Google Web Toolkit

GWT

(pronounced *gwit)*

"...a development toolkit for building and optimizing complex browser-based applications."

Thad Humphries
thad.humphries@gmail.com

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Outline

- Things to Know
- Getting Started
- Living in an Asynchronous World
- Other Pieces of Interest
  - UiBinder
  - Browser History
  - Dynamic Binding
  - MVP
- Examples

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Things to Know

- **Initial Release**: May 2006 (1.0)
- **Current Release**: March 2013 (2.5.1)
- **License**: Apache 2.0
- **Requires**: Java 1.6 minimum
- **Recommended**: Eclipse IDE
- **Links**:

  `https://developers.google.com/web-toolkit/` (download, docs, samples)

  `https://code.google.com/p/google-web-toolkit/`

  (source, developer plug-in)

Creative Commons Attribution License 3.0: `http://creativecommons.org/licenses/by/3.0/`

Getting Started

- Creating a sample web application

```
$ webAppCreator -out MyWebApp -junit $JUNIT_HOME/junit-4.10.jar \
com.mycompany.mywebapp.MyWebApp
```

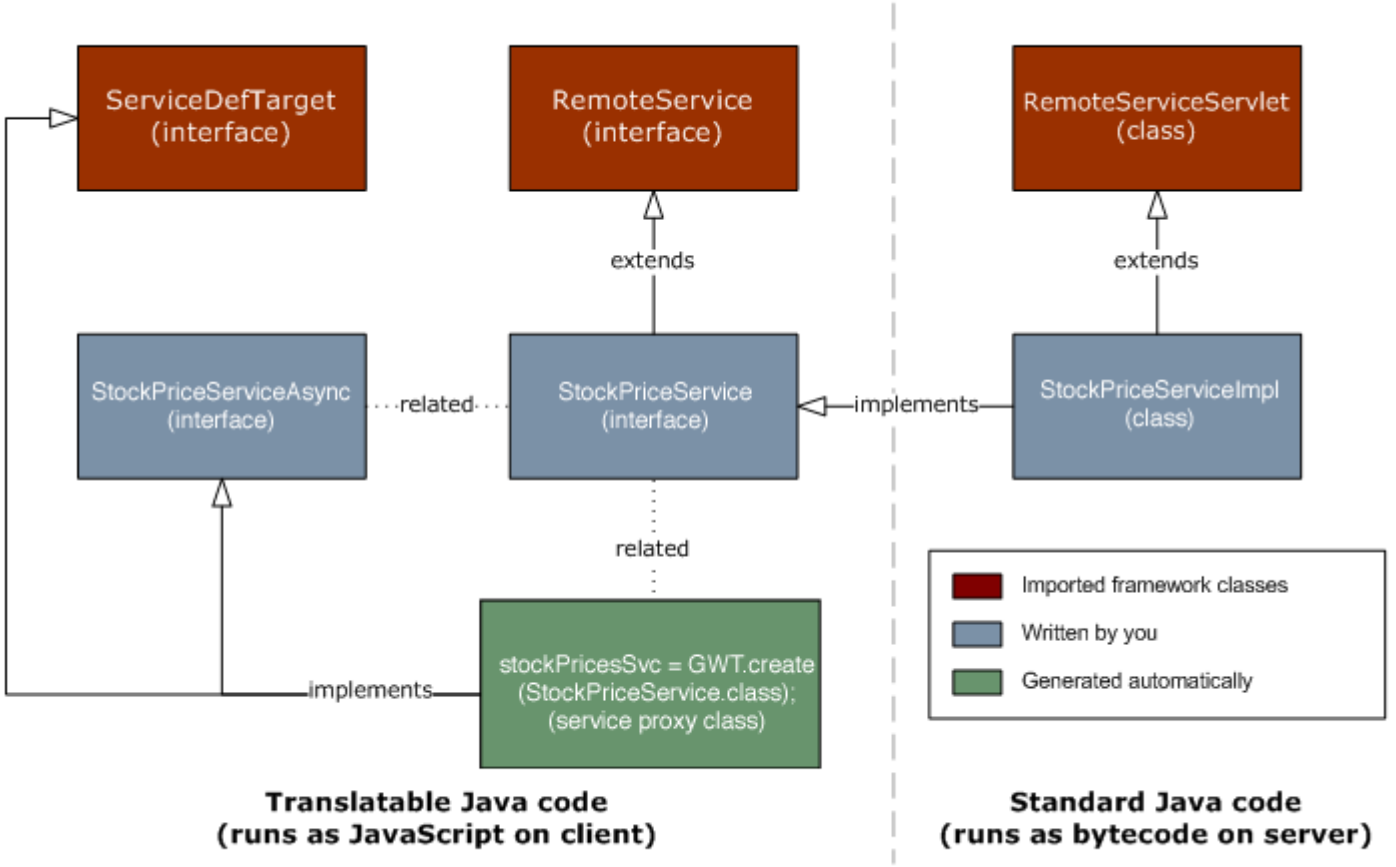
- Running in development mode

```
$ ant devmode
```

- Importing project into Eclipse
- Using the debugger
- Testing with JUnit

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Asynchronous Calls



Source: `https://developers.google.com/web-toolkit/doc/latest/tutorial/RPC`

Creative Commons Attribution License 3.0: `http://creativecommons.org/licenses/by/3.0/`

The GWT RPC Pieces

```
@RemoteServiceRelativePath("greet")
public interface GreetingService extends RemoteService {
  String greetServer(String name) throws IllegalArgumentException;
}


public interface GreetingServiceAsync {
  void greetServer(String input, AsyncCallback<String> callback)
      throws IllegalArgumentException;
}


@SuppressWarnings("serial")
public class GreetingServiceImpl extends RemoteServiceServlet implements
    GreetingService {

  public String greetServer(String input) throws IllegalArgumentException {
    // Verify that the input is valid.
    if (!FieldVerifier.isValidName(input)) {
      // If the input is not valid, throw an IllegalArgumentException back to
      // the client.
      throw new IllegalArgumentException(
          "Name must be at least 4 characters long");
    }

    String serverInfo = getServletContext().getServerInfo();
    String userAgent = getThreadLocalRequest().getHeader("User-Agent");
    ...
```

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Making the RPC Call

```
GreetingServiceAsync greetingService = GWT.create(GreetingService.class);
greetingService.greetServer(textToServer, new AsyncCallback<String>() {
          public void onFailure(Throwable caught) {
            // Show the RPC error message to the user
            dialogBox.setText("Remote Procedure Call - Failure");
            serverResponseLabel.addStyleName("serverResponseLabelError");
            serverResponseLabel.setHTML(SERVER_ERROR);
            dialogBox.center();
            closeButton.setFocus(true);
          }

          public void onSuccess(String result) {
            dialogBox.setText("Remote Procedure Call");
            serverResponseLabel.removeStyleName("serverResponseLabelError");
            serverResponseLabel.setHTML(result);
            dialogBox.center();
            closeButton.setFocus(true);
          }
        });
```

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Serializing Over RPC

- Primitive types: `char`, `byte`, `short`, `int`, `long`, `boolean`, `float`, or `double`
- `String`, `Date`, or primitive wrapper, e.g., `Character`, `Byte`, `Short`, etc.
- Enumerations
- Arrays of serializable types
- A `Serializable` user-defined class
  - Default|no-args constructor
  - All instance fields are serializable
  - Implements `Serializable`

Creative Commons Attribution License 3.0: `http://creativecommons.org/licenses/by/3.0/`

UiBinder Template

HelloWidget.ui.xml

```
<!DOCTYPE ui:UiBinder SYSTEM "http://dl.google.com/gwt/DTD/xhtml.ent">
<ui:UiBinder
    xmlns:ui="urn:ui:com.google.gwt.uibinder"
    xmlns:g="urn:import:com.google.gwt.user.client.ui">
  <ui:style>
    .important {
      font-weight: bold;
    }
  </ui:style>
  <g:HTMLPanel>
    Hello,
    <g:Button styleName="{style.important}" ui:field="button"/>
  </g:HTMLPanel>
</ui:UiBinder>
```

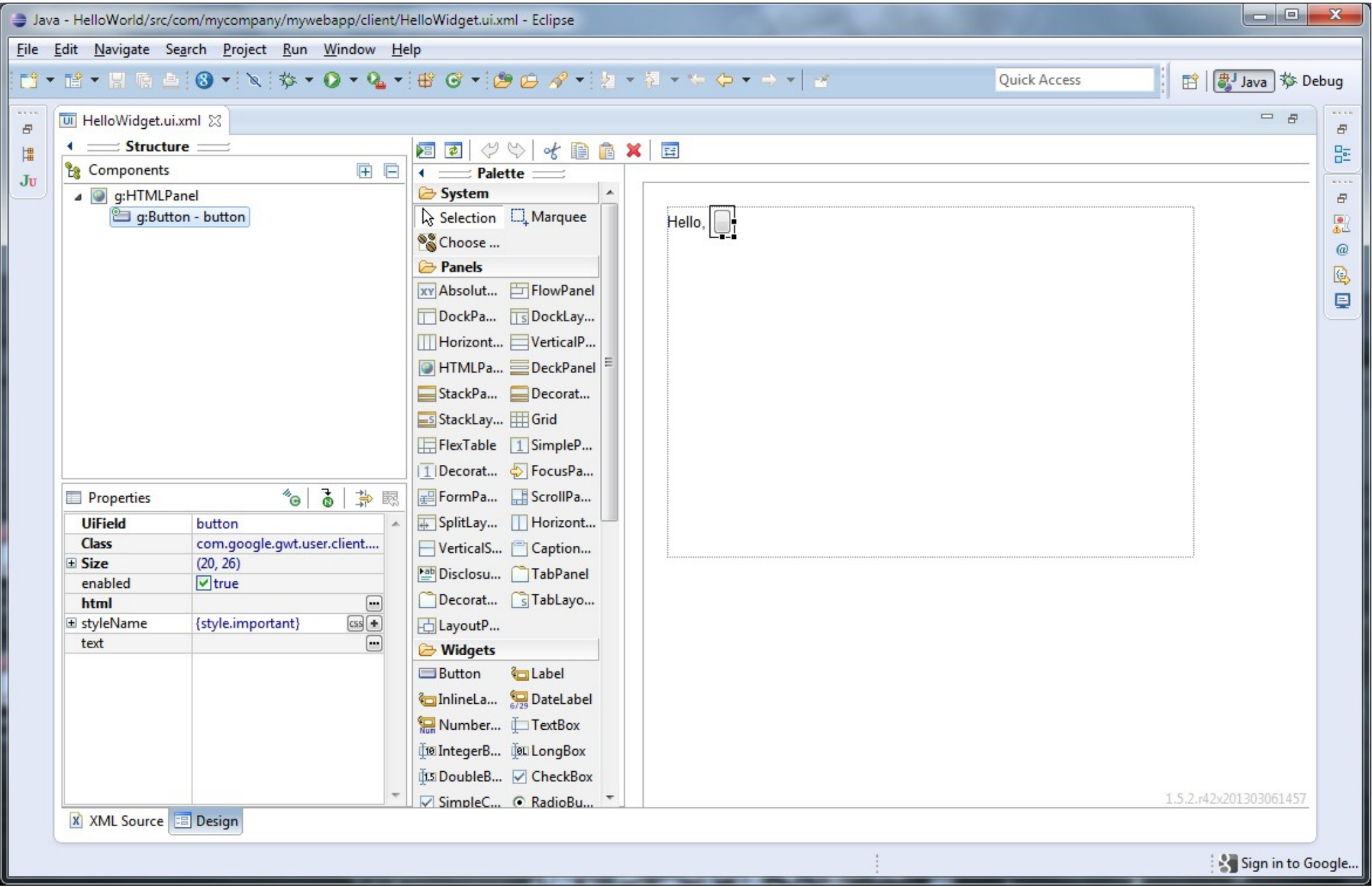
Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Using a Template

```
public class HelloWidget extends Composite implements HasText {
  private static HelloWidgetUiBinder uiBinder = GWT
      .create(HelloWidgetUiBinder.class);
  interface HelloWidgetUiBinder extends UiBinder<Widget, HelloWidget> {
  }
  public HelloWidget() {
    initWidget(uiBinder.createAndBindUi(this));
  }
  @UiField
  Button button;
  public HelloWidget(String firstName) {
    initWidget(uiBinder.createAndBindUi(this));
    button.setText(firstName);
  }
  @UiHandler("button")
  void onClick(ClickEvent e) {
    Window.alert("Hello!");
  }
  public void setText(String text) {
    button.setText(text);
  }
  public String getText() {
    return button.getText();
  }
}
```

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

GWT Designer



Deferred Binding

```
<!-- 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>
```

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

GWT Development Framework

- Browser history handled with Activities and Places
  - EntryPoint
  - EventBus
  - PlaceController
  - AppController
- Primary design pattern: Model-Viewer-Presenter (MVP)
  - *Not required* for Activities and Places but a good match
  - Supports test design
- ClientFactory (speed, deferred binding)

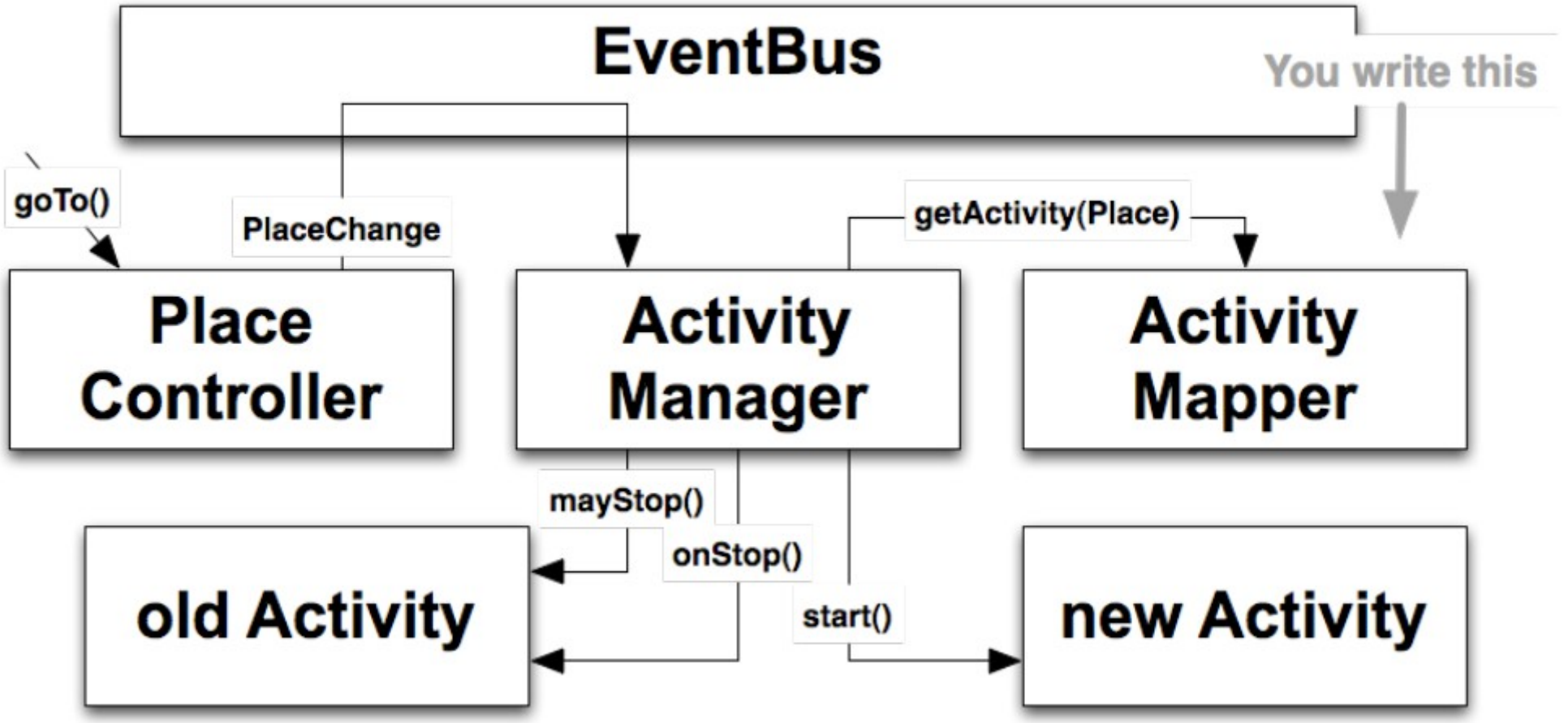
Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Activities and Places

- AppController coordinates `goTo(Place)`
- Places are boilerplate code (cut & paste)
  - Essentially handles the URL token
- ActivityMapper maps Place to Activity
- Activities are passed a Place on creation
  - Analyze token
  - In MVP, pass data to|from View

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Activities and Places



Source: http://www.google.com/events/io/2011/sessions/high-performance-gwt-best-practices-for-writing-smaller-faster-apps.html

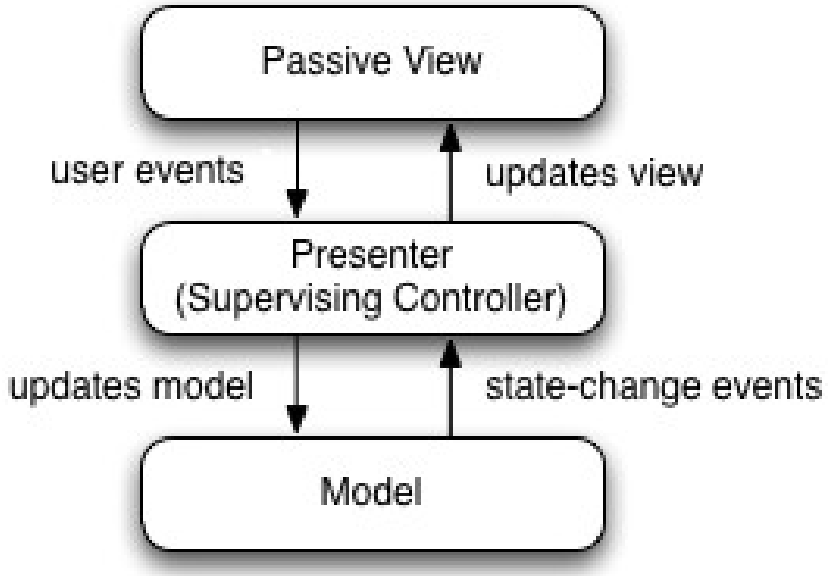
Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

ActivityMapper

```
public class AppActivityMapper implements ActivityMapper {

  private ClientFactory clientFactory;

  public AppActivityMapper(ClientFactory clientFactory) {
    super();
    this.clientFactory = clientFactory;
  }


  @Override
  public Activity getActivity(Place place) {
    if (place instanceof FooPlace)
      return new FooActivity((FooPlace) place, clientFactory);

    if (place instanceof BarPlace)
      return new BarActivity((BarPlace) place, clientFactory);

    ...
    return null;
  }
}
```

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Model-Viewer-Presenter



- View
  - DOM (build once)
  - UIBinder
  - GWTTestCase
- Presenter
  - Activity
  - JUnit
- Model
  - Can be simple POJO
  - Possibly serializable

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Sample View

```
public interface FooView extends IsWidget {
  public interface Presenter {
    void setBar(String s);
    void goTo(Place place);
  }
  void setPresenter(Presenter presenter);
  void updateUI(Foo foo);
}

public class FooViewImpl extends ResizeComposite implements FooView {
  FooView.Presenter presenter;
  @UiField TextBox textBox;

  public FooViewImpl() {...

  public void setPresenter(Presenter presenter) {...

  public void updateUI(Foo foo) {…

  @UiHandler("textBox")
  void changeText(ChangeEvent e) {
    presenter.setBar(textBox.getText());
  }
```

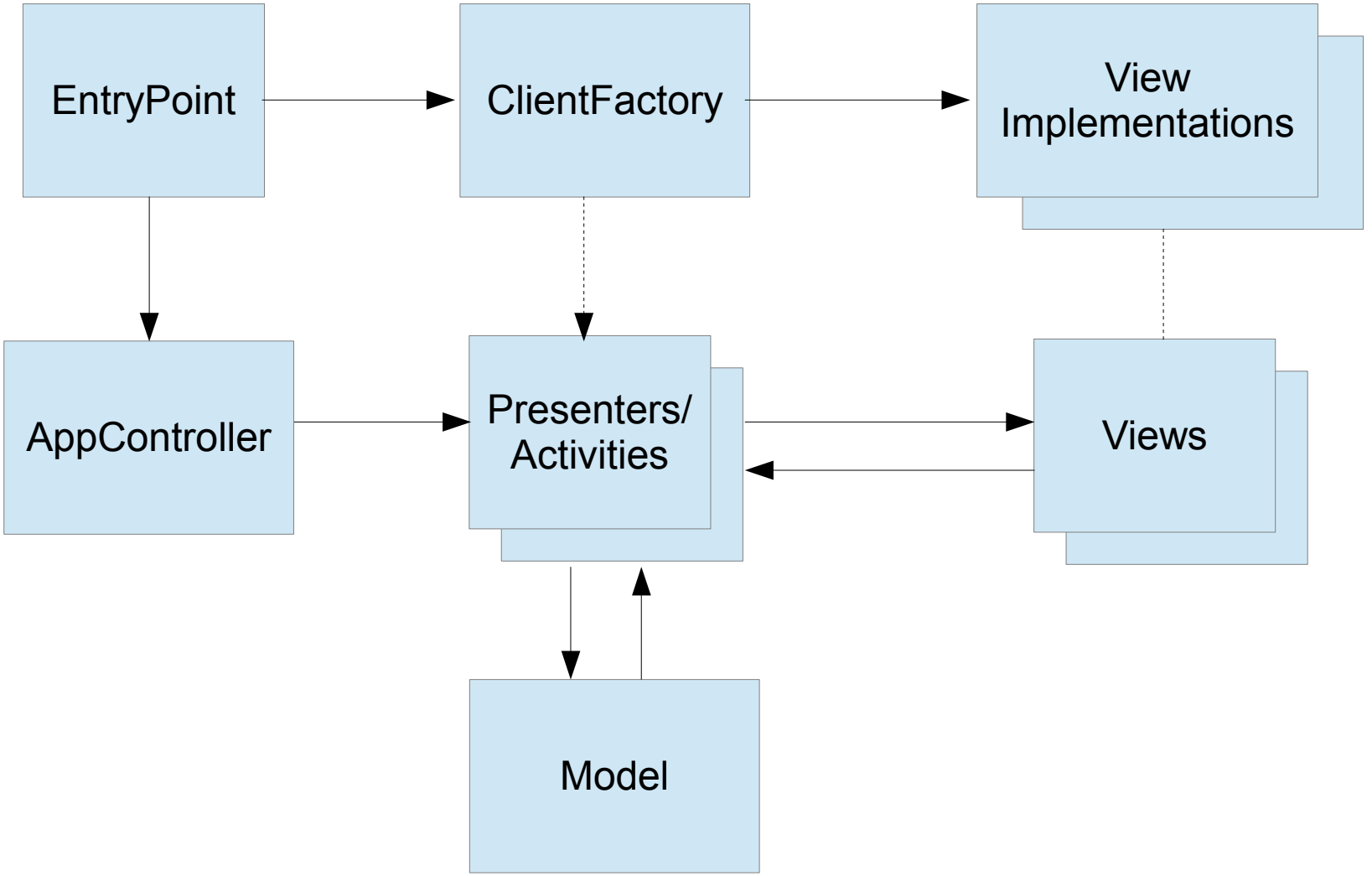
Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Sample Activity

```
public class FooActivity extends AbstractActivity implements FooView.Presenter {
  private Foo foo;
  private final ClientFactory clientFactory;
  private final FooView view;

  public FooActivity(FooPlace place, ClientFactory clientFactory) {
    this.clientFactory = clientFactory;
    this.view = clientFactory.getFooView();
    String token = place.getToken();
  }

  public Widget asWidget() {
    return view.asWidget();
  }

  public void start(AcceptsOneWidget arg0, EventBus arg1) {
    view.setPresenter(this);
  }

  public void onStop() {...

  @Override
  public void setBar(String s) {
    this.foo.bar = s;
    // misc logic; RPC call
    view.updateUI(foo);
  }

  @Override
  public void goTo(Place place) {
    clientFactory.getPlaceController().goTo(place);
  }
}
```

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

How It Fits Together



Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

Why the ClientFactory?

- Create DOM objects once
  - Speed
  - A simple place for mock test objects
- Deferred binding changes DOM on-the-fly

```
<replace-with class="com.google.gwt.sample.mobilewebapp.client.ClientFactoryImpl">
  <when-type-is class="com.google.gwt.sample.mobilewebapp.client.ClientFactory"/>
</replace-with>

<replace-with class="com.google.gwt.sample.mobilewebapp.client.ClientFactoryImplMobile">
  <when-type-is class="com.google.gwt.sample.mobilewebapp.client.ClientFactory"/>
  <when-property-is name="formfactor" value="mobile"/>
</replace-with>

<replace-with class="com.google.gwt.sample.mobilewebapp.client.ClientFactoryImplTablet">
  <when-type-is class="com.google.gwt.sample.mobilewebapp.client.ClientFactory"/>
  <when-property-is name="formfactor" value="tablet"/>
</replace-with>
```

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

GWT Pitfalls

- It's still a browser
  - HTML and JavaScript limitations
  - Single window
- GWT Builder
  - Slow, buggy, poor for global CSS
  - Can't handle dynamic elements
- Embracing asynchronicity
  - You can't just stop and wait
  - Watch for race conditions

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/

More Examples

- Mail (GWT sample with UiBinder)
- Showcase (GWT's kitchen sink)
- Optix Web 8
- Optix COLD 8 (includes SVG)

Creative Commons Attribution License 3.0: http://creativecommons.org/licenses/by/3.0/