Because code should be written only once
Because code should be well tested
Because code should be shared
Genesis

How did we get there?
A LITTLE BIT OF HISTORY

• Was born in **2006** inside the OW2 EasyBeans project

• Used in **JOnAS, CAROL, CMI, EasyBeans, Shelbie, JASMINe for years**

• **Extracted** as a separate maven project in 2008

• More than **35 releases** up to today!

• Proposed as **top level** OW2 project in 2012

• We’re here **now**:)

**OW2 Annual Conference 2012, 27-29 November, Orange Labs, Paris**

www.ow2.org
Modules

15 minutes is so short, and we have so many modules
QUITE BASIC STUFF

• I18n and Logger
  • Internationalized loggers, Java 5 style

• Xml parsing
  • Obtain a DocumentParser, extract values from Elements (trimmed, Properties, default values, ...)

• Xml Config

• Simple Java/Xml mapper (read your Xml configuration files)

• Base64
  • Encode, decode

• File/URL conversion

• Classloader aware ObjectInputStream

OW2 Annual Conference 2012, 27-29 November, Orange Labs, Paris
www.ow2.org
ANNOTATION PROCESSOR

Want a nice way to visit annotations of an instance?

- Traverse Class, Method and Field declaration
- Support inheritance

- Call interested AnnotationHandler(s)

```java
public class SampleAnnotationHandler extends AbstractAnnotationHandler {
    public boolean isSupported(Class<?> clazz) {
        return clazz.equals(Hello.class);
    }

    @Override
    public void process(Annotation annotation, Class<?> clazz, Object target)
            throws ProcessorException {
        System.out.printf("Type %s is annotated with @%s(%s)",
                clazz,
                annotation.annotationType().getSimpleName(),
                ((Hello) annotation).name());
    }
}
```
EVENT SYSTEM

- On steroid topic notification system
  - Hierarchical, multi-threaded
- Dispatcher fires event on a topic
- Listeners receive events matching a topic pattern
  - Declares a priority (SYNC, ASYNC)

```java
// Simple asynchronous listener
IEventListener listener = new IEventListener() {
    public void handle(IEvent event) {
        // Do your own casting here
        System.out.printf("Received Event %s", event);
    }
    public EventPriority getPriority() {
        return EventPriority.ASYNC_HIGH;
    }
    public boolean accept(IEvent event) { return true; }
};
```
// Create a Dispatcher
EventDispatcher dispatcher = new EventDispatcher();
dispatcher.setNbWorkers(3);
dispatcher.start();

// Listen to all Event published in /result/**
eventService.registerListener(listener, "/*result/.*");

// Topic registration
eventService.registerDispatcher("/result/success", dispatcher);

// Dispatch an Event
dispatcher.dispatch(new MyEvent("/result/.*"));
POOL

• Simple Pooling API
  • Pool size, waiters, timeout
  • Thread-safe

• Basic implementation
  • Synchronous

• Advanced implementation
  • Asynchronous, composable

```java
public static void main(String[] args) throws Exception {
    PoolFactory<Pooled, Long> poolFactory = new PooledFactory();
    JPool<Pooled, Long> pool = new JPool<Pooled, Long>(poolFactory);
    pool.start();
    Pooled one = pool.get();
    Pooled two = pool.get();
    Pooled three = pool.get();
    Pooled four = pool.get(); // Will block until timeout or a release
}

private static final class PooledFactory implements PoolFactory<Pooled, Long> {
    public Pooled create(Long clue) throws PoolException {
        return new Pooled(clue);
    }
    public boolean isMatching(Pooled instance, Long clue) {
        return instance.id == clue;
    }
    public void remove(Pooled instance) {
        // Destroy instance
    }
    public boolean validate(Pooled instance, PoolEntryStatistics stats) {
        return true;
    }
}

@Pool(min = 1, max = 3)
private static class Pooled {
    long id;
    public Pooled(Long clue) { id = clue; }
}
```
SUBSTITUTION

- Extract variables declaration from String
  - ‘Hello ${speaker.name} !’

- PropertyResolvers
  - Provides value for expression
  - Composable
  - Support recursion (variables in variable)

```java
DefaultSubstitutionEngine engine = new DefaultSubstitutionEngine();
ChainedResolver chain = new ChainedResolver();
chain.getResolvers().add(new SpeakerResolver());
chain.getResolvers().add(new DateResolver());
engine.setResolver(chain);
engine.substitute("Hello ${speaker.name} ! -- ${date}");

public class SpeakerResolver implements IPropertyResolver {
    public String resolve(String expression) {
        return "speaker.name".equals(expression) ? "Guillaume" : null;
    }
}

public class DateResolver implements IPropertyResolver {
    public String resolve(String expression) {
        return "date".equals(expression) ? (new Date()).toString() : null;
    }
}
```
ARCHIVE

• Uniform API for resource consumption

• Supporting (out of the box)
  • Directory
  • Jar
  • OSGi Bundle

• Extensible

```java
public interface IArchive {
    String getName();
    URL getURL() throws ArchiveException;
    URL getResource(String resourceName) throws ArchiveException;
    Iterator<URL> getResources() throws ArchiveException;
    Iterator<URL> getResources(String resourceName) throws ArchiveException;
    Iterator<String> getEntries() throws ArchiveException;
    boolean close();
    IArchiveMetadata getMetadata();
}
```
Around 20 common libraries wrapped as Bundles

- commons-{collections, logging, modeler}, javassist, jaxb2-ri, jgroups, jsch, opencsv, weld, zookeeper, bouncycastle, ...
- Correct exported packages version, verified imports

All other modules are OSGi Bundles

- Versioned API, content exported
Contributions

What about you?
ANONYMOUS QUOTES

‘I want to code, where are the repositories?’
http://gitorious.ow2.org/ow2-utilities

‘You convinced me, let me write some documentation’
http://utilities.ow2.org

‘Wow, they even have a Bamboo for CI’
http://bamboo.ow2.org/browse/UTIL

‘Do they really need a JIRA, it’s working so great!’
http://jira.ow2.org/browse/UTIL
Questions
RESOURCES

• http://www.flickr.com/photos/emagic/56206868/