Developing Java Applications with OSGi

Emerging Technologies for the Enterprise 2008

Michael P. Redlich

March 26, 2008

My Background (1)

- Degree
  - B.S. in Computer Science
  - Rutgers University (go Scarlet Knights!)
- ExxonMobil Research & Engineering
  - Senior Research Technician (1988-1998, 2004-present)
  - Systems Analyst (1998-2002)
- Ai-Logix, Inc.
  - Technical Support Engineer (2003-2004)
- Amateur Computer Group of New Jersey (ACGNJ)
  - Java Users Group Leader (2001-present)
  - President (2007-present)
  - Secretary (2006)

My Background (2)

- Publications
  - Java Boutique (http://www.javaboutique.com/)
    - Co-authored with Barry Burd
    - Design Patterns
  - http://publications.redlich.net/
- Presentations
  - Trenton Computer Festival (TCF) since 1998
  - TCF IT Professional Seminars since 2006
  - Princeton Java Users Group
  - Capital District Java Developers Network
  - New York Software Industry Association (NYSIA)

Other OSGi-Related Seminars at ETE 2008

- OSGi with Spring DM
  - Dmitry Sklyut
  - Wednesday, March 26
  - 5:15 – 6:15pm
  - Behrakis Grand Hall (South)
- 2010: An Acronym Odyssey
  - Brian O'Neill, Technical Architect, Gestalt LLC
  - Wednesday, March 26
  - 5:15 – 6:15pm
  - MacAlister 4011

Objectives

- What is OSGi?
  - OSGi Alliance
  - OSGi Frameworks
  - OSGi Layers
  - How to interact with the framework
- Develop a first OSGi bundle
- Develop a more “Real-World” application
- Source code, source code, source code (*yea!*)

What is OSGi?

- Open Services Gateway initiative (OSGi)
- Originally designed to promote dynamic systems for embedded Java and network devices
- Foundation for an enhanced service-oriented architecture
  - Defines a standardized, component-oriented environment
  - Applications can be started, stopped, updated, and uninstalled without requiring the JVM to be restarted

OSGi to the Rescue!

- Component issues
  - Installation
  - Start and stop
  - Manage multiple versions
  - Update
  - Uninstall
- JAR file dependency issues
  - **`Class-Path`** attribute in MANIFEST file
  - Versioning in file name

OSGi Alliance

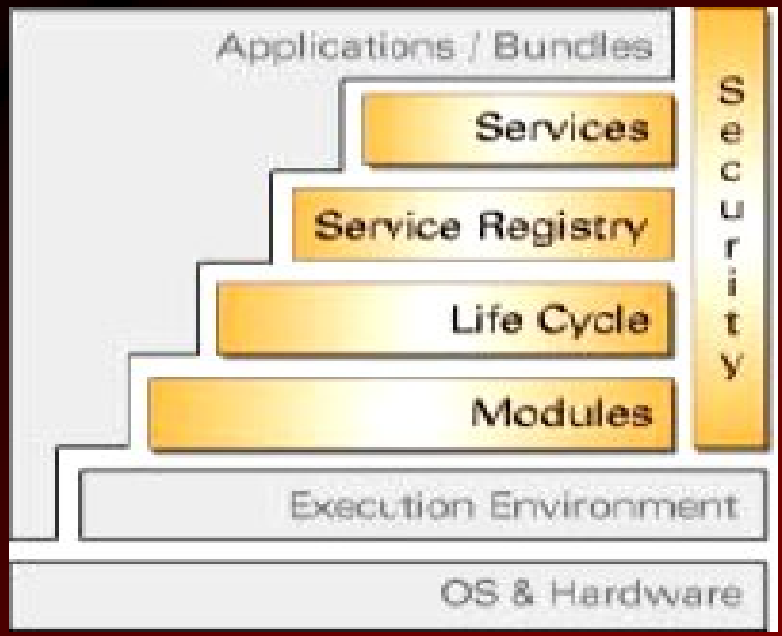
- A non-profit organization formed in March 1999
- "…a worldwide consortium of technology innovators that advances a proven and mature process to assure interoperability of applications and services based on its component integration platform."
- "…provides specifications, reference implementations, test suites and certification to foster a valuable cross-industry ecosystem."

OSGi Frameworks

- Eclipse Equinox (3.3.2)
- Apache Felix (1.0.3)
- Knopflerfish (2.0.5)
- Spring OSGi Modules (1.0.1)
- Newton (0.2)

OSGi Framework Layers

- Modules
  - Class loading policies
- Life Cycle
  - Bundle management
- Service Registry
  - Communication among bundles
  - Service discovery
- Security Layer
  - Spans all other layers

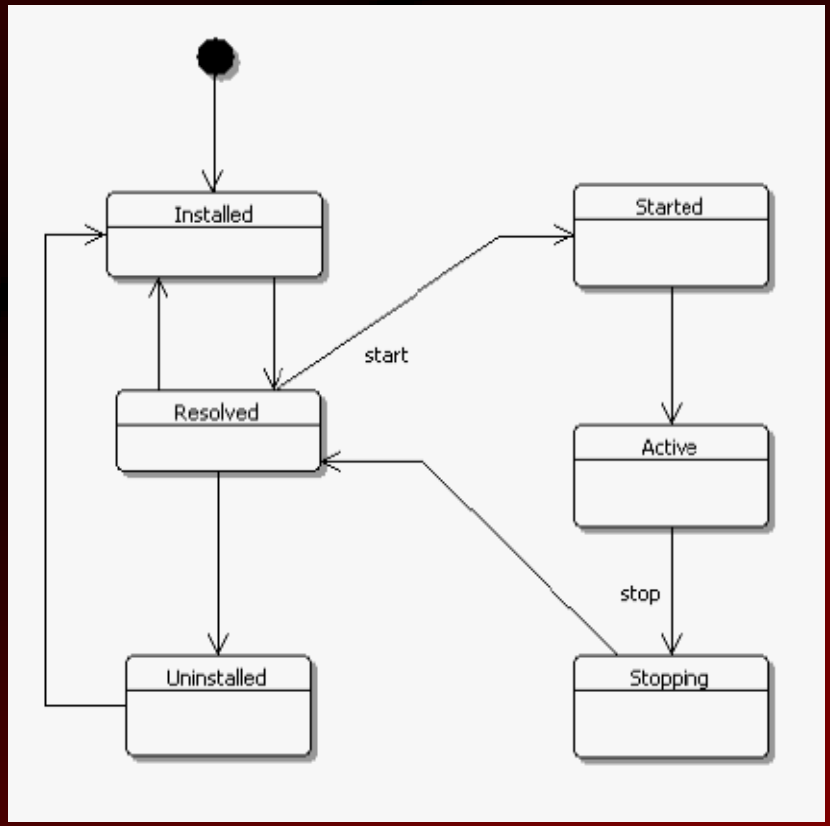


Modules Layer

- Defines class loading policies
- Every bundle can export and import packages
  - A package is always exported with a unique version
  - A bundle can specify a range of versions that it can import
- Supports multiple class spaces where multiple versions of the same class can be used simultaneously
- Ensures that bundles can bind to a new exporting package should the original one be uninstalled

Life Cycle Layer

- Defines how bundles are installed, updated, and uninstalled
  - **UNINSTALLED**
  - **INSTALLED**
    - Framework has the bundle
    - Dependencies not resolved
  - **RESOLVED**
    - Dependencies resolved
    - Bundle can be started
  - **STARTING**
  - **STOPPING**
  - **ACTIVE**
    - Bundle is running



Basic APIs to Interact with the Framework

- **`Bundle`**
- **`BundleActivator`**
- **`BundleContext`**
- **`ServiceRegistration`**
- **`ServiceTracker`**
- **`ServiceReference`**

Bundles

- A “module” or “plug-in”
- Packaged in a JAR file
  - Java CLASS files and other resources
  - Special MANIFEST file
- Started through a **`BundleActivator`**
- Each of the frameworks support special bundles
  - e.g., system bundle that represents the framework

```
Manifest-Version: 1.0
Bundle-Name: Basic Movie Finder
Bundle-Activator:
 org.emergingtech.BasicMovieFinderActivator
Bundle-SymbolicName: org.emergingtech.basicmoviefinder
Bundle-Version: 1.0.0
Import-Package: org.emergingtech;version="[1.0.0,2.0.0)"
Export-Package: org.emergingtech;version="1.0.0"
```

Specifies the name of the class used to start and stop the bundle

Defines a human-readable name (that may contain spaces) for this bundle

Declares imported (including version ranges) and exported packages for this bundle

Specifies a unique, global name for this bundle

BundleActivator

- An interface that may be implemented to start and stop a bundle
- Created by the framework as specified through the **`Bundle-Activator`** entry in the manifest file
  - Fully-qualified Java class name

```
void start(BundleContext context);
void stop(BundleContext context);
```

BundleContext

- A bundle's execution context within the framework
  - A “magic ticket” maintained by the framework
- Created by the framework when a bundle is started
- **`BundleContext`** allows a bundle to:
  - Install new bundles
  - Interrogate other bundles
  - Register services with the Service Registry
  - Subscribe to/consume registered services
  - Retrieve a list of **`ServiceReference`**s from the Service Registry

And Now For…

- …our first OSGi example?

Are You Ready?

OSGi Services

- Bundles registered with the Service Registry
  - Services “live” in the Services Registry
- Registered under a Java interface and an optional set of properties
- The framework automatically unregisters all services from a bundle that is stopped
  - Notifies all of its dependents
- Services can be tracked and retrieved

Registering a Service

- To register a service with the framework, use **`BundleContext.registerService()`** method
- Returns a **`ServiceRegistration`** object

```
MovieFinder finder = new BasicMovieFinder();

ServiceRegistration registration =
context.registerService(MovieFinder.class.getName(),finder
,properties);
```

Register the specified service, **`finder`**, with an optional set of specified properties under the specified class name, **`MovieFinder`**

Consuming a Service

- An separate bundle that is interested in using a registered service
- To discover and retrieve services, use **`ServiceTracker`**

```
ServiceTracker tracker = new
ServiceTracker(context,MovieFinder.class.getName(),null);

tracker.open();

MovieFinder finder = (MovieFinder)tracker.getService();
```

Creates a **`ServiceTracker`** object for the specified class name, i.e., the **`MovieFinder`** service

Returns a **`MovieFinder`** service being tracked by ServiceTracker

Start tracking instances of **`MovieFinder`**, i.e., the service, registered in the Service Registry

ServiceRegistration

- A registered service
- For the private use of the registering bundle
  - Should not be shared with other bundles
  - Object returned upon a successful call to **`BundleContext.registerService()`**
- Used to update the properties of the service or to unregister the service
- Every service registered in the framework has a unique **`ServiceRegistration`** object

ServiceTracker

- Simplifies using services from the framework's service registry
  - Abstracts away all the gory details of dealing with the Service Registry
- **`BundleContext`** can also retrieve services, but requires additional maintenance

ServiceReference

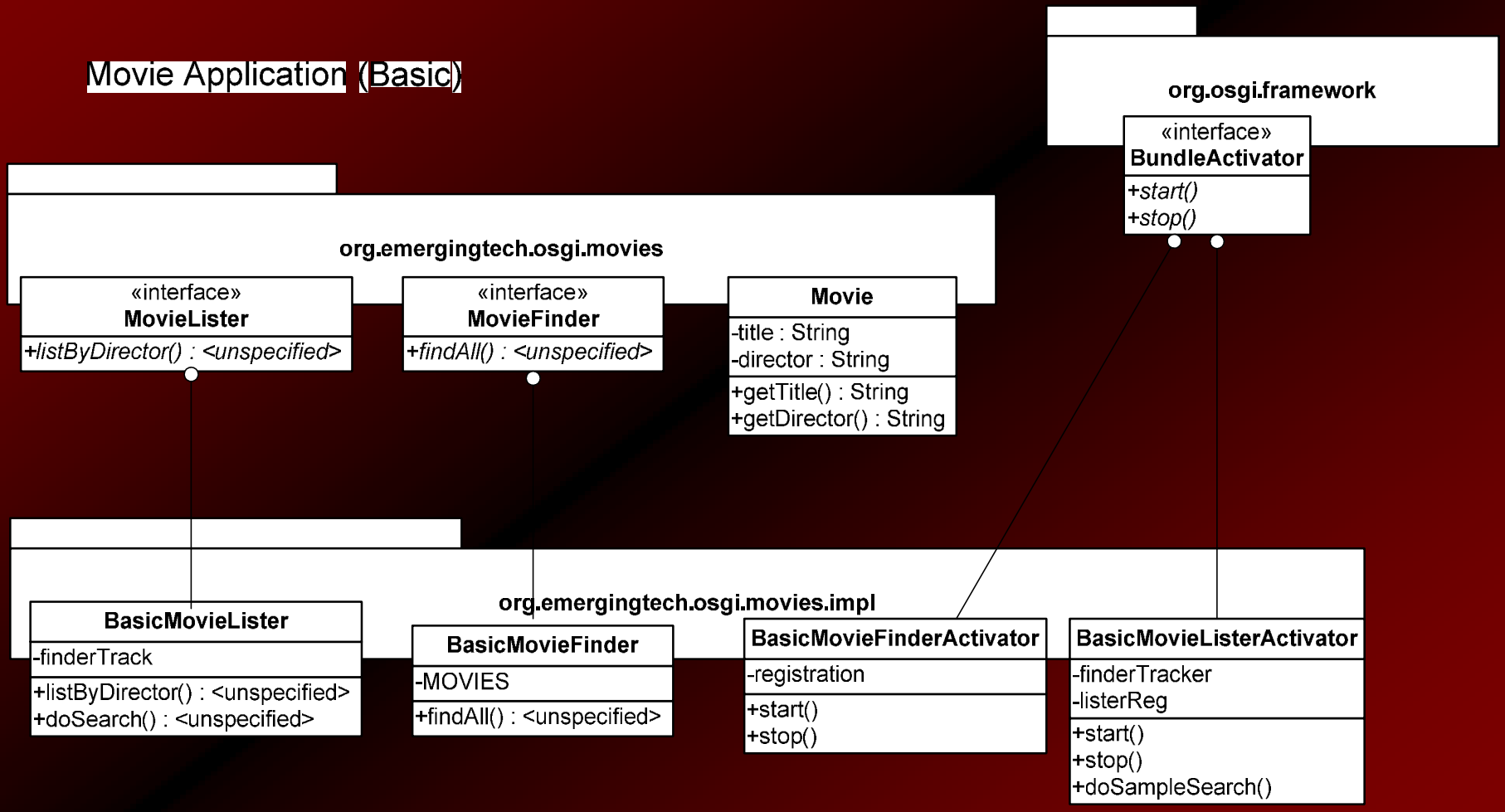
- A reference to a service
- A lightweight handle
- Provides access to a service's properties but not the actual service object
  - The service object must be acquired through a bundle's **`BundleContext`**
  - Can be queried by a bundle to assist in the selection of a service
- Avoids creating unnecessary dynamic service dependencies among bundles

And Now For…

- …a more "real-world" example (*woo-hoo!*)

Are You Ready?

MovieLister Application



OSGi Resources (1)

- Equinox
  - http://www.eclipse.org/equinox/
- Apache Felix
  - http://felix.apache.org/
- Knopflerfish
  - http://www.knopflerfish.org/
- Spring OSGi
  - http://www.springframework.org/osgi/
- Newton
  - http://newton.codecauldron.org/

OSGi Resources (2)

- OSGi Alliance
  - http://www.osgi.org/
- OSGi Technical Whitepaper
  - http://www.osgi.org/wiki/uploads/Links/OSGiTechnicalWhitePaper.pdf
- OSGi Best Practices Presentation
  - http://www.osgi.org/wiki/uploads/Conference/OSGiBestPractices.pdf
- Getting Started with OSGi
  - Neil Bartlett's Point-Free Blog
  - http://www.neilbartlett.name/blog/osgi-articles/

Java Resources

- ACGNJ Java Users Group
  - facilitated by Mike Redlich
  - http://www.javasig.org/
- Princeton Java Users Group
  - facilitated by Yakov Fain
  - http://www.myflex.org/princetonjug/
- NYJavaSIG
  - facilitated by Frank Greco
  - http://www.javasig.com/
- Chariot Solutions
  - http://www.chariotsolutions.com/