



Peter Kriens | OSGi Evangelist/Director

OSGi R4.3 //

Next Release Overview

Agenda

- Framework & ServiceTracker update to Java 5 generics
- A replacement for Package Admin and Start Level
- A Shell standard
- Managing System.out and System.in per thread
- Formatting and Conversion
- Coordination Service
- Resolver Hook
- Generic Capabilities and Requirements
- Configuration Admin Overhaul
- An easier way to receive configuration updates
- Bytecode weaving
- Minor parts

Framework & ServiceTracker update to Java 5 generics

- Current API feels “old-fashioned”
 - Lack of generics
 - Uses of arrays and null return
- Generics have been held back because the embedded world is still still on Java 1.4

JSR 14!

- Developed to run Java 5 code on 1.4 VMs
- All compilers recognize the
 - source 5 –target jsr14 switches
- Output class format 48 (1.4)
 - stores generics in skippable attributes
- Works like a charm
 - No annotations
 - No enums

Generic Services

```
ServiceReference<LogService> lr =  
    context.getServiceReference( LogService.class );  
LogService log = context.getService(lr);
```

```
Collection<EventAdmin> rs =  
    context.getServiceReference(EventAdmin.class);
```

```
ServiceRegistration<ConfigurationListener> sr =  
    context.registerService(ConfigurationListener.class)
```

```
ServiceTracker<EventListener> st =  
    new ServiceTracker<EventListener>( context,  
        EventListener.class, null);
```

Adapt Method

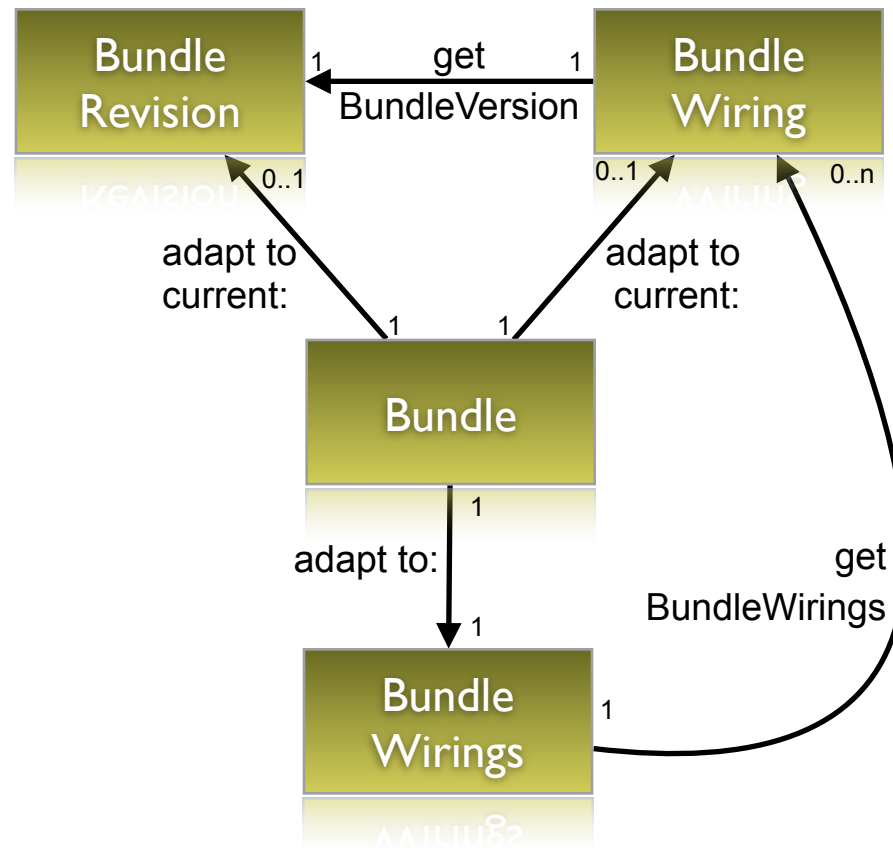
- Package Admin and Start Level were cumbersome to use
 - Bundle was almost always parameter
 - Not very OO
- The adapt method converts an object another type (if supported)

```
public <T> T adapt( Class<T> clazz );
```

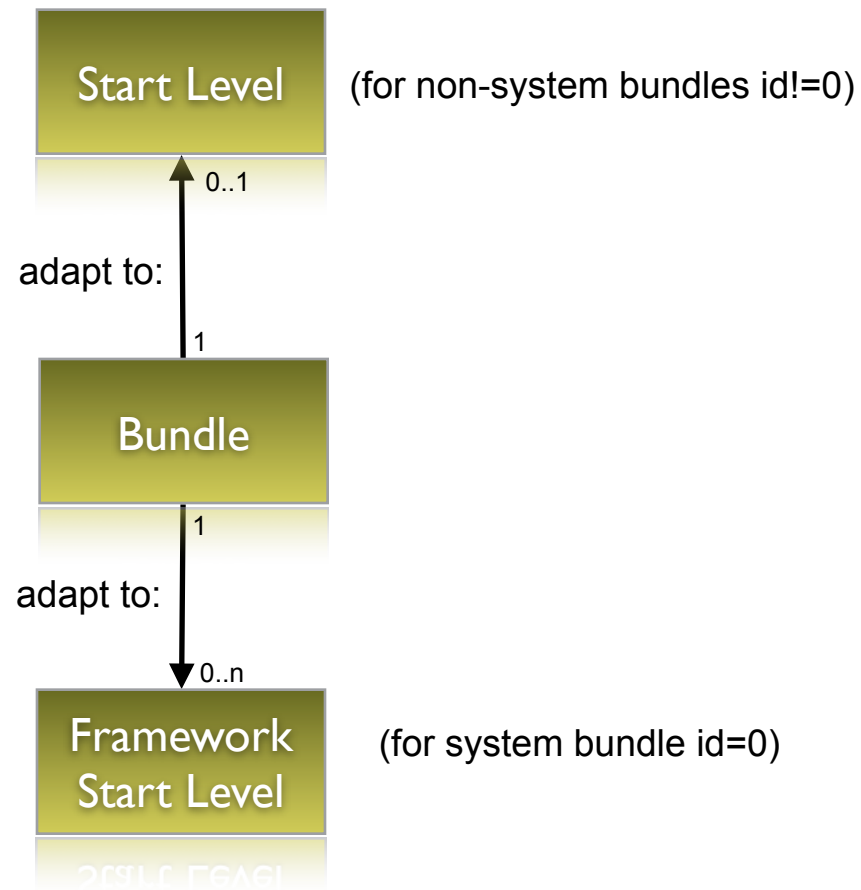
A Replacement for Package Admin and Start Level Service

- Package Admin and Start Level now use the `adapt()` method.
- Package Admin translated to `BundleWiring`, `BundleRevision`, and `BundleWirings`
- Start Level Service translated to `Start Level` and `Framework Start Level`

Bundle Wiring



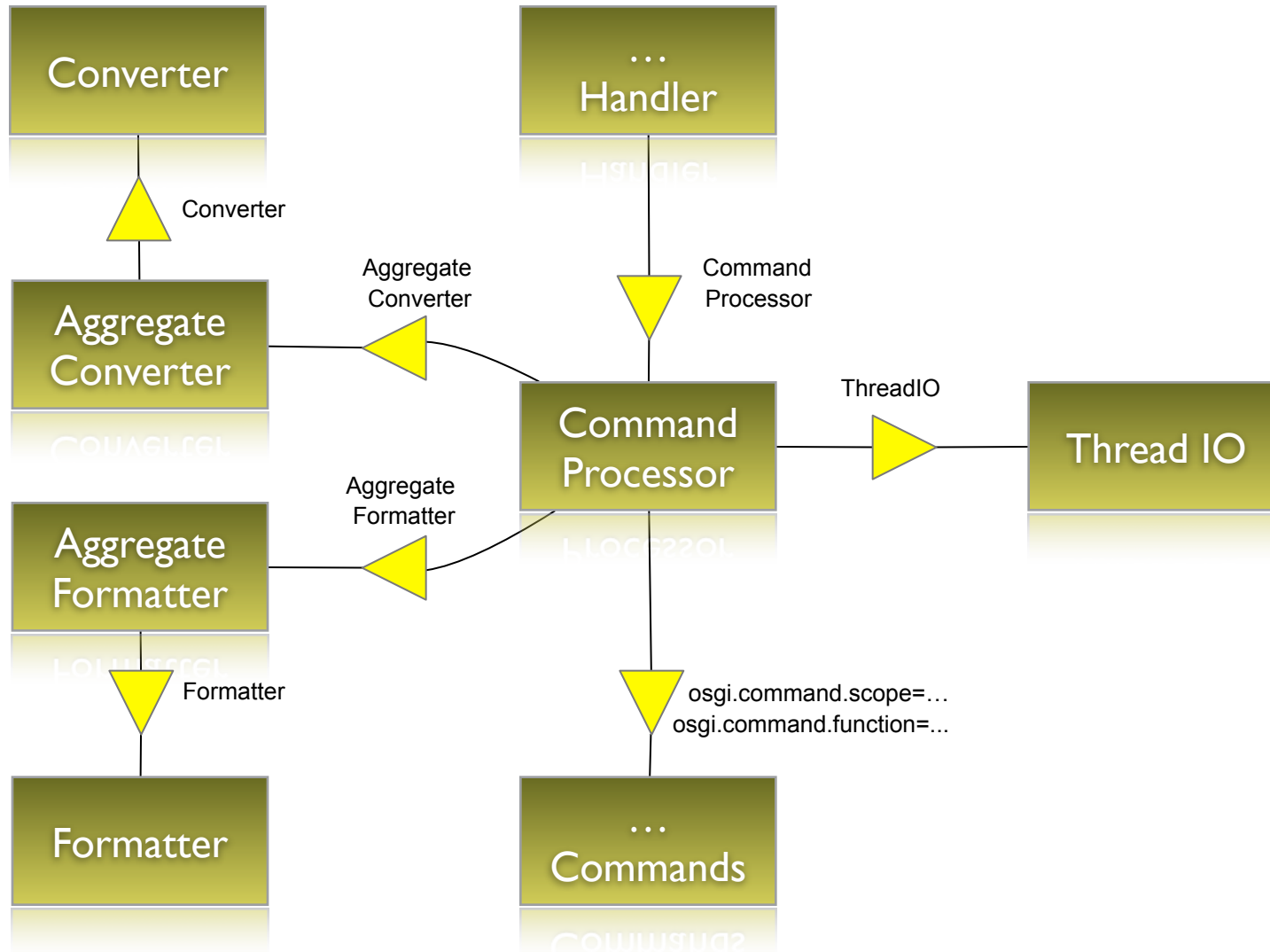
Start Level



Shell

- Uses reflection to provide the commands
 - Aligns commands with the specified API
- Optimizes for typing
- Has closures
- Methods are commands
- Very easy to provide commands
- Terminal abstraction

A Shell



An OSGi Shell

```
$ bundles | grep eclipse
000000 ACT org.eclipse.osgi-3.3.1.R33x_v20070828
000001 ACT org.eclipse.osgi.util-3.1.200.v20070605
...
000002 ACT org.eclipse.osgi.services-3.1.200.v20070605

$ <bundle 6> headers
Tool                Bnd-0.0.169
...
Bundle-SymbolicName biz.aQute.fileinstall
$ <<bundle 6> headers> get Tool
Bnd-0.0.169
$ fi = bundle 6
RegisteredServices  [ManagedServiceFactory]
...
StateChanging       null

$ $fi stop
$ $fi location
initial@reference:file:biz.aQute.fileinstall_1.3.jar/
```

Managing System.out, System.err, and System.in per thread

- System.out, System.in, and System.err are singletons
- The ThreadIO service turns them into thread specific streams
- The shell allows commands to output to System.out

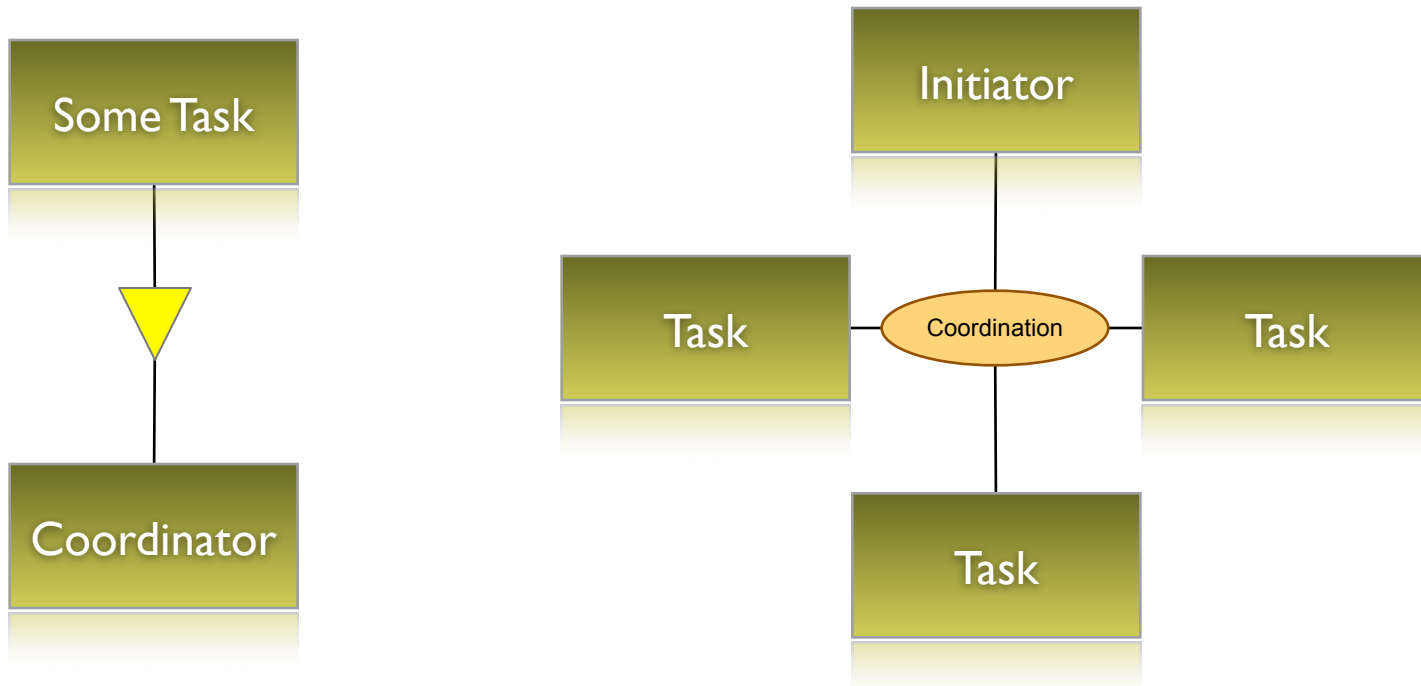
Formatting and Conversion

- Formatter formats an object to a string
 - PART, LINE, INSPECT
- Converter converts an object to another type
- Uses whiteboard pattern
- Aggregates provided to simplify normal usage and simple cases

Coordination Service

- Allow multiple parties to collaborate by signalling the end of a work item
 - ended(Coordination)
 - failed(Coordination)
- Thread based or specifically passed as parameter
- Nested

Coordination Service



Coordination Service

```
Coordination c = coordinator.begin();  
try {  
    ... do work  
  
    c.end();  
} catch( FailedCoordinationfinally {  
  
}
```

Resolver Hook

- Allows interception of resolver
- Deny certain dependencies
 - Provide context
- Replaces nested frameworks
 - Went through nested frameworks, virtual bundles, and now hooks
- Allows scoping
 - Needed for subsystems

Generic Capabilities and Requirements

- Package-Import, Require-Bundle, and Bundle-Host are requirements
- Bundle-SymbolicName and Package-Export are capabilities
- Requirements are filter expressions
- Capabilities are sets of properties
- Pioneered in OBR
- Screen size, certificate, etc.

Configuration Admin Overhaul

- Security
 - Segmentation
- Allow a service to register under multiple pids
 - Use aspects: web config, system config
- Coordinated Update
- Version count on Configuration object
- Extension PIDs
 - Target a single bundle
 - `com.example.WebConf-com.bundle.foo-1.2.3-123`

An Easier Way to Receive Configuration Updates

- Configuration Admin uses properties
 - Hard to use in Java
- Would be nice to use Java type ...
- Schema of properties defined in a Java Interface
 - PID is classname
 - Interfaces can extend other interfaces


An Easier Way to Receive Configuration Updates

- Works for Managed Services and Managed Service Factories
- Data Validation a la Meta Type using annotations
 - mandatory/default
 - options
 - size
- Implemented in Apache Cade already (Karaf)...

An Easier Way to Receive Configuration Updates

```
public interface WebConf {  
    String host();  
    @Default(80) int port();  
}
```

PID



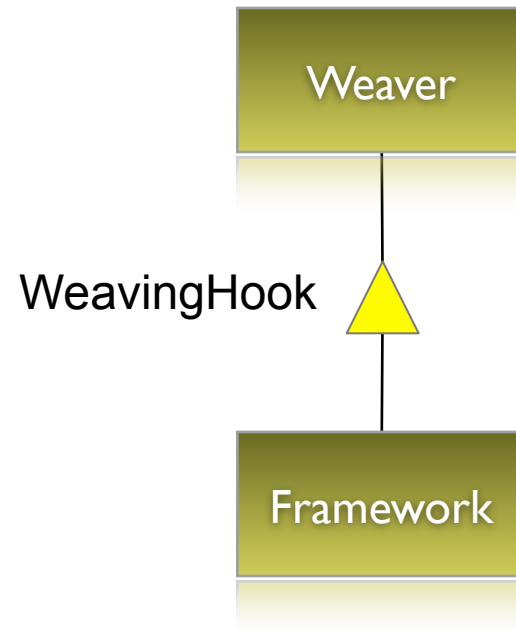
```
public class X implements Configurable<WebConf> {  
    public void updated( WebConf web ) {  
        setPort( web.port() );  
    }  
}
```

```
public interface MyConf extends WebConf, SysConf { }
```

Bytecode weaving

- Bytecode weaving very popular in enterprise applications
- Needs to changes the bytes loaded by the class loader
- Introduces the service:
`org.osgi.framework.weaving.WeavingHook`
- Ordering defined by ranking
- Can add additional requirements

Byte Code Weaving



Minor Updates

- Event Admin
 - Reusing properties
- Classpath scanning
 - Following the bundle class path

Q & A