



EUROPE'S LEADING AEM DEVELOPER CONFERENCE
27th – 29th SEPTEMBER 2021

OSGi R8, Felix 7, Atomos and the future of OSGi@Eclipse

Karl Pauls, David Bosschaert – Adobe

Who we are



- David Bosschaert
 - Senior Computer Scientist @ Adobe
 - Member of the Apache Software Foundation
 - Apache Sling, Felix & Aries PMC
 - OSGi Working Group @ Eclipse Member



- Karl Pauls
 - Senior Computer Scientist @ Adobe
 - Member of the Apache Software Foundation
 - Apache Sling & Felix PMC
 - OSGi Working Group @ Eclipse Member

Agenda

- OSGi at Eclipse
- OSGi R8
 - OSGi R8 core
 - OSGi R8 compendium
- OSGi Connect
 - Felix Atomos
- Felix 7
- Demo



OSGi R8

and the future of OSGi@Eclipse



OSGi at Eclipse



- OSGi is now a project at Eclipse
- Fully Open Source
- Compatible implementations at Eclipse/Apache or elsewhere
- Just join in and contribute!
- Info here:

<https://projects.eclipse.org/projects/technology.osgi>

- OSGi R8 = Core R8 + Compendium R8
- Core R8 Released late 2020
- New specs:
 - 60 – Connect Specification
 - 59 – Condition Service Specification

- When is your system fully ready?
- Generally very app-specific
- Sometimes multiple levels
- Introducing:
`org.osgi.service.condition.Condition`
service interface

- A client can just listen for a condition:

```
@Reference(  
    target="(osgi.condition.id=mycondition)")  
Condition ready
```


Compendium R8 Specs coming up

New

- 153 Service Layer for OneM2M (impl at [Eclipse](#))
- 157 Typed Event Service (impl at [Apache Aries](#))
- 159 Feature Service (impl at [Apache Felix](#))

- Send asynchronous Type-safe Events
- Locally in JVM
- Events are OSGi DTOs
 - Primitive types / wrappers
 - Strings
 - Collections / arrays
 - DTOs

```
public class ExampleEvent {  
    public String message;  
    public boolean public;  
}
```

Sending:

```
@Reference  
TypedEventBus bus;  
  
public void sendEvent() {  
    ExampleEvent event = new ExampleEvent();  
    event.message = "Hello there";  
  
    bus.deliver("org/osgi/example/ExampleEvent", event);  
}
```

Receiving:

```
@Component  
public class ExampleTypedConsumer implements  
TypedEventHandler<ExampleEvent> {  
    @Override  
    public void notify(String topic, ExampleEvent event) {  
        System.out.println("Received event: "  
            + event.message);  
    }  
}
```

- Design and work with entities
 - Larger than individual bundles
- Configuration
- Additional metadata
- A design artifact
 - can be mapped to runtime implementations (Sling/Karaf/Eclipse etc...)



- Specifies JSON model
- ... and API
- Building block
 - Authoring
 - Tooling
 - Launcher

```
{  
  "id": "org.acme:acmeapp:1.0.1",  
  
  "name": "The Acme Application",  
  "complete": true,  
  
  "bundles": [  
    { "id": "org.osgi:org.osgi.util.function:1.1.0" },  
    { "id": "org.osgi:org.osgi.util.promise:1.1.1" },  
    {  
      "id": "org.apache.commons:commons-email:1.5",  
      "org.acme.javadoc.link":  
        "https://commons.apache.org/.../javadocs/api-1.5"  
    },  
    { "id": "com.acme:acmelib:1.7.2" }  
  ]  
}
```



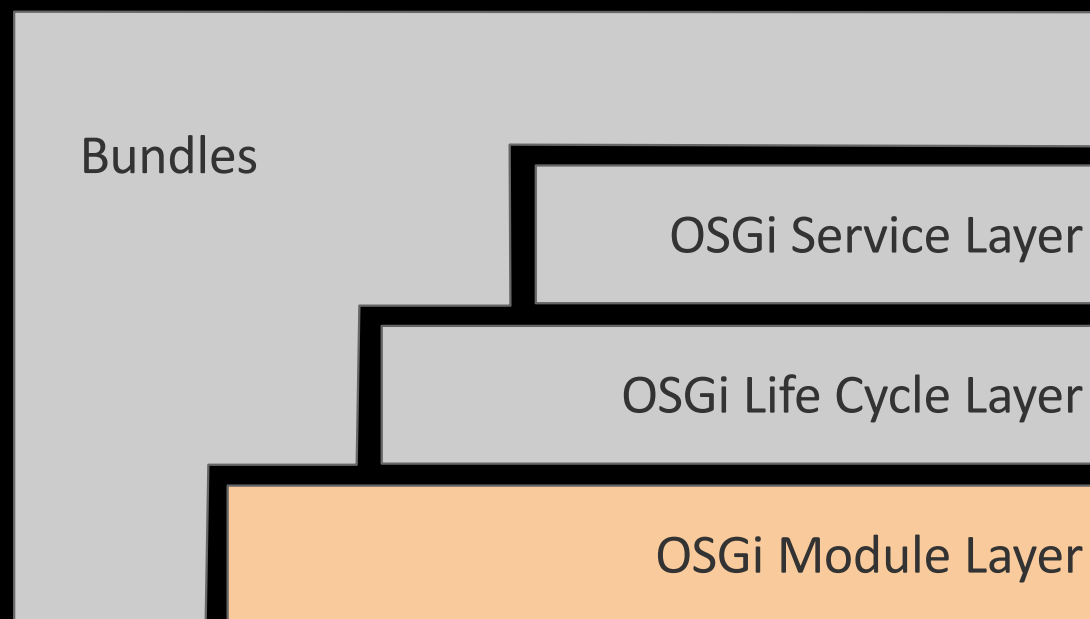
Compendium R8 Specs coming up

- Final specs aimed for October 2021
- Current drafts at <https://osgi.github.io/osgi/cmpn/>

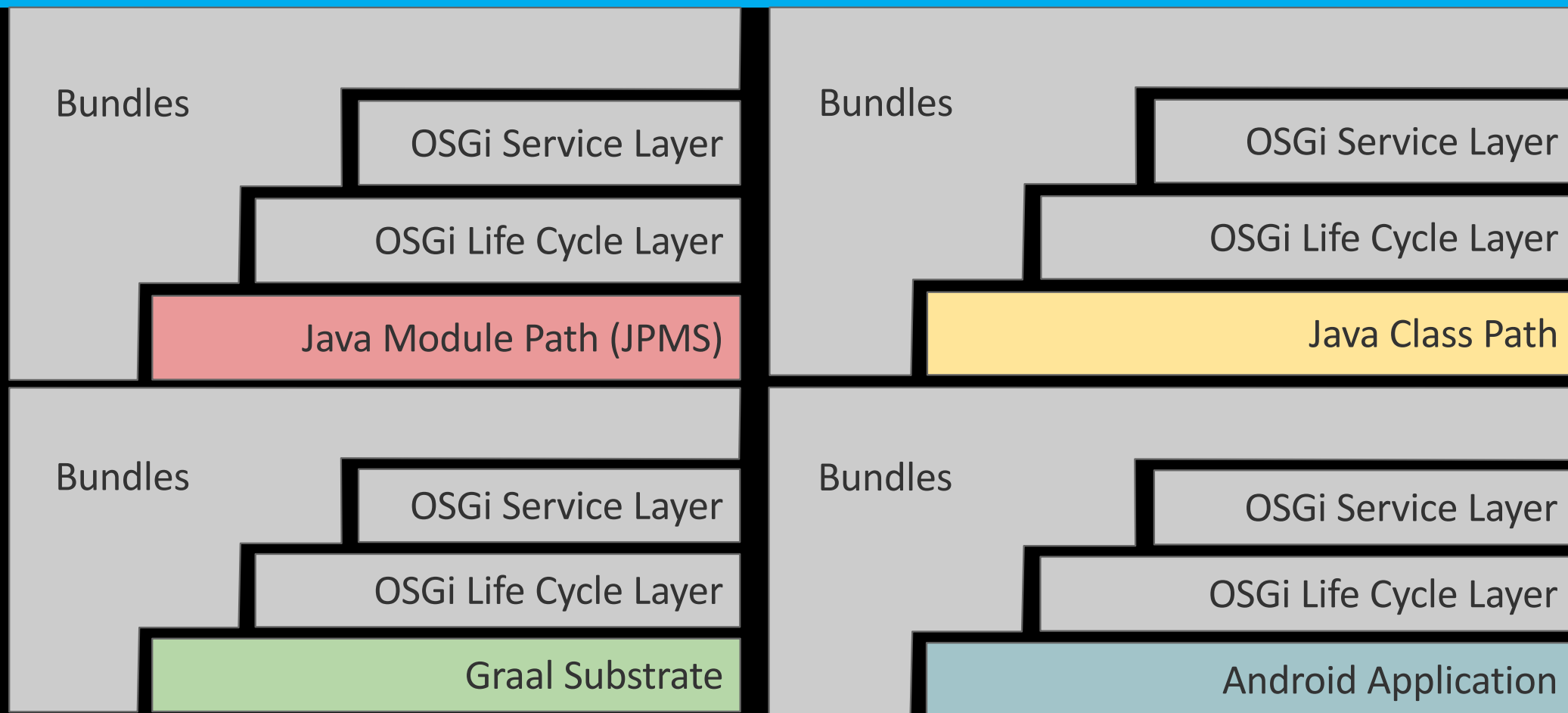
OSGi Connect

- How can we enable use of OSGi technology in more environments?
 - Integrate with JPMS
 - Native Compilation
 - and more...
- By allowing bundles whose content is not managed by the framework!
 - OSGi Connect
 - Enable content managed outside the Framework to be connected to Bundles installed in the Framework

- Module Layer controls class loading
- Life Cycle provides entry point to code through activation
- Service Layer provides powerful programming model for developing components



OSGi Connect





Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```



Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```

Mandatory unique location to bundle, may be in the form of a URL



Framework Managed Bundle Content

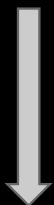
```
installBundle(String location, InputStream content)
```

Optional content to read the bundle content from



Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```



Running Framework



Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```

Running Framework

If content is available:
framework persists content to
storage



Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```

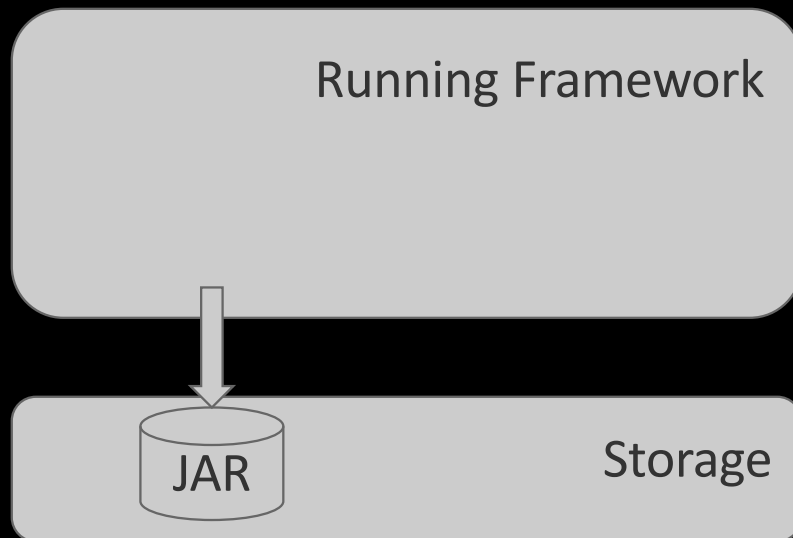
Otherwise: location string is
used to determine content

Running Framework



Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```

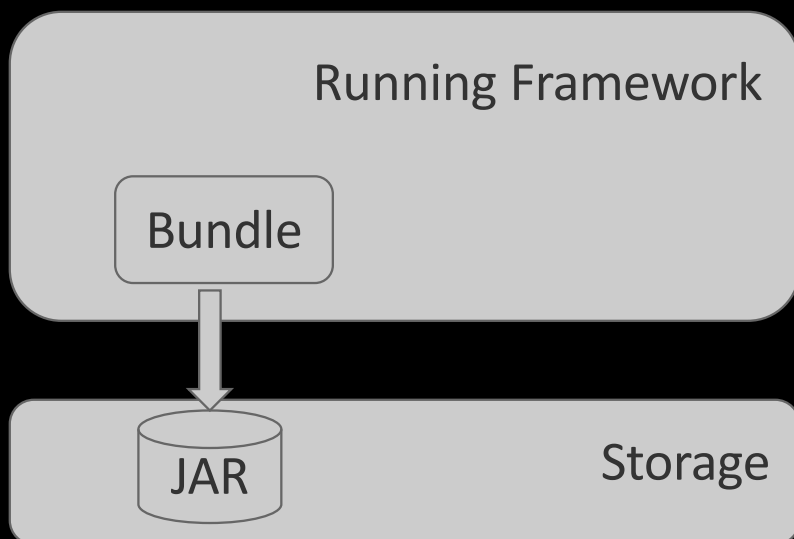


Persist Bundle JAR to
Framework Storage



Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```

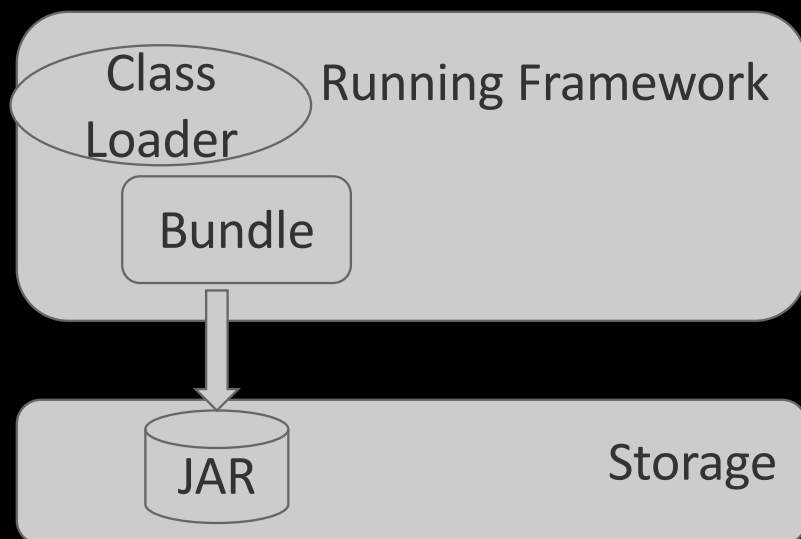


Read bundle manifest; create
Bundle object INSTALLED in the
Framework



Framework Managed Bundle Content

```
installBundle(String location, InputStream content)
```



RESOLVED



Framework Managed Bundle Content

```
ConnectFrameworkFactory.newFramework( Map<String,String> configuration,  
                                     ModuleConnector moduleConnector )
```

Framework
Impl

Module
Connector
Impl

Launcher

Module
Bundle A

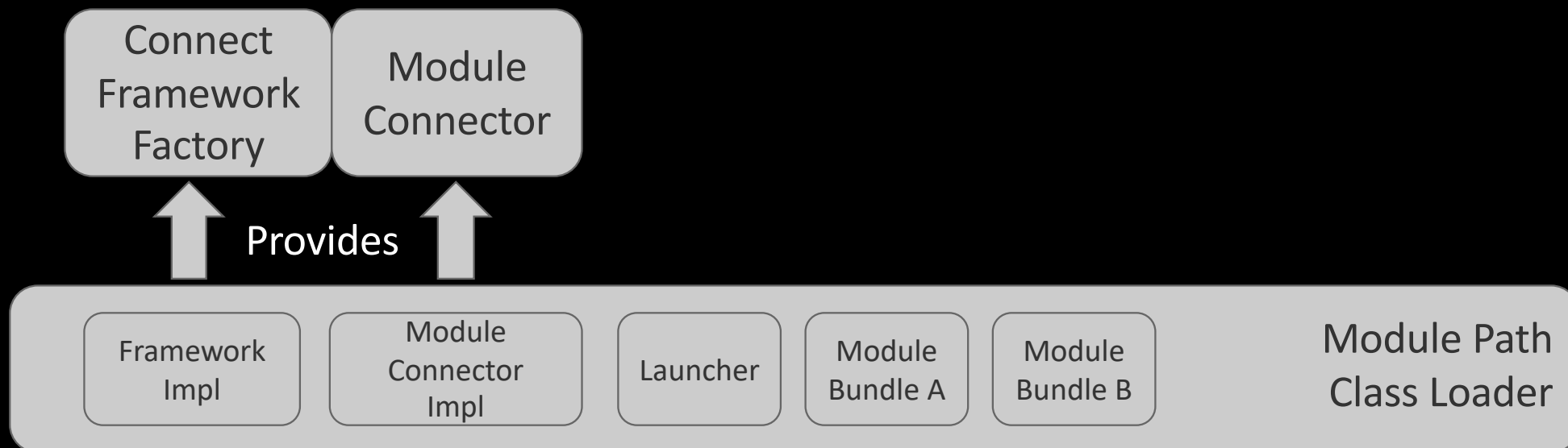
Module
Bundle B

Module Path
Class Loader



Framework Managed Bundle Content

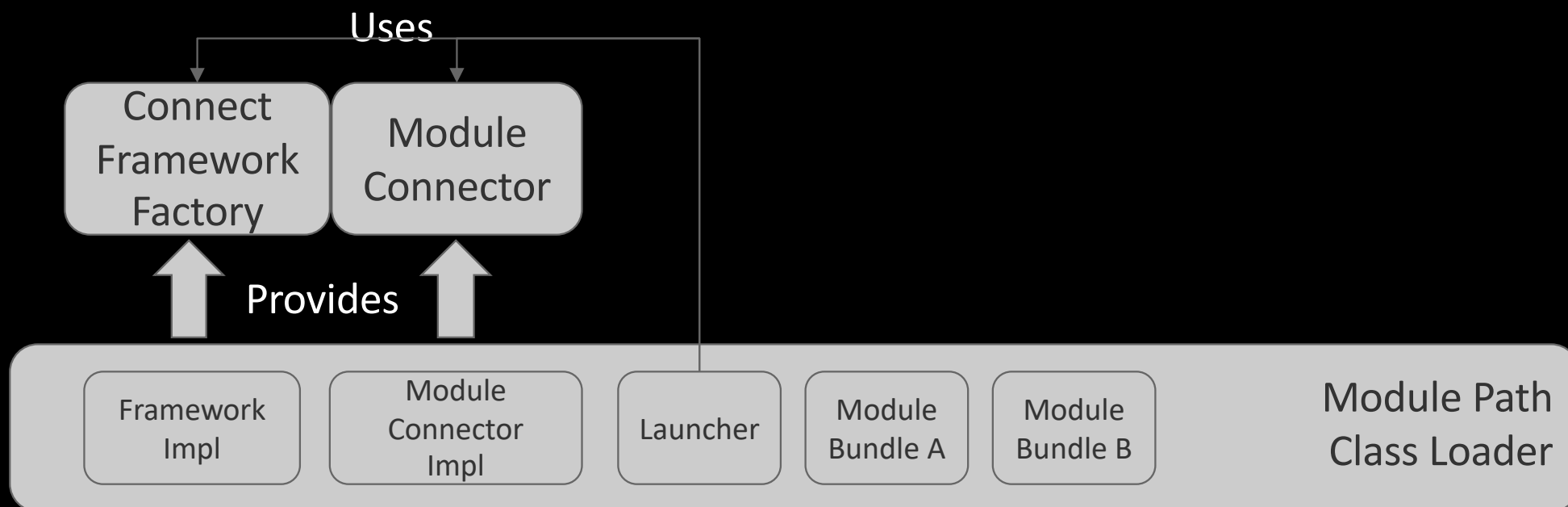
```
ConnectFrameworkFactory.newFramework( Map<String,String> configuration,  
                                     ModuleConnector moduleConnector )
```





Framework Managed Bundle Content

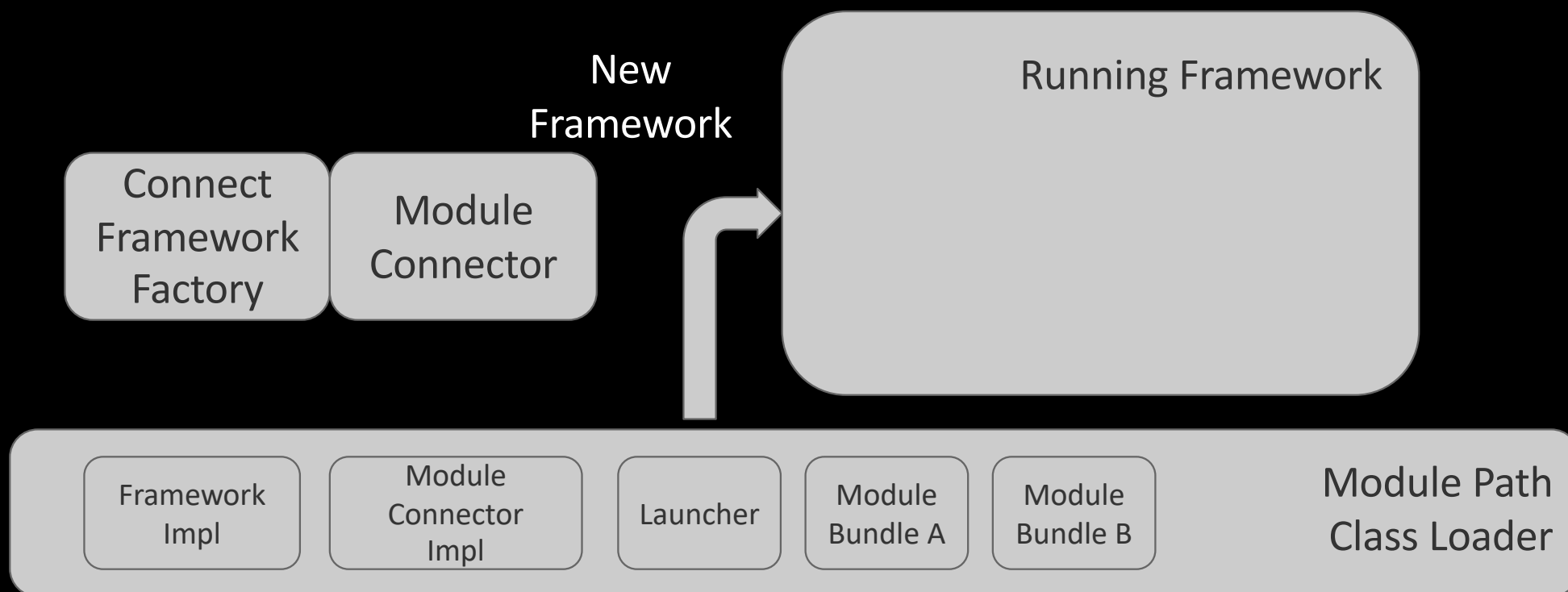
```
ConnectFrameworkFactory.newFramework( Map<String,String> configuration,  
                                     ModuleConnector moduleConnector )
```





Framework Managed Bundle Content

```
ConnectFrameworkFactory.newFramework( Map<String,String> configuration,  
                                     ModuleConnector moduleConnector )
```





Framework Managed Bundle Content

```
installBundle(String loc, InputStream content)
```



loc = "BundleA"

Running Framework

Module
Connector

Framework
Impl

Module
Connector
Impl

Launcher

Module
Bundle A

Module
Bundle B

Module Path
Class Loader

Framework Managed Bundle Content

```
installBundle(String loc, InputStream content)
```

↓
loc = "BundleA"

connect "BundleA"

Module
Connector

Running Framework

Framework
Impl

Module
Connector
Impl

Launcher

Module
Bundle A

Module
Bundle B

Module Path
Class Loader

Framework Managed Bundle Content

```
installBundle(String loc, InputStream content)
```

↓ loc = "BundleA"

INSTALLED

connect "BundleA"

Module
Connector

Bundle A

Running Framework

Connect
Module
BundleA

Framework
Impl

Module
Connector
Impl

Launcher

Module
Bundle A

Module
Bundle B

Module Path
Class Loader



Framework Managed Bundle Content

```
installBundle(String loc, InputStream content)
```

↓
loc = "BundleA"

RESOLVED

Module
Connector

Bundle A

Running Framework

Connect
Module
BundleA

Framework
Impl

Module
Connector
Impl

Launcher

Module
Bundle A

Module
Bundle B

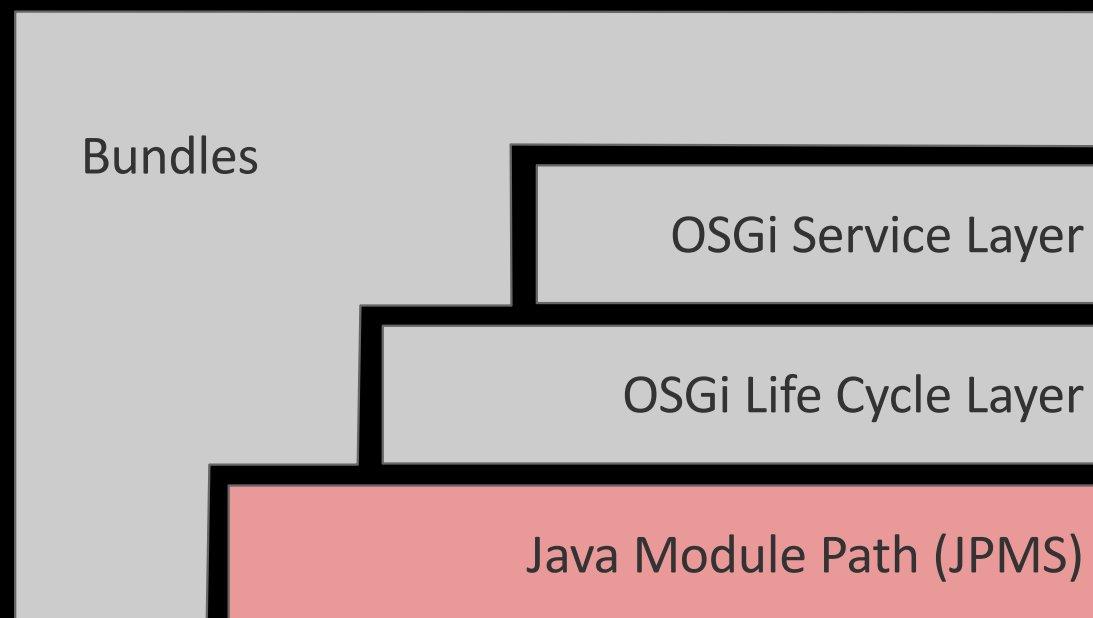
Module Path
Class Loader



Atomos - Apache Felix Project

<https://github.com/apache/felix-atomos>

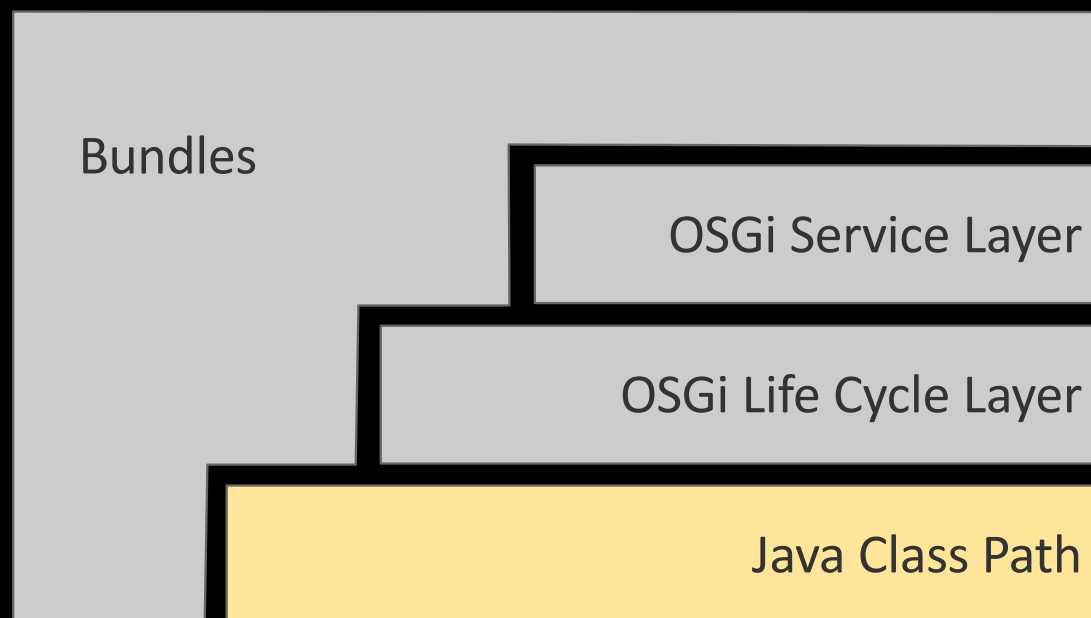
- JPMS controls the class loader
- Modules and Bundles live together in the same layer
- Generation of OSGi meta-data for Modules
- JRE Boot modules are represented by bundles





Atomos - OSGi, On the Class Path

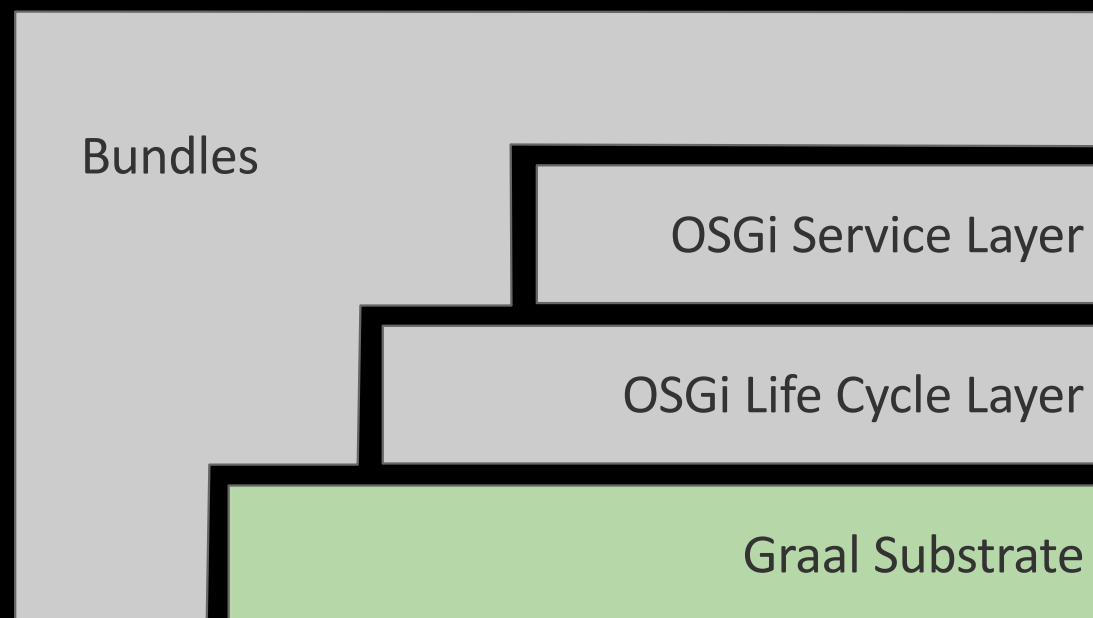
- Java Class Path controls the class loader
- Other JARs and Bundles live together in the same class loader
- No isolation provided at the class loader level
- Java 9+ JRE Boot modules are represented by bundles
- Other URL Class Loader like loaders work (e.g. Spring Boot Loader)





Atomos - OSGi, Native

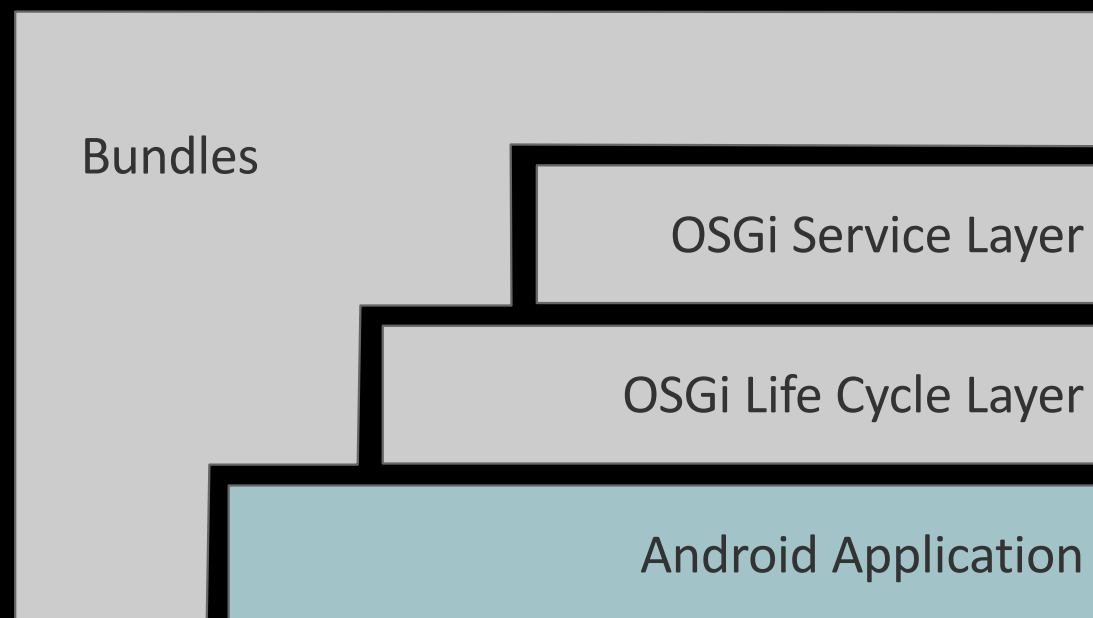
- Substrate native controls “class loading”
- Atomos indexes resources for each bundle
- Build tools available to configure necessary reflection for OSGi





Atomos - OSGi, Android Application

- Android Runtime controls “class loading”
- Atomos indexes resources for each bundle - similar to Substrate
- Build Android Application from a single “uber” JAR that contains all required bundles



Felix 7



Felix 7

- Latest framework release
 - OSGi R8 core compliant
 - OSGi Connect support
 - Works with java 17 (since 7.0.1)
 - Actually is a JPMS module (can run from the module path)
 - No more `--add-opens` required*

Demo



Thank you