MIGRATE LARGE GWT APPLICATIONS

LESSONS LEARNED

Harald Pehl, GWTCon 2017

About Me

Senior Software Engineer at Red Hat

Component Lead of the WildFly Management Console

Working with GWT since 2008

http://hpehl.info

https://github.com/hpehl

@haraldpehl

HAL - WildFly Management Console

History

First commit

Feb 8 2011

5.723 commits / 179 releases

200k LoC

Today

2.9.14.Final

Current Version

Stack

GWT 2.8

GWTP / GIN

Some Code Generation

UI

GWT Stock Widgets

Cell Widgets

No Widget Library

⇨ Lots of Repeating Code ⇦

New Version

Stack

Latest GWT Version

GWTP / GIN

Lots of Code Generation

UI

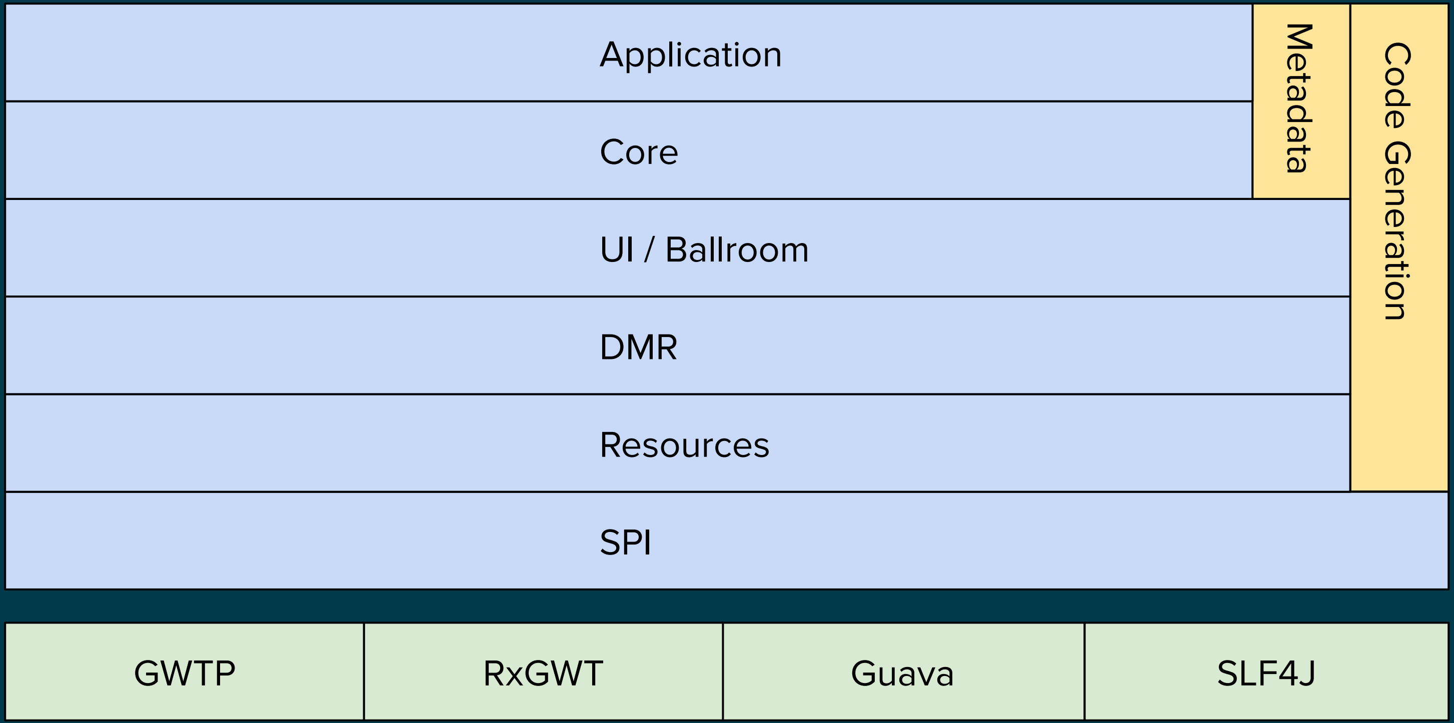
No Widgets

PatternFly

3rd Party Libraries

⇨ Cleaner Architecture ⇦

Architecture



Application

Core

UI / Ballroom

DMR

Resources

SPI

Metadata

Code Generation

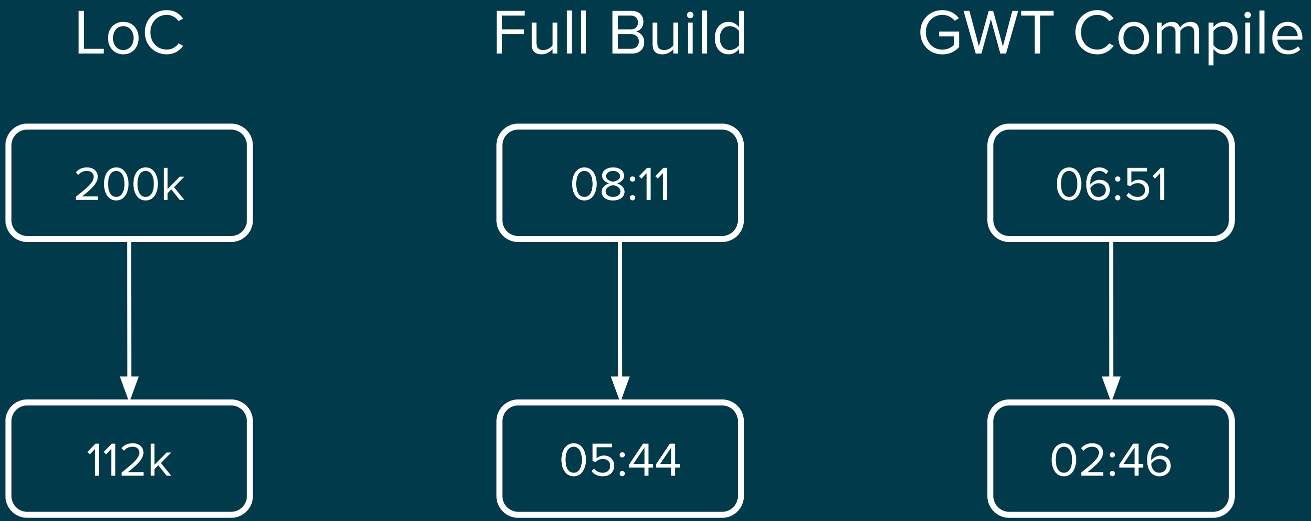
GWTP

RxGWT

Guava

SLF4J

Comparison



LoC

200k

112k

Full Build

08:11

05:44

GWT Compile

06:51

02:46

Demo

http://localhost:9090/

User Interface

PatternFly

Elemento

MBUI

3rd Party Libs

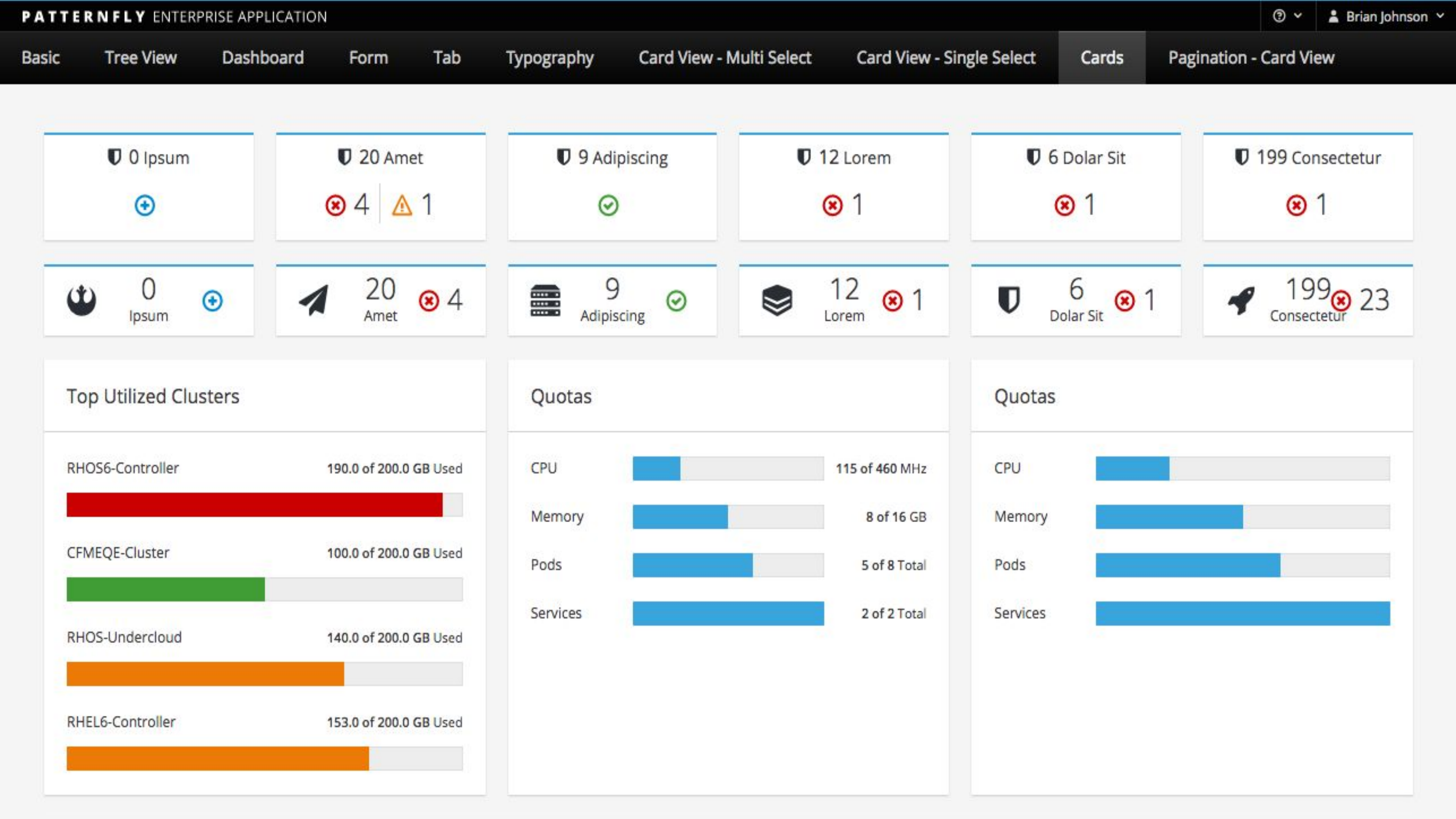
PatternFly

UI Framework for Enterprise Web Applications

Pattern Library

- Communication
- Content Views
- Data Visualization
- Forms & Control, Navigation & Widgets

Common Styles, Colors & Icons



0 Ipsum ⊕

20 Amet ⊗ 4 ⚠ 1

9 Adipiscing ✓

12 Lorem ⊗ 1

6 Dolar Sit ⊗ 1

199 Consectetur ⊗ 1

0 Ipsum ⊕

20 Amet ⊗ 4

9 Adipiscing ✓

12 Lorem ⊗ 1

6 Dolar Sit ⊗ 1

199 Consectetur ⊗ 23

Top Utilized Clusters

RHOS6-Controller 190.0 of 200.0 GB Used

CFMEQE-Cluster 100.0 of 200.0 GB Used

RHOS-Undercloud 140.0 of 200.0 GB Used

RHEL6-Controller 153.0 of 200.0 GB Used

Quotas

CPU 115 of 460 MHz

Memory 8 of 16 GB

Pods 5 of 8 Total

Services 2 of 2 Total

Quotas

CPU

Memory

Pods

Services

Elemento

Type Safe Builders and Event Handlers

HTML Templates w/ Expression Support

Helper Methods

```
HTMLDivElement div = div().css(progressContainewr)
       .add(div().css(progressDescription)
               .title(label)
               .textContent(label))
       .add(div().css(progressCss)
               .add(valueBar = div().css(progressBar)
                       .title(Names.NOT_AVAILABLE)
                       .attr(ROLE, PROGRESSBAR)
                       .aria(VALUE_MIN, "0")
                       .add(valueElement = span().asElement())
                       .asElement())
               .add(remainingBar = div().css(progressBar, progressBarRemaining)
                       .title(Names.NOT_AVAILABLE)
                       .add(remainingElement = span().css(srOnly).asElement())
                       .asElement()))
       .asElement();
```

```
public class Breadcrumb implements IsElement<HTMLElement> {

   private final HTMLElement root;

   public Breadcrumb() {
       root = ol().css(breadcrumb).asElement();
   }

   public void clear() {
       Elements.removeChildrenFrom(root);
   }

   public Breadcrumb append(String segment, SegmentHandler handler) {
       root.appendChild(li()
               .add(a().css(clickable)
                       .textContent(segment)
                       .on(click, e -> handler.onClick()))
               .asElement());
       return this;
   }

   @Override
   public HTMLElement asElement() {
       return root;
   }
}
```

MBUI

XML schema for common UI building blocks

- Navigation
- HTML
- CRUD Operations
- Tables
- Forms

Annotation Processor

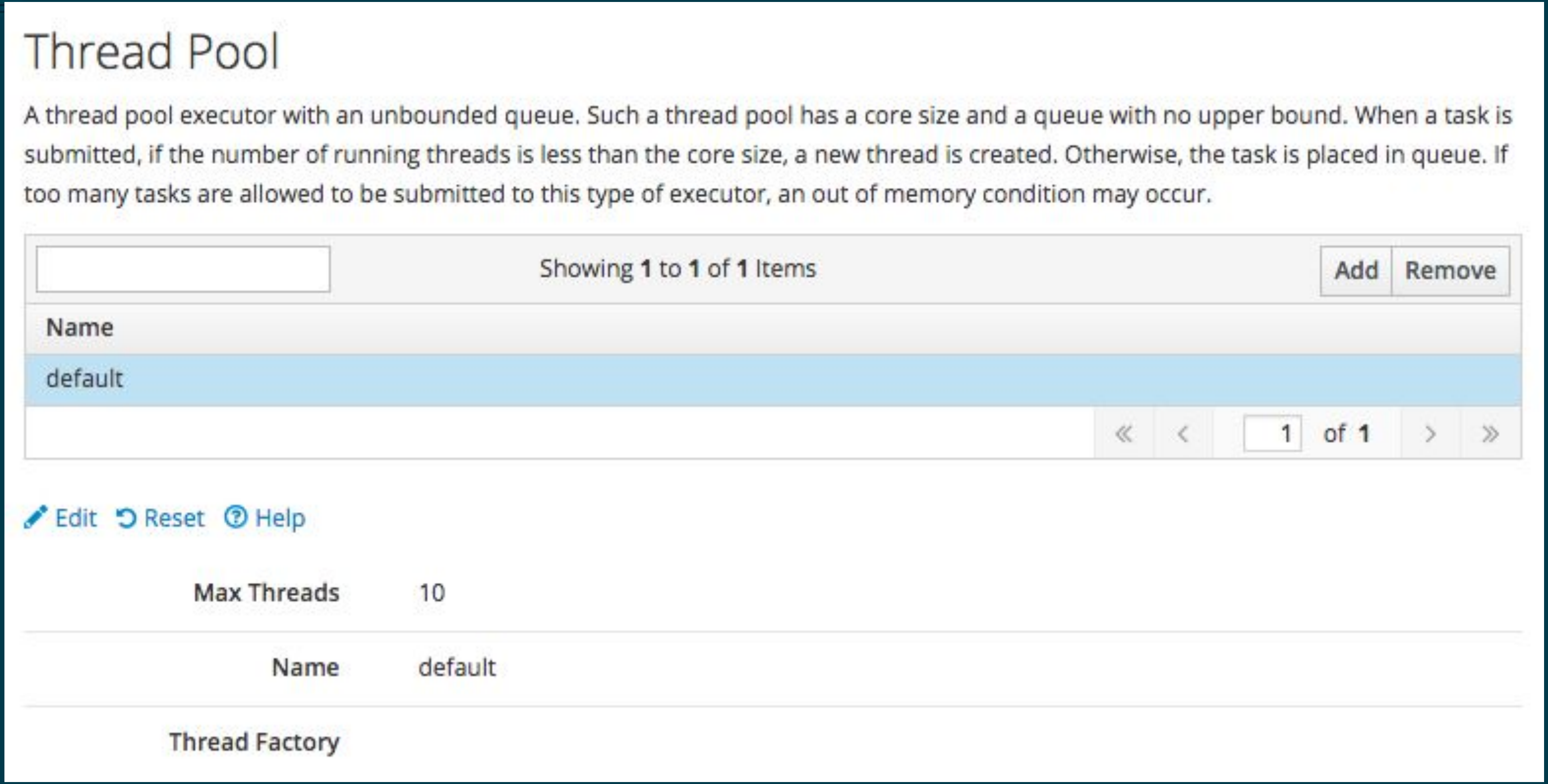
```
<sub-item id="ejb-thread-pool-item" title="Thread Pool">
   <metadata address="/{selected.profile}/subsystem=ejb3/thread-pool=*">
       <h1>Thread Pool</h1>
       <p>${metadata.getDescription().getDescription()}</p>
       <table id="ejb-thread-pool-table" title="Thread Pool" form-ref="ejb-thread-pool-form">
           <actions>
               <action handler-ref="add-resource"/>
               <action handler-ref="remove-resource" scope="selected"
                       name-resolver="${table.selectedRow().getName()}"/>
           </actions>
           <columns>
               <column name="name" value="${row.getName()}"/>
           </columns>
       </table>
       <form id="ejb-thread-pool-form" title="Thread Pool" auto-save="true" reset="true"
             name-resolver="${form.getModel().getName()}"/>
   </metadata>
</sub-item>
```

```java
@MbuiView
public abstract class EjbView
      extends MbuiViewImpl<EjbPresenter>
      implements EjbPresenter.MyView {

   public static EjbView create(MbuiContext mbuiContext) {
       return new Mbui_EjbView(mbuiContext);
   }

   @MbuiElement("ejb-thread-pool-table") Table<NamedNode> threadPoolTable;
   @MbuiElement("ejb-thread-pool-form") Form<NamedNode> threadPoolForm;

   EjbView(MbuiContext mbuiContext) {
      super(mbuiContext);
   }
}
```



Thread Pool

A thread pool executor with an unbounded queue. Such a thread pool has a core size and a queue with no upper bound. When a task is submitted, if the number of running threads is less than the core size, a new thread is created. Otherwise, the task is placed in queue. If too many tasks are allowed to be submitted to this type of executor, an out of memory condition may occur.

Showing **1** to **1** of **1** Items

Name
default

1 of **1**

Max Threads	10
Name	default
Thread Factory	

http://localhost:9090/#ejb3-configuration

3rd Party Libraries

```
"dependencies": {
  "ace-builds": "^1.2.6",
  "datatables.net": "~1.10.13",
  "datatables.net-buttons": "~1.2.2",
  "datatables.net-keytable": "^2.1.3",
  "datatables.net-select": "~1.2.0",
  "font-awesome": "~4.7.0",
  "js-cookie": "~2.1.3",
  "javascript-auto-complete": "1.0.4",
  "jquery": "~2.2.4",
  "jstree": "~3.3.3",
  "patternfly": "~3.26.1",
  "tagmanager": "~3.0.2",
  "zeroclipboard": "^2.2.0"
}
```

```java
@JsType(isNative = true)
public abstract class Modal {

    @JsFunction @FunctionalInterface
    public interface ModalHandler {
        void handle();
    }

    @JsType(isNative = true, namespace = GLOBAL, name = OBJECT)
    public static class ModalOptions {

        public boolean keyboard;

        @JsOverlay
        public static ModalOptions create(boolean closeOnEsc) {
            ModalOptions options = new ModalOptions();
            options.keyboard = closeOnEsc;
            return options;
        }
    }

    @JsMethod(namespace = GLOBAL)
    public native static Modal $(@NonNls String selector);

    public native void modal(ModalOptions modalOptions);

    public native void on(@NonNls String event, ModalHandler handler);
}
```

Model View Presenter

MVP

GWTP

- DI through GIN
- Simple and powerful history management
- Support for nested presenters
- Lifecycle events using GWT eventbus
- Lazy instantiation for presenters and views
- Effortless and efficient code splitting

Widget ↔ Elemental2 Adapter

Escape From Callback Hell

RxGWT

GWT specific bindings for RxJava

Turn callbacks into “Rx types”

Developed and maintained by @ibaca

Documentation

- http://reactivex.io/
- https://github.com/ReactiveX/RxJava
- https://github.com/intendia-oss/rxgwt

```java
interface FooService extends RemoteService {
   void prepare();
   void main();
   void cleanup();
}


interface FooServiceAsync {
   void prepare(AsyncCallback<Void> async);
   void main(AsyncCallback<Void> async);
   void cleanup(AsyncCallback<Void> async);
}
```

```
public class ExecuteInOrder implements EntryPoint {

    @Override
    public void onModuleLoad() {
        FooServiceAsync service = GWT.create(FooService.class);

        service.prepare(new AsyncCallback<Void>() {
            public void onFailure(Throwable throwable) {
                console.log("Failed: " + throwable.getMessage());
            }

            public void onSuccess(Void aVoid) {
                service.main(new AsyncCallback<Void>() {
                    public void onFailure(Throwable throwable) {
                        console.log("Failed: " + throwable.getMessage())
                    }

                    public void onSuccess(Void aVoid) {
                        service.cleanup(new AsyncCallback<Void>() {
                            public void onFailure(Throwable throwable) {
                                console.log("Failed: " + throwable.getMessage())
                            }

                            public void onSuccess(Void aVoid) {
                                console.log("Finished")
                            }
                        });
                    }
                });
            }
        });
    }
}
```

```
public class ExecuteInOrder implements EntryPoint {

    @Override
    public void onModuleLoad() {
        FooServiceAsync service = GWT.create(FooService.class);

        Completable prepare = Completable.fromEmitter(em -> service.prepare(callback(em)));
        Completable main = Completable.fromEmitter(em -> service.main(callback(em)));
        Completable cleanup = Completable.fromEmitter(em -> service.cleanup(callback(em)));

        Completable.concat(prepare, main, cleanup)
                .doOnError(throwable -> console.log("Failed: " + throwable.getMessage()))
                .doOnCompleted(() -> console.log("Finished"))
                .subscribe();
    }

    AsyncCallback<Void> callback(CompletableEmitter emitter) {
        return new AsyncCallback<Void>() {
            @Override
            public void onFailure(Throwable throwable) { emitter.onError(throwable); }

            @Override
            public void onSuccess(Void aVoid) { emitter.onCompleted(); }
        };
    }
}
```

```
public class RxDemo implements EntryPoint {

   @Override
   public void onModuleLoad() {
       Observable.interval(1, SECONDS)
               .take(10)
               .flatMapSingle(this::xhr)
               .forEach(timestamp -> console.log(timestamp));
   }

   Single<String> xhr(long id) {
       String url = "http://server.test-cors.org/server?enable=true&id=" + id;
       return Single.fromEmitter(emitter -> {
           XMLHttpRequest xhr = new XMLHttpRequest();
           xhr.open("GET", url);
           xhr.onload = event -> emitter.onSuccess(new Date().toLocaleTimeString());
           xhr.send();
       });
   }
}
```

Annotation Processing

Annotation Processing

Registries

Composite GIN

MBUI

EsDoc

```java
package org.jboss.hal.processor;

@AutoService(Processor.class)
public class NameTokenProcessor extends AbstractProcessor {

    private final Set<TokenInfo> tokenInfos;

    public NameTokenProcessor() {
        super(NameTokenProcessor.class, TEMPLATES);
        tokenInfos = new HashSet<>();
    }

    @Override
    protected boolean onProcess(Set<? extends TypeElement> annotations, RoundEnvironment roundEnv) {
        for (Element e : roundEnv.getElementsAnnotatedWith(NameToken.class)) {
            TypeElement tokenElement = (TypeElement) e;
            NameToken nameToken = tokenElement.getAnnotation(NameToken.class);
            TokenInfo tokenInfo = new TokenInfo(nameToken.value()[0]);
            tokenInfos.add(tokenInfo);
            ...
        }
        return false;
    }
}
```

```
<#-- @ftlvariable name="generatedWith" type="java.lang.String" -->
<#-- @ftlvariable name="packageName" type="java.lang.String" -->
<#-- @ftlvariable name="className" type="java.lang.String" -->
<#-- @ftlvariable name="tokenInfos"
type="java.util.Set<org.jboss.hal.processor.NameTokenProcessor.TokenInfo>" -->
package ${packageName};

import ...

/*
 * WARNING! This class is generated. Do not modify.
 */
@Generated("${generatedWith}")
public class ${className} implements org.jboss.hal.meta.token.NameTokens {

   private final Set<String> tokens;

   public ${className}() {
       this.tokens = new HashSet<>();
       <#list tokenInfos as tokenInfo>
       this.tokens.add("${tokenInfo.token}");
       </#list>
   }

   @Override
   public Set<String> getTokens() { return tokens; }
}
```

Annotation Processing

Freemarker Templates

Use 3rd party libraries

- https://github.com/google/auto/tree/master/common
- https://github.com/google/auto/tree/master/service

Test your annotation processor!

- https://github.com/google/compile-testing

```java
public class FormTest {

    @Test
    public void simpleForm() {
        Compilation compilation = javac()
                .withOptions("-proc:only")
                .withProcessors(new MbuiViewProcessor())
                .compile(JavaFileObjects.forResource("SimpleForm.java"));

        assertThat(compilation)
                .generatedSourceFile("Mbui_SimpleForm")
                .hasSourceEquivalentTo(
                    JavaFileObjects.forResource("Mbui_SimpleForm.java"));
    }
}
```

Build Process

Build & Run

Maven First

3rd Party Libraries: Frontend Maven Plugin

- NPM
- Bower
- Grunt

Build: Maven Plugin for GWT by Thomas Broyer

Run: Browser Dev Tools

Open Issues

Open Issues

JSNI

GWT.create()

ClientBundle / i18n

GIN

Code Splitting

Links

https://github.com/hal/hal.next/

https://github.com/hal/elemento

http://www.patternfly.org/

http://wildfly.org/

Thanks!
Questions?