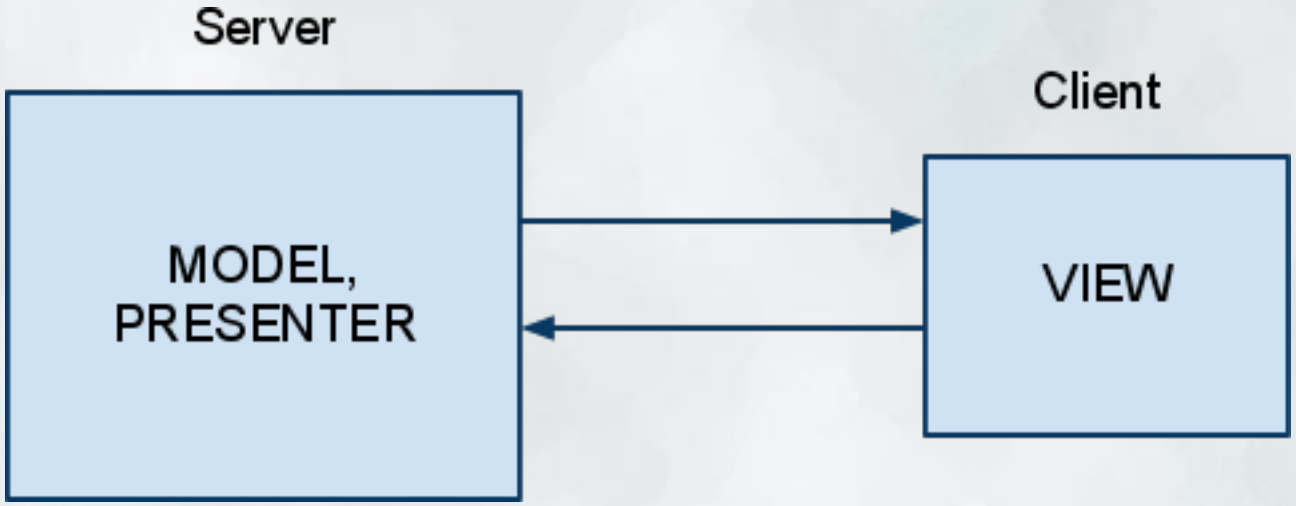
- Why?
- Brief Example Demonstration
- What can it do?
- How does it work?

Quick Background: Model-View-Presenter



Web Applications

In the before time, in the long, long ago...

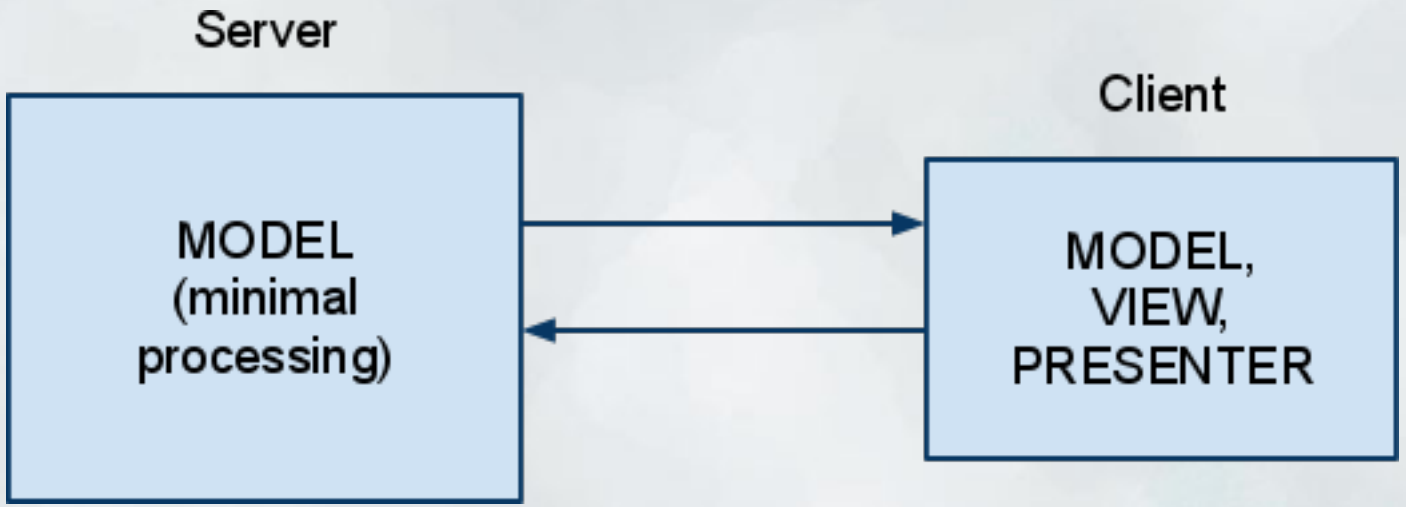


Problems:

- Server is a bottleneck
- Latency

Case in Point:

Solution: Move Processing to Client



(Also keep model cached locally, for responsiveness)

Oops, now we have full-fledged applications running in the browser (written in a "scripting language")!

Writing a Web Application is Easy*!

All you need to do is learn:

- HTML/XML
- CSS
- JavaScript
- Server Language (Java, Python, etc.)

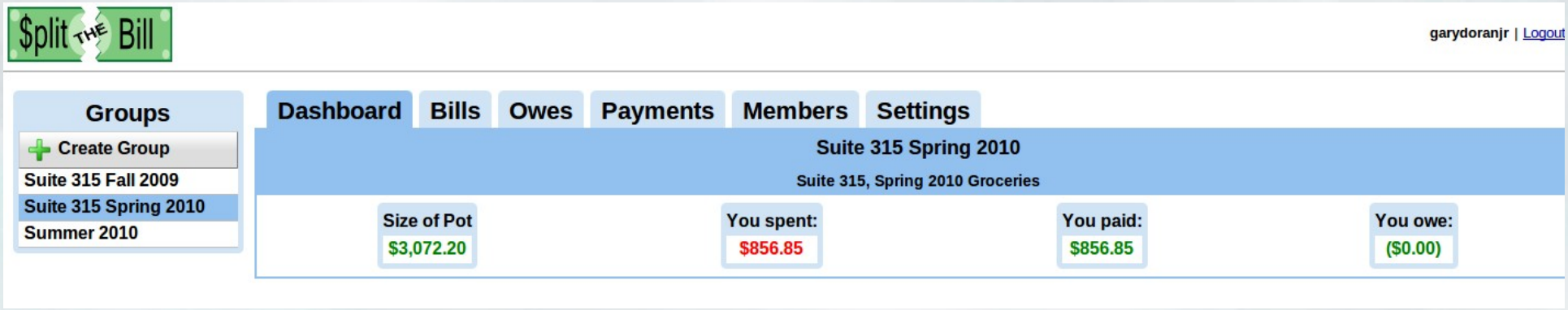
Then, write a bunch of interfaces to pass data between the server and client (e.g. using AJAX).

Finally, don't forget to learn about the nuanced differences between the implementations of HTML/CSS/JavaScript, garbage collection, etc. on various browsers!

(Testing and debugging this application will also be easy.)

*Sarcasm

Split the Bill



Widgets, Widgets Everywhere

- Think of Java Components in AWT/Swing
- Can be completely specified in Java, but are compiled into JavaScript, CSS and HTML
- Compiled differently for different browsers (e.g. rounded corners)
- GWT comes with bunch of widgets, but you can write your own widget libraries, or use others

Java isn't always the best option...

UIBinder:

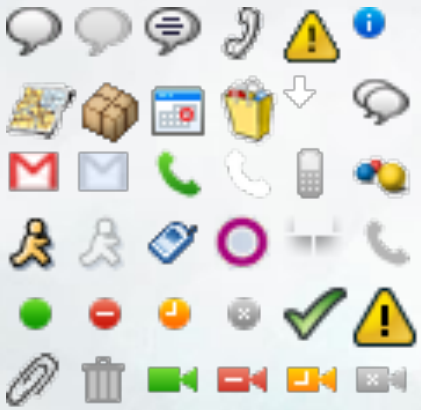
- Allows specifying layout and style in an XML file
- Control code is placed in a Java class, the the compiler "binds" the two together
- Good design pattern (MVP again)
- Some boilerplate, but Eclipse can generate it for you

<- History ->

- Support for Back/Forward operations in browser
- History state manager (like an Undo Manager)

Resource Bundling

- Put a bunch of static resources inside a single object
- Prevent browser from requesting a brazilian tiny resources
- Can be icons or strings (e.g. for externalization)
- Also takes care of caching, etc.



So you're a JS Guru/Ninja...

```
public static native void alert(String msg) /*-{
  $wnd.alert(msg);
}-*/;
```

Don't worry, use can still use native JS (if you must)

Code Splitting

- Don't download the entire application at once!
- Liberally apply the following pattern:

```
GWT.runAsync(new RunAsyncCallback() {
      public void onFailure(Throwable caught) {
        Window.alert("Code download failed");
      }

      public void onSuccess() {
        Window.alert("Hello, AJAX");
      }
    });
```

- The compiler will figure out how to split up your code
- You can pre-fetch code, if necessary

GWT RPC FTW!

- Allows calling Java methods on server directly from client code, asynchronously
- Serialization/Deserialization of objects handled automatically by compiler
- You're not limited to using this to make requests from the server.

Testing and Debugging

- GWT Supports Unit Testing
- Run the web application in "hosted mode"
- Supports "hot swapping" code
- Put a breakpoint in your code. Mind. Blown.

Some Technical Stuff

- Generating multiple compiler outputs
  - Uses a technique called "Deferred Binding"

```
<module>
 <!-- Fall through to this rule is the browser isn't IE or Mozilla -->
 <replace-with class="com.google.gwt.user.client.ui.impl.PopupImpl">
  <when-type-is class="com.google.gwt.user.client.ui.impl.PopupImpl"/>
 </replace-with>

 <!-- Mozilla needs a different implementation due to issue #410 -->
 <replace-with class="com.google.gwt.user.client.ui.impl.PopupImplMozilla">
  <when-type-is class="com.google.gwt.user.client.ui.impl.PopupImpl" />
  <any>
   <when-property-is name="user.agent" value="gecko"/>
   <when-property-is name="user.agent" value="gecko1_8" />
  </any>
 </replace-with>

 <!-- IE has a completely different popup implementation -->
 <replace-with class="com.google.gwt.user.client.ui.impl.PopupImplIE6">
  <when-type-is class="com.google.gwt.user.client.ui.impl.PopupImpl"/>
  <when-property-is name="user.agent" value="ie6" />
 </replace-with>
</module>
```

(You usually don't have to worry about doing this.)

Compiler Principles/Optimizations

- "Don't do at runtime what you can do with a compiler."
- Obfuscated JavaScript/CSS code is minimized and optimized for compression.
- Minimize DOM operations by compiling into HTML as much as possible (as in UIBinder)

Conclusions

- GWT makes it much easier to create powerful, efficient web applications
  - Compile Java to browser-specific "native" code
  - MVP encouraged through UIBinder
  - Code Splitting, Compiler Optimizations, Resource Bundles
  - Development and Testing Tools
- Plenty of good tutorials available online for specifics

Thank You!

Questions/Discussion?

GWT Home: http://code.google.com/webtoolkit/

Example Application: http://split-bill.appspot.com/

Example Source: http://code.google.com/p/split-the-bill/