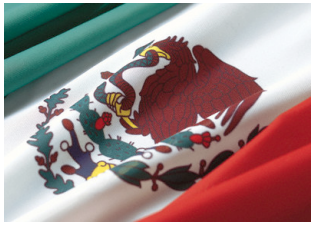
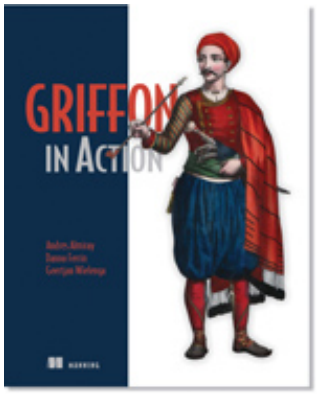
JSR 377: WHAT'S UP AND WHAT'S NEXT

ANDRES ALMIRAY
@AALMIRAY

IX-CHEL RUIZ
@IXCHELRUIZ



canoo



JAVA™
CHAMPIONS

PREVIOUS ATTEMPTS

JSR 193 – Client Side Container

JSR 296 – Swing Application Framework

JSR 295 – Beans Binding

JSR 296 had the following goals

- application life cycle
- localized resources (and injection)
- persisted session state
- loosely coupled actions
- targeted only Swing for obvious reasons

WHICH UI TOOLKIT?

FRAMEWORKS

Eclipse 4 Platform, NetBeans

Griffon, Basilisk, Gluon Particle,

DataFX, JacpFX, MvvmFX, JVx,

Jrebirth, and more …

FRAMEWORKS

Many of the listed frameworks offer the following capabilities implemented in different ways:

- **application life cycle**
- **localized resources (and injection)**
- **persisted session state**
- **loosely coupled actions**
- **dependency injection**
- **event system**
- **centralized error management**
- **extension points via plugins**

TARGET ENVIRONMENT

All of the listed frameworks support the Desktop as target environment.

Only a few can be used in an Embedded environment (where Java SE is supported).

Embedded Java UI applications can be built as applications that target the Desktop; share codebase even.

GOALS OF JSR 377

Target Desktop and Embedded environments

Support several toolkits

Be an standalone JSR, i.e, no need to include it in the JDK

Leverage existing JSRs:

- JSR 330 – Dependency Injection
- JSR 365 – Event bus (from CDI 2.0 ?)

CORE FEATURES

application life cycle

localized resources (and injection)

configuration

MVC artifacts

loosely coupled actions

dependency injection

event system

centralized error management

extension points via plugins

POSSIBLE ADDITIONS

Runtime:

- **persisted session state**
- **artifact introspection API**

Buildtime:

- **test support**
- **deployment**

APPLICATION PHASES

```
package javax.application;


public enum ApplicationPhase {
    INITIALIZE,
    STARTUP,
    READY,
    MAIN,
    SHUTDOWN
}
```

APPLICATION LIFECYCLE

```
packahe javax.application;

public interface Application {
    void initialize();

    void startup();

    void ready();

    boolean shutdown();

    boolean canShutdown();

    void addShutdownHandler(@Nonnull ShutdownHandler handler);

    void removeShutdownHandler(@Nonnull ShutdownHandler handler);
}
```

APPLICATION PROPERTIES

```
package javax.application;

public interface Application {
    . . .
    Configuration getConfiguration();

    ApplicationPhase getPhase();

    Locale getLocale();

    String[] getStartupArguments();
    . . .
}
```

UI THREADING

Toolkit	isUIThread	runSync	RunAsync
Swing	yes	yes	yes
JavaFX	yes	no	yes
SWT	yes	yes	yes
Pivot	yes	yes	yes
Lanterna	yes	yes	no

UI THREADING

```
package javax.application;

public interface ThreadingHandler {
    boolean isUIThread();

    void runInsideUIAsync(Runnable runnable);

    void runInsideUISync(Runnable runnable);

    <R> R runInsideUISync(Callable<R> callable);

    void runOutsideUI(Runnable runnable);
}
```

I18N

```
package javax.application;

public interface MessageSource {
    String getMessage(String key)
         throws NoSuchMessageException;

    String getMessage(String key,
                      String defaultValue);
}
```

Combined arguments (Object[] and/or List), Locale

RESOURCE INJECTION

```
package javax.application;

@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.FIELD, ElementType.METHOD})
public @interface InjectedResource {
    String key() default "";

    String[] args() default {};

    String defaultValue() default "";

    String format() default "";
}
```

RESOURCE INJECTION

```
import javax.application.InjectedResource;


import javafx.scene.paint.LinearGradient;


public class SomeBean {
    @InjectedResource
    private LinearGradient gradient;
}
```

RESOURCES

Desktop|Embedded Application API

https://jcp.org/en/jsr/detail?id=377

https://github.com/jsr377/

http://jsr377.github.io/site/

THANK YOU!

HTTP://PEOPLE.CANOO.COM/SHARE

ANDRES ALMIRAY
@AALMIRAY

IX-CHEL RUIZ
@IXCHELRUIZ