ECLIPSE
2020 CON

Ubiquitous OSGi:

Android, Graal Substrate, Java Modules, Flat Class Path

Tom Watson - IBM
Karl Pauls - Adobe

Tom Watson - IBM

- Senior Software Engineer - IBM, Austin, Texas
- Open Source
  - Eclipse/Equinox projects
  - Apache Aries/Felix
  - Open Liberty
- OSGi Alliance

Karl Pauls - Adobe

- Computer Scientist @ Adobe, Basel, Switzerland
- Member of the Apache Software Foundation
- Apache Sling and Apache Felix PMC (VP) member
- Co-Author of OSGi in Action

Ubiquitous OSGi

- 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
  - Atomos
    - Connector implementation

Demos

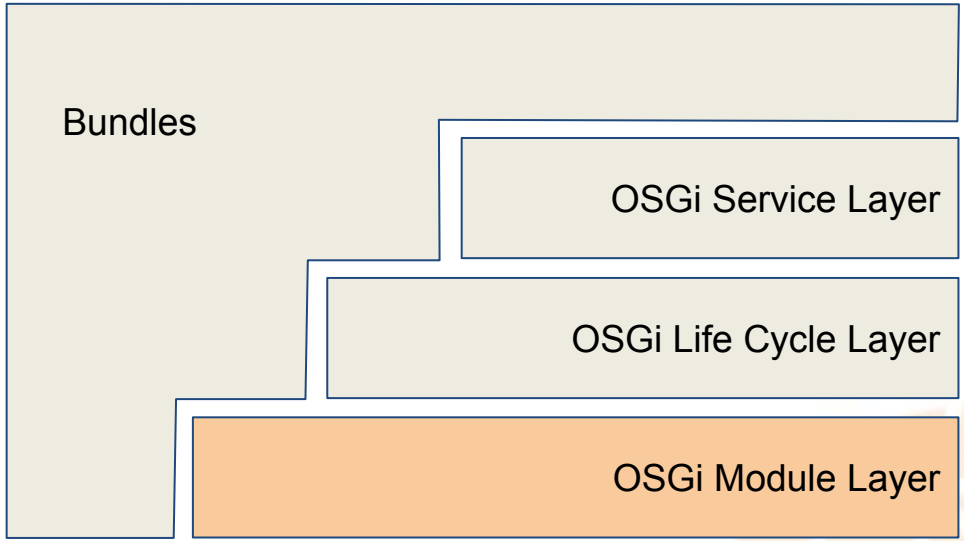
Apache Felix Atomos

Atomos - Apache Felix Project

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

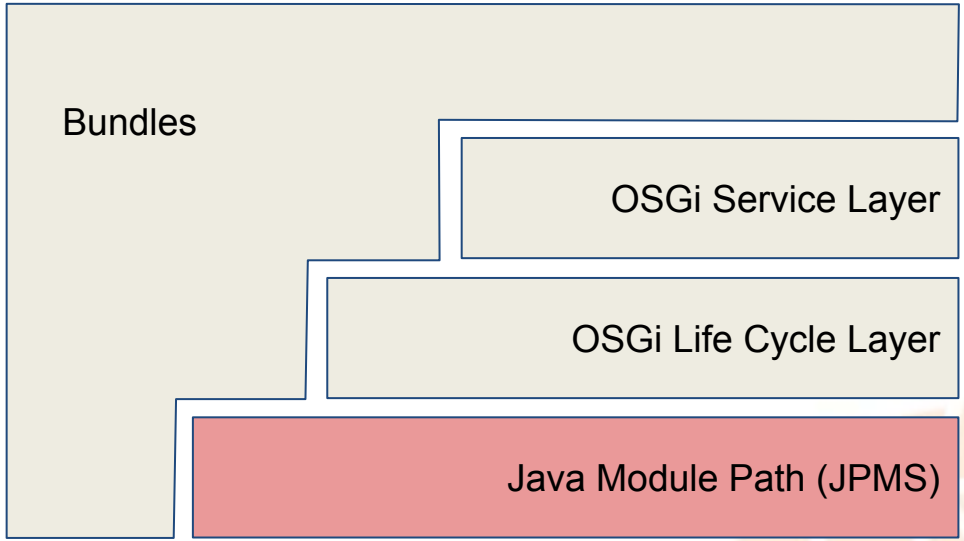
Atomos - OSGi, More Than a Module System

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



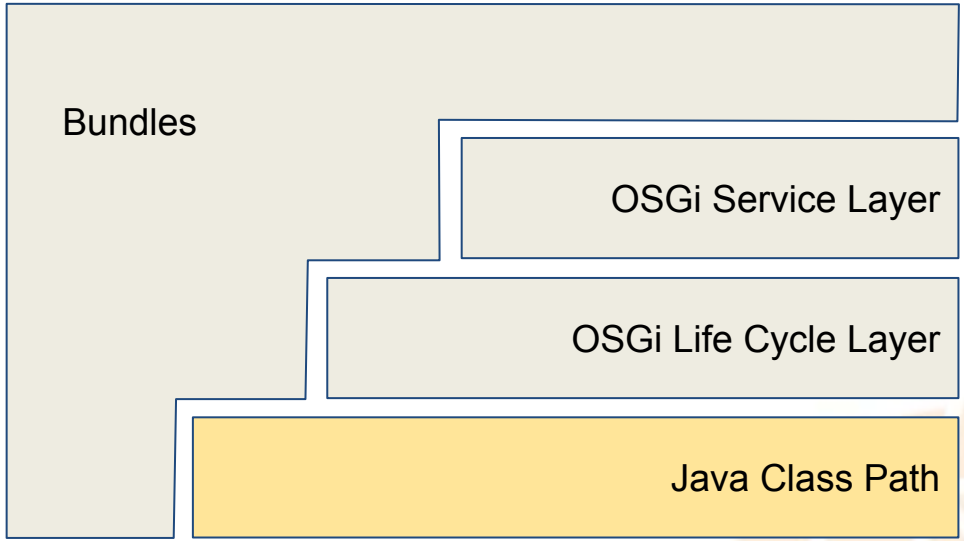
Atomos - OSGi, On JPMS

- 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



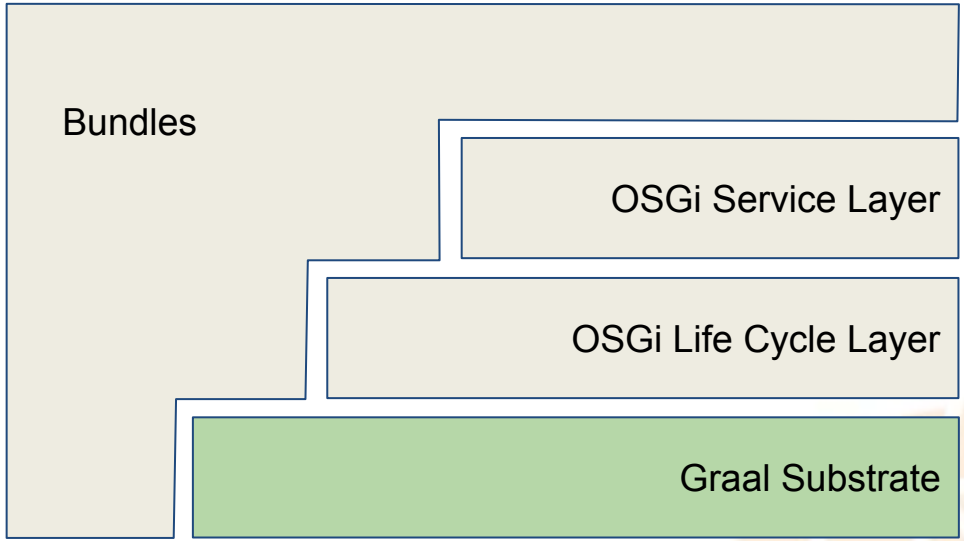
Atomos - OSGi, On the Class Path

- 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)



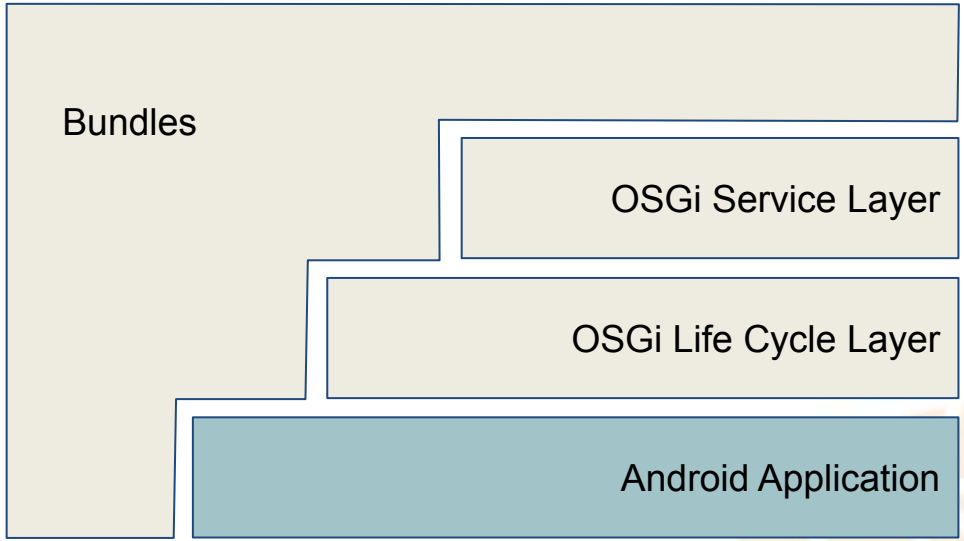
Atomos - OSGi, Native

- 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



OSGi Core Release 8 - OSGi Connect

OSGi Core Release 8

Final Draft Available
https://docs.osgi.org/specification/osgi.core/8.0.0/

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

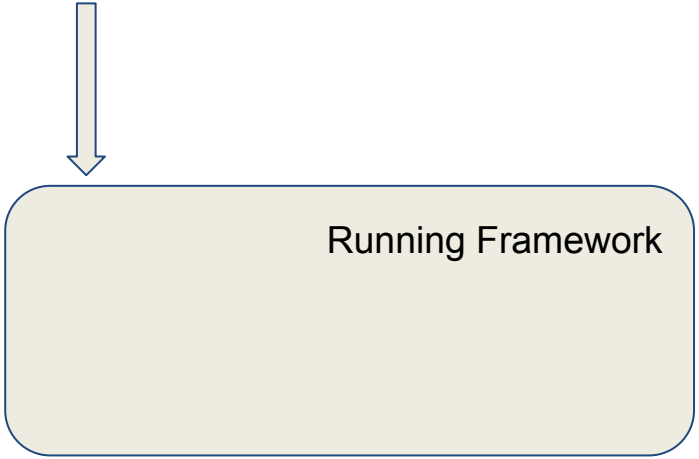
Framework Managed Bundle Content

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

Optional content to read the bundle content from

Framework Managed Bundle Content

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



Framework Managed Bundle Content

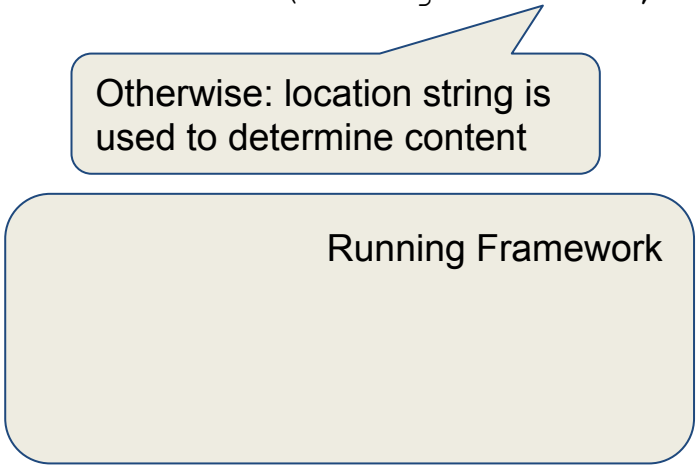
```
installBundle(String location, InputStream content)
```

If content is available: framework persists content to storage

Running Framework

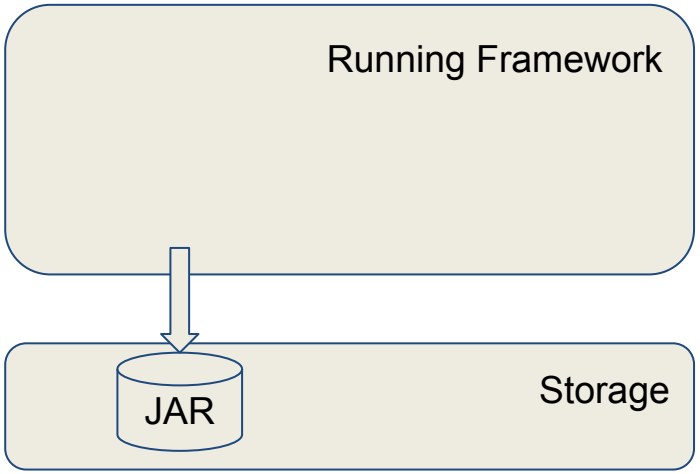
Framework Managed Bundle Content

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



Framework Managed Bundle Content

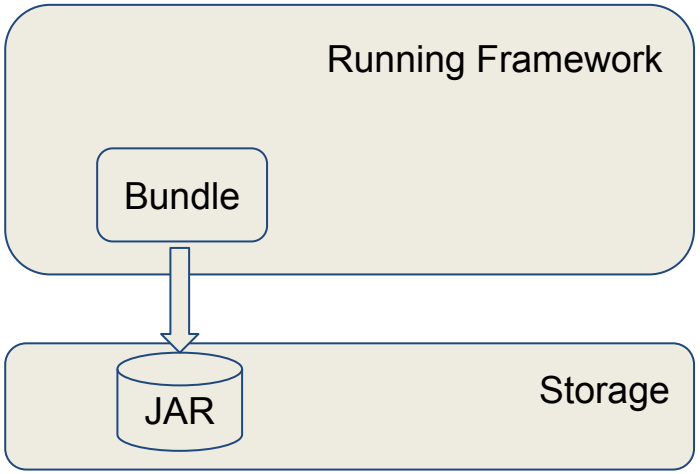
```
installBundle(String location, InputStream content)
```



Persist Bundle JAR to Framework Storage

Framework Managed Bundle Content

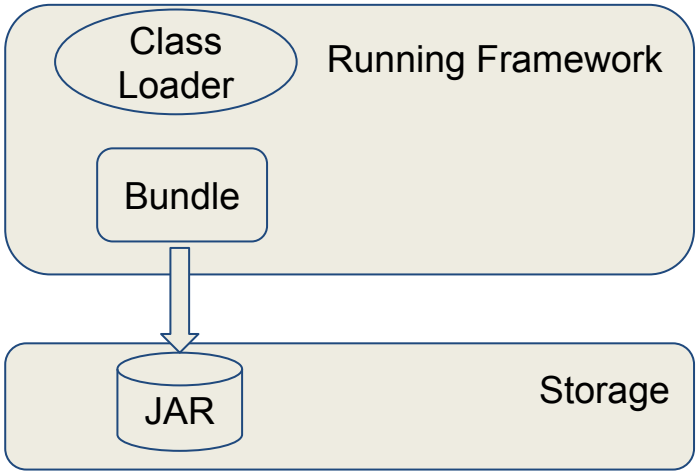
```
installBundle(String location, InputStream content)
```



Read bundle manifest; create Bundle object INSTALLED in the Framework

Framework Managed Bundle Content

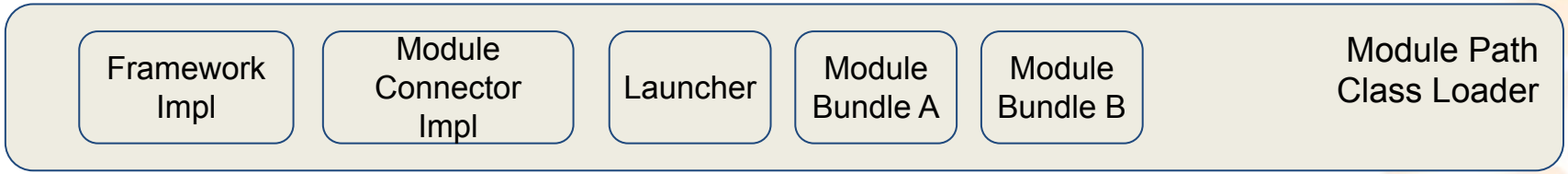
```
installBundle(String location, InputStream content)
```



RESOLVED

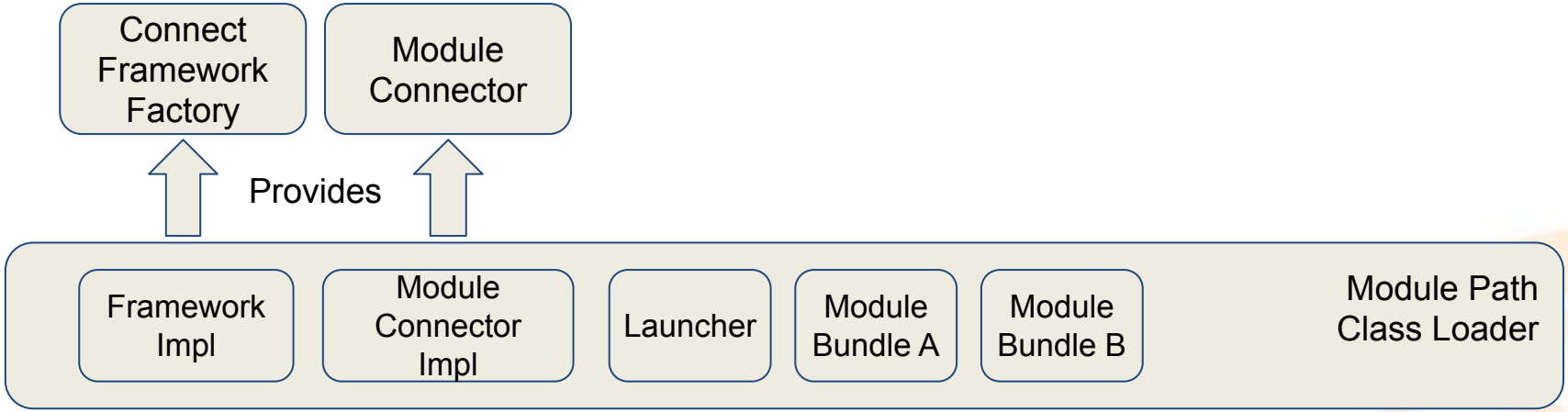
Create a Connect Framework (Module Path Example)

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



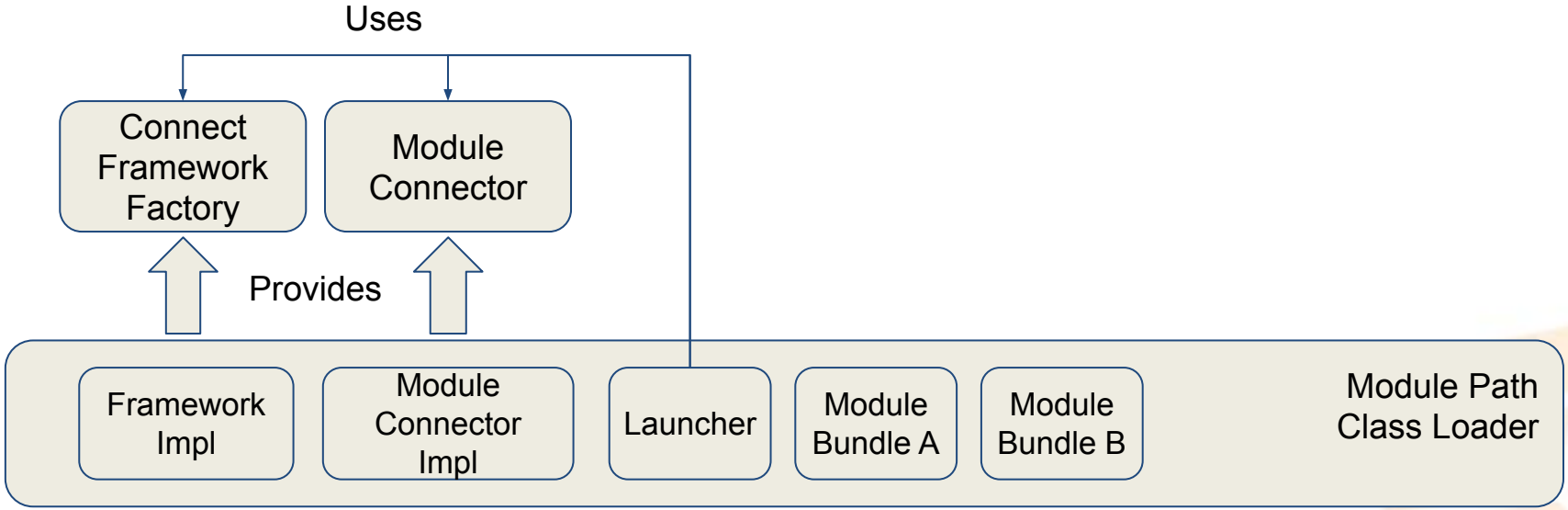
Create a Connect Framework (Module Path Example)

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



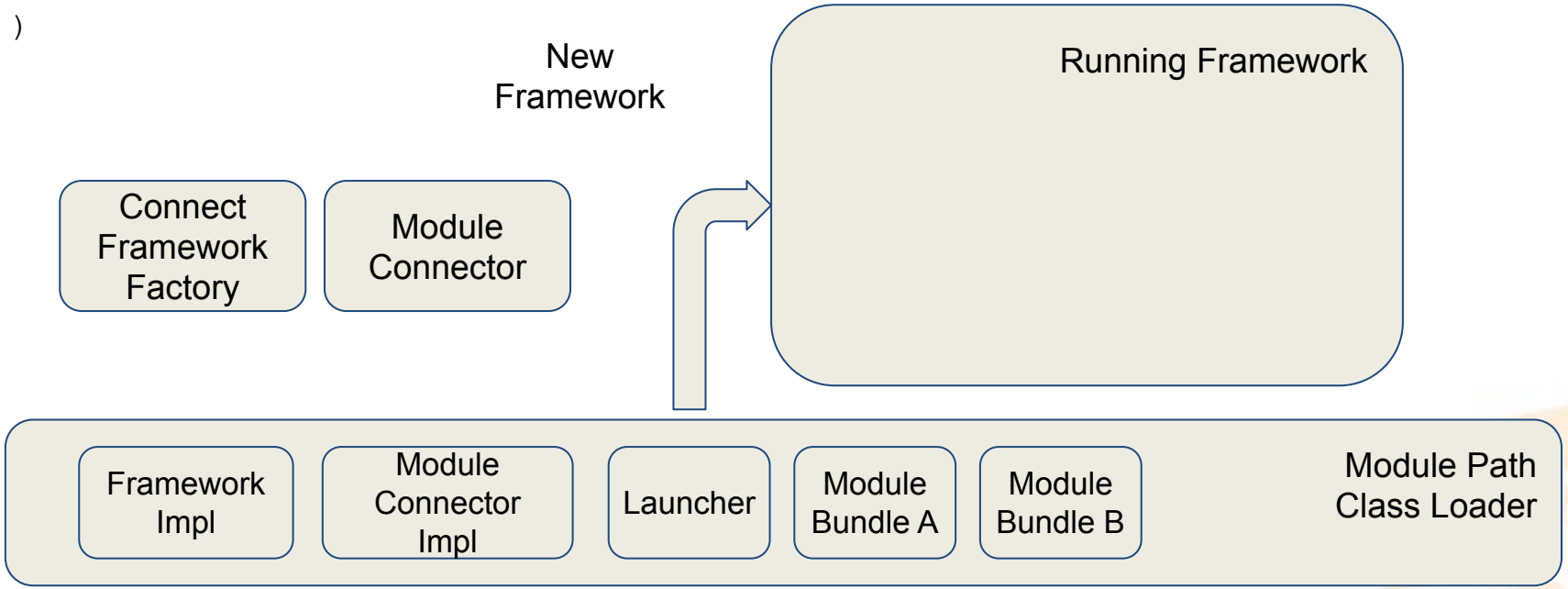
Create a Connect Framework (Module Path Example)

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

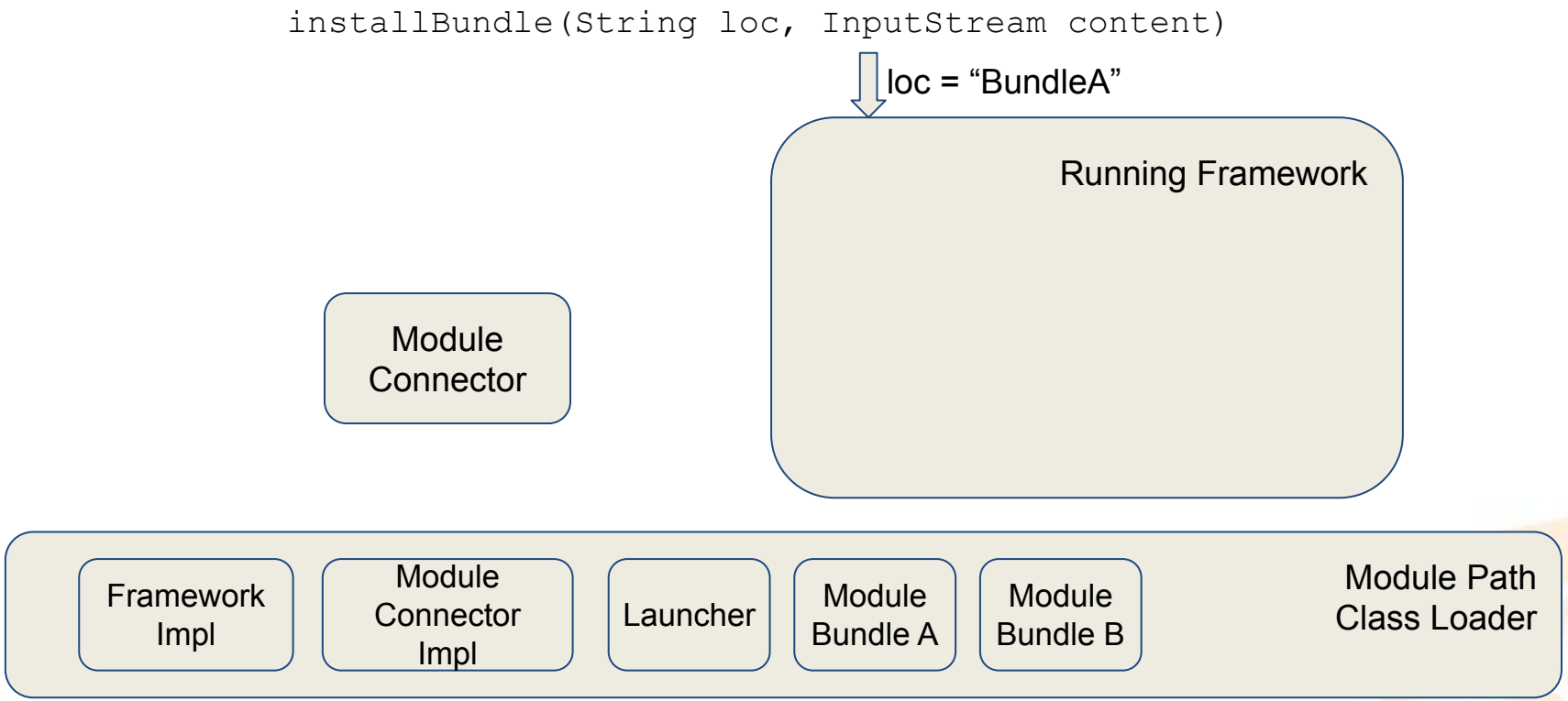


Create a Connect Framework (Module Path Example)

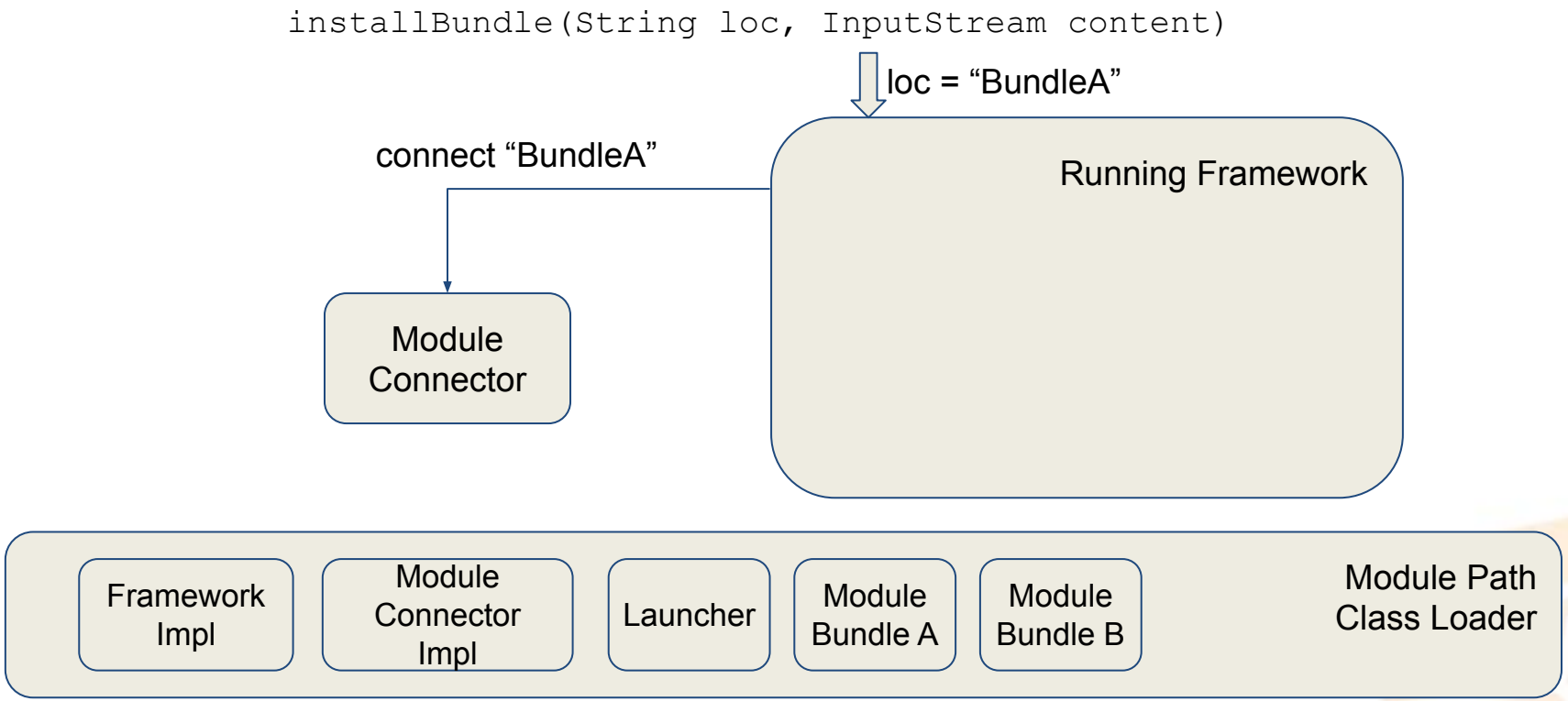
```
ConnectFrameworkFactory.newFramework( Map<String,String>
configuration,

                                      ModuleConnector moduleConnector

)
```



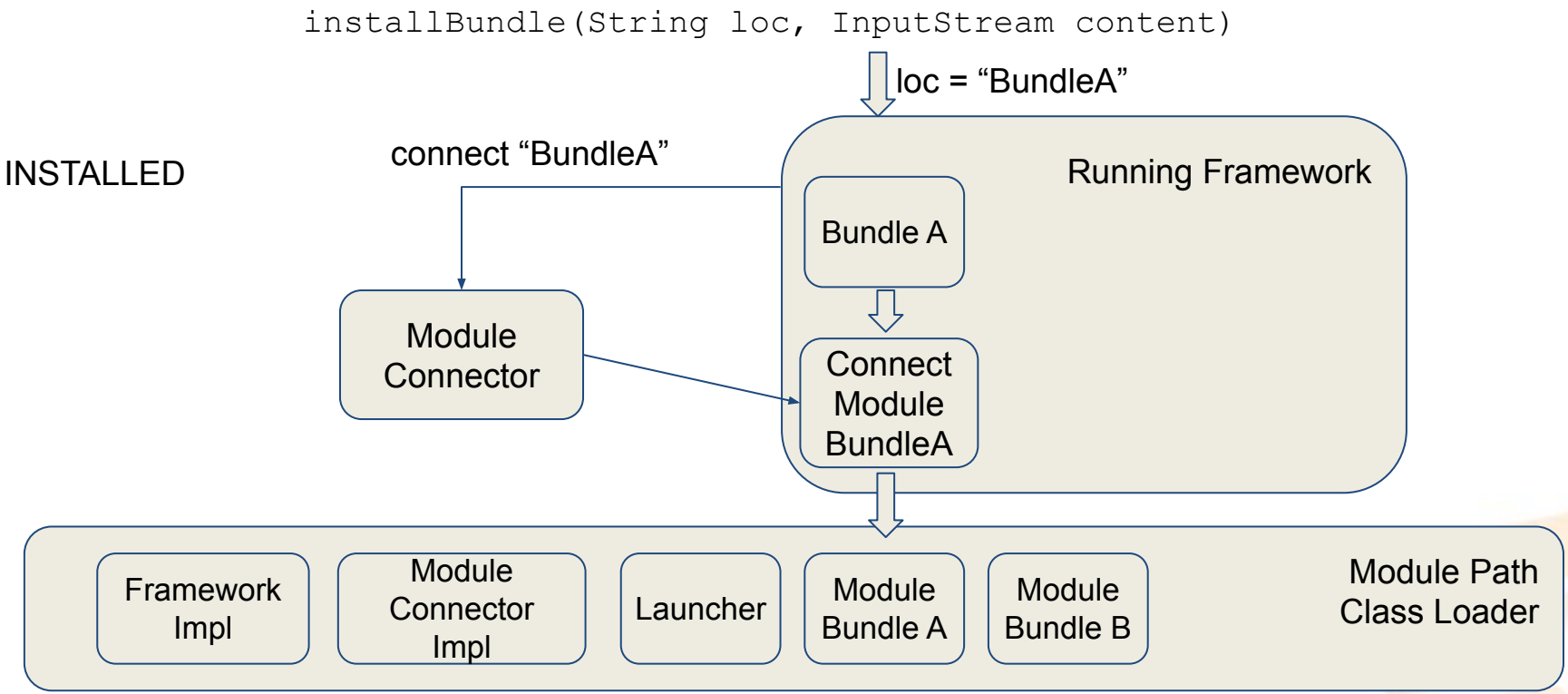
Create a Connect Framework (Module Path Example)



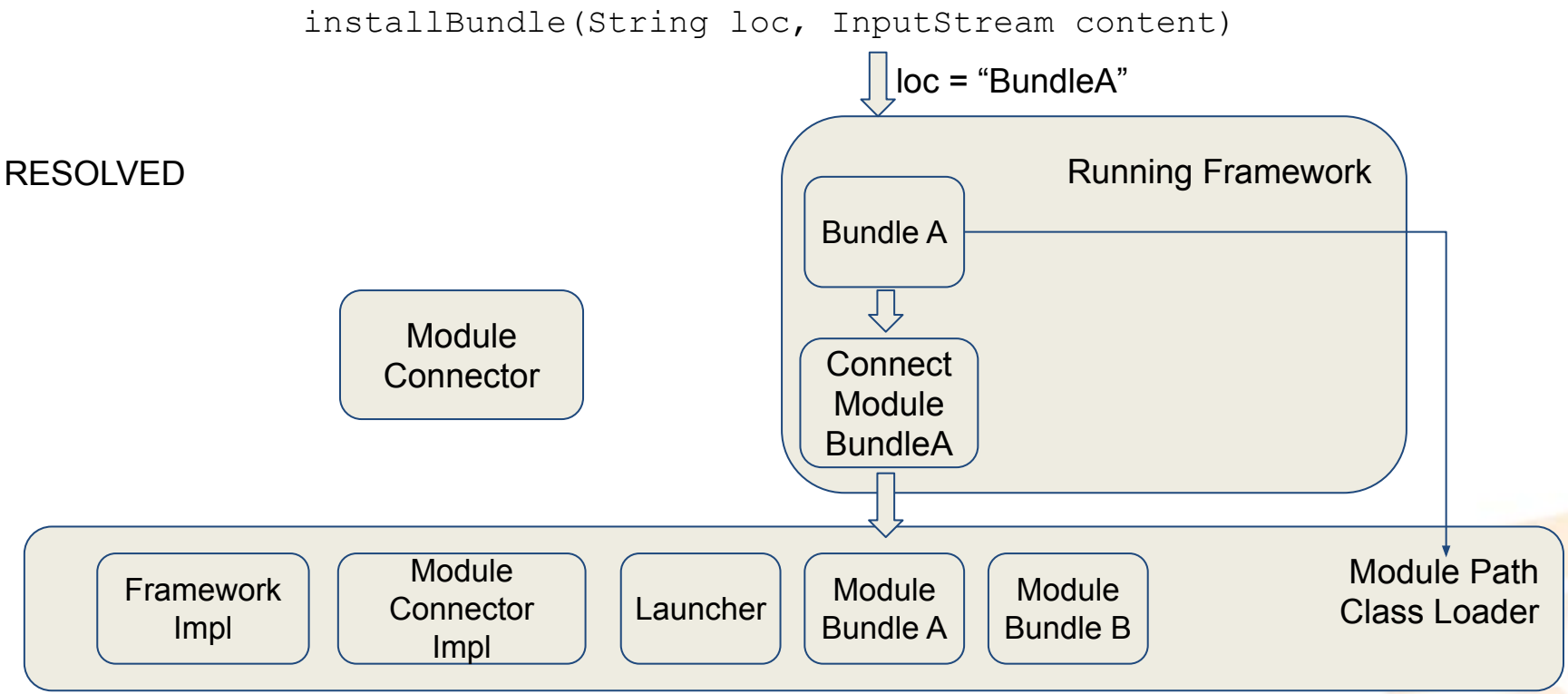
Create a Connect Framework (Module Path Example)



Create a Connect Framework (Module Path Example)



Create a Connect Framework (Module Path Example)



Thank you!

Join the conversation:

 @EclipseCon | #EclipseCon

Evaluate the Sessions

Sign in and vote at Eclipsecon.org:

-1	0	+1

ECLIPSE 2020 CON