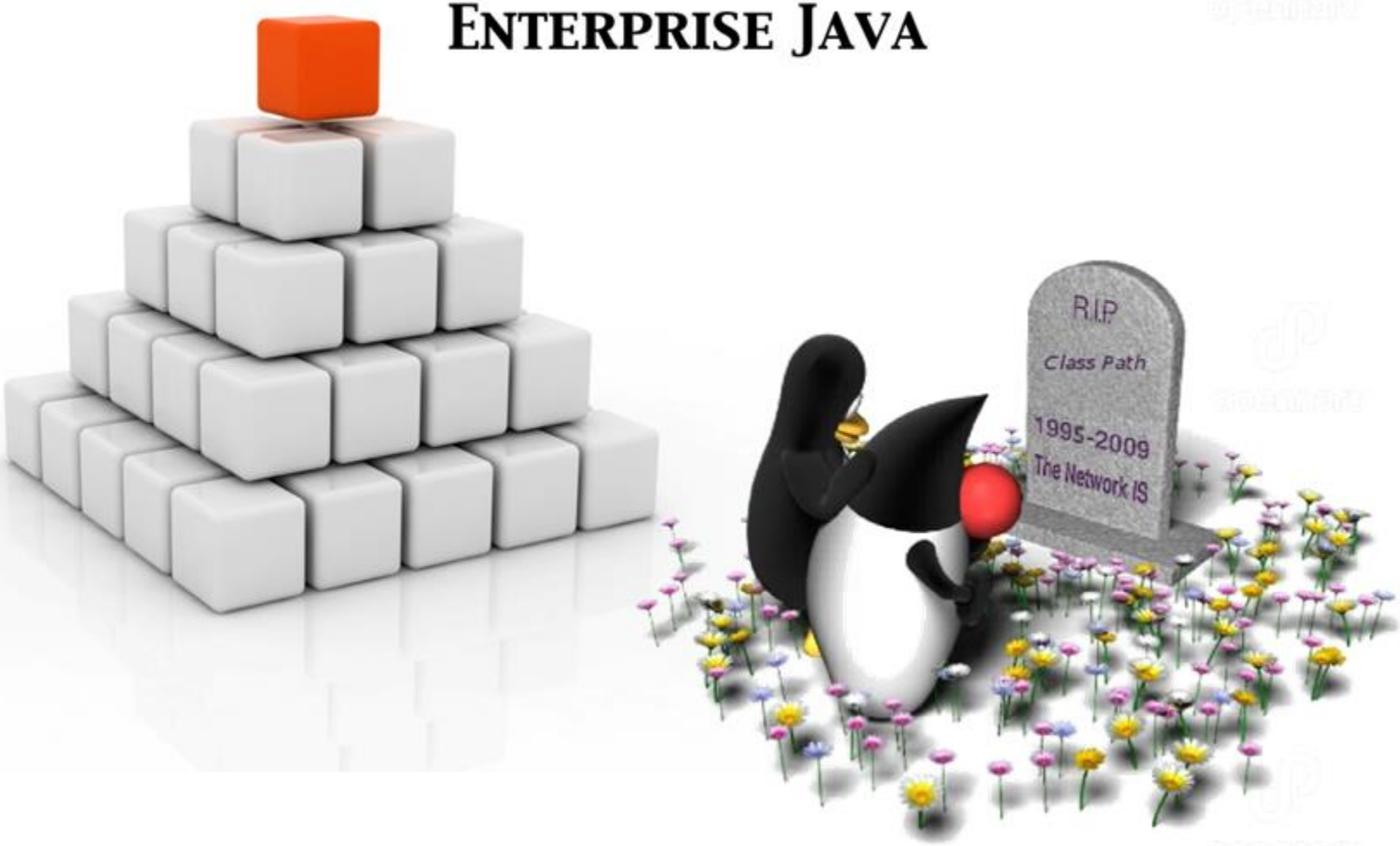
JBoss EAP 7 모듈 아키텍쳐

Opennaru, Inc.

Agenda

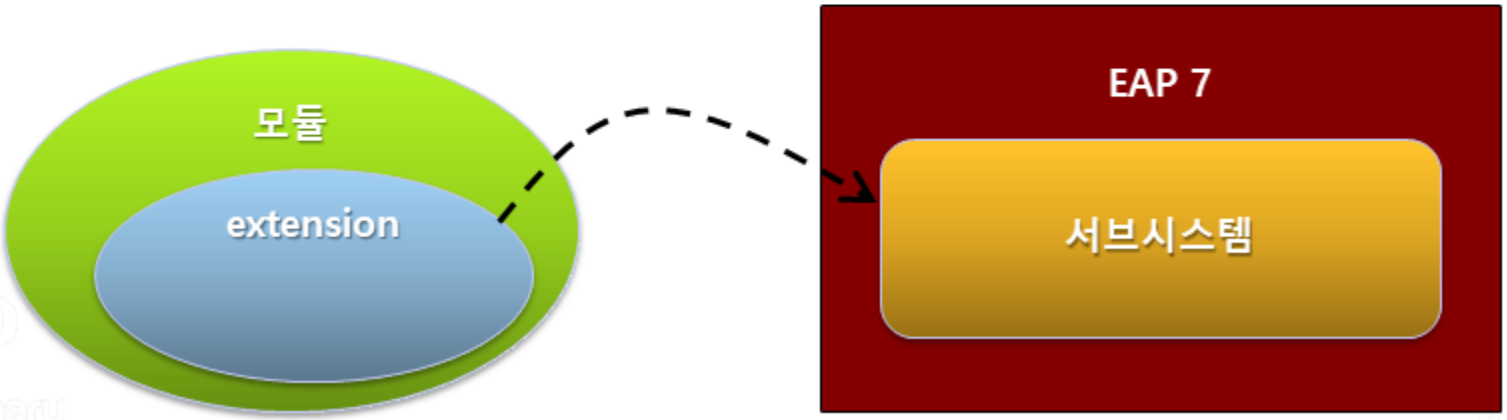
- **모듈 아키텍쳐**
  - MSC(Modular Service Container)
  - Module
  - Dependency
  - Class Loading
  - 사용자 정의 모듈
  - jboss-deployment-structure.xml
  - Global Module

No CLASSPATH !!!

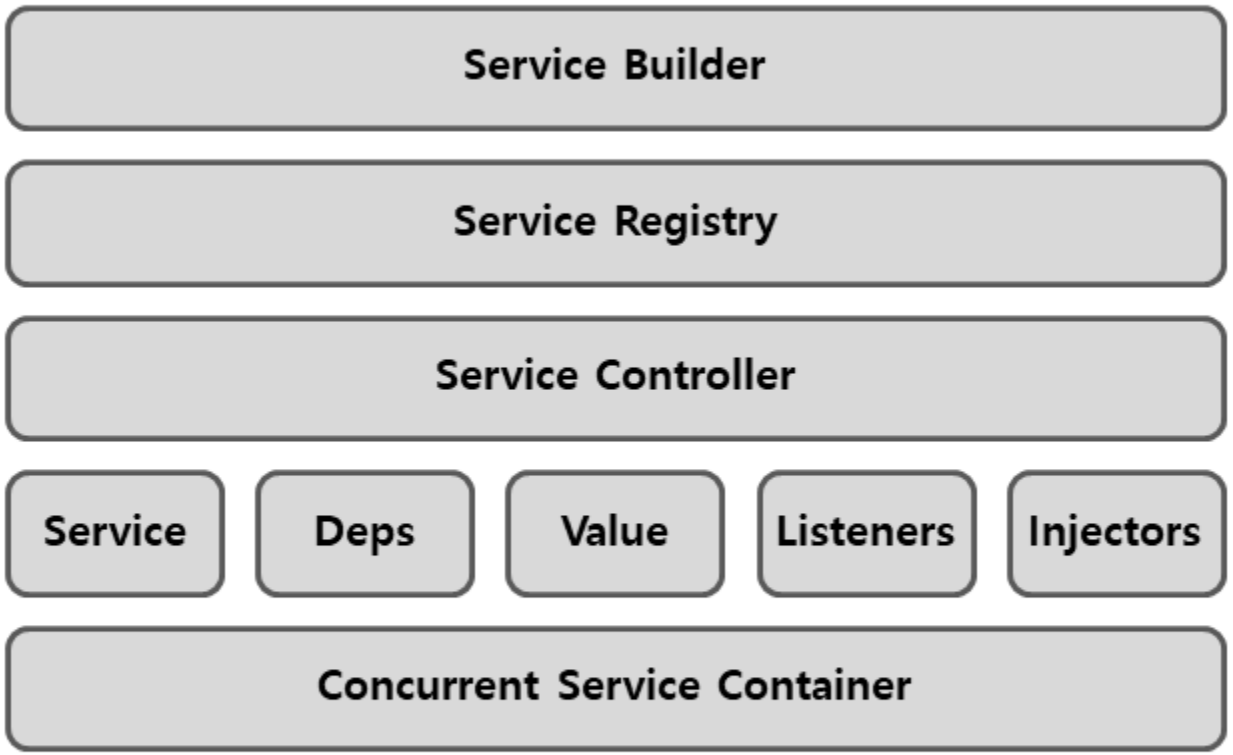


extension와 서브시스템

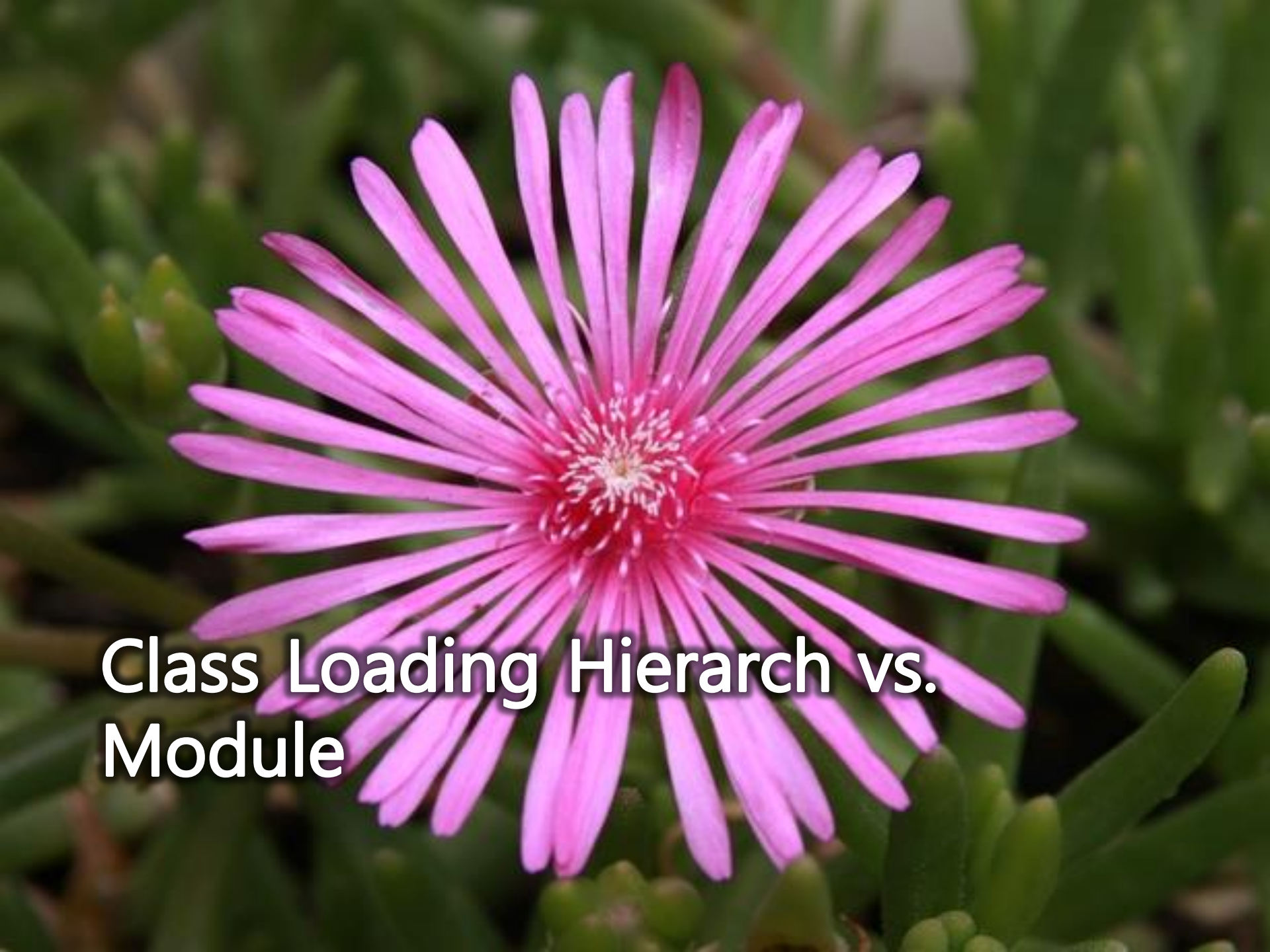
- extension : EAP 7의 기능을 확장하기 위한 모듈
  - Logging, Web, EJB3, Naming, Security…
  - 사용하는 extension를 설정 파일에 등록
- 서브시스템 : extension를 EAP 7에 설치하여 실행시킨 기능(서비스)
  - 서브시스템마다 여러 가지 설정이 가능



JBoss MSC(Modular Service Container)의 코어 서비스

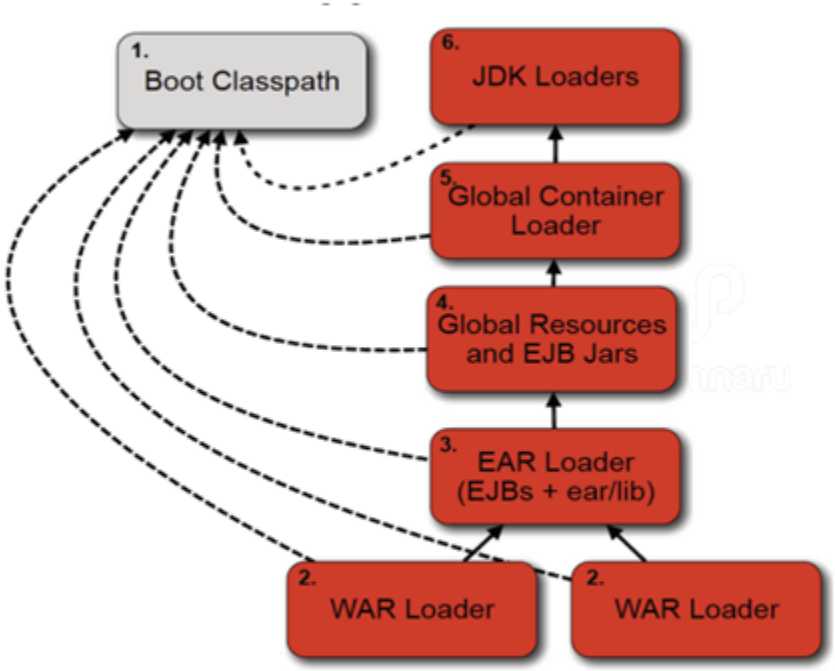


거침없이 배우는 JBoss, 158 page


Class Loading Hierarch vs.
Module

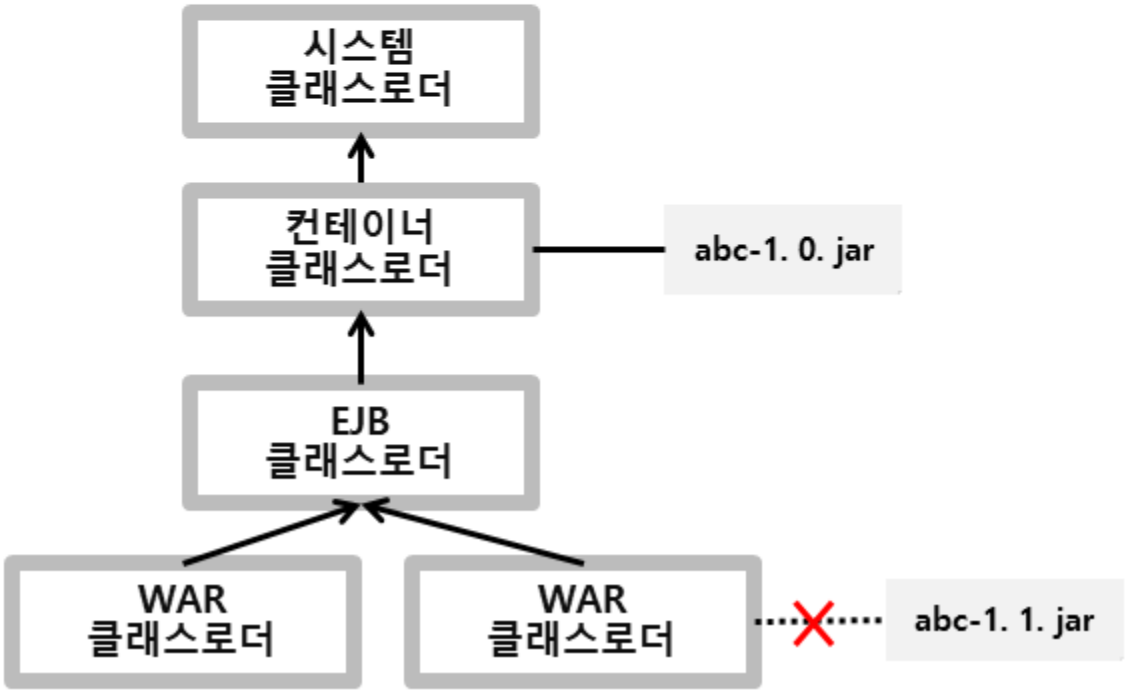
클래스 로딩 모델 – Hierarchy Model

- Java 표준적인 클래스 로딩 모델에서는 여러 개의 클래스로더가 클래스 Hierarchy에 따라 단일의 트리 구조를 구성
- 다양한 어플리케이션이 동시 병렬로 동작하는 어플리케이션 서버는 트리 전체에서 막대한 수의 클래스를 로딩/관리
- 어느 클래스가 어느 클래스로더에서 로딩되는지는 트리를 순회하지 않으면 모르기 때문에 하나의 클래스를 로딩 하는데 다수의 라이브러리가 검색
- 하위의 클래스로더는 다수의 어플리케이션에 공유되기 때문에 다른 버젼 의 동일 라이브러리를 사용하는 경우 모든 클래스의 의존관계(dependencies)를 유지하면서, 하나의 트리 구조에 전체 클래스를 배치하는 것은 매우 어려운 작업

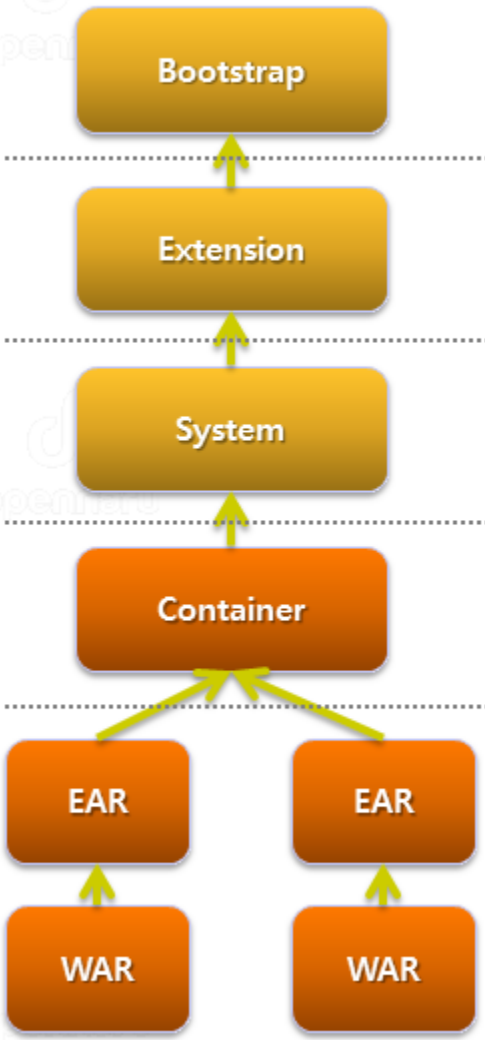


계층형 클래스로더 문제점

- 클래스 로딩 시 상위 계층을 검색하기 때문에 시간이 많이 소요됨
- 애플리케이션에서 사용하지 않는 클래스도 많이 로드됨
- 메모리 사용량이 많아지고 시작 시간이 오래 걸림
- 상위 클래스 로더가 로드한 라이브러리와 다른 버전의 라이브러리를 하위 클래스 로더에서 로드할 수 없음



Java VM의 클래스 로더

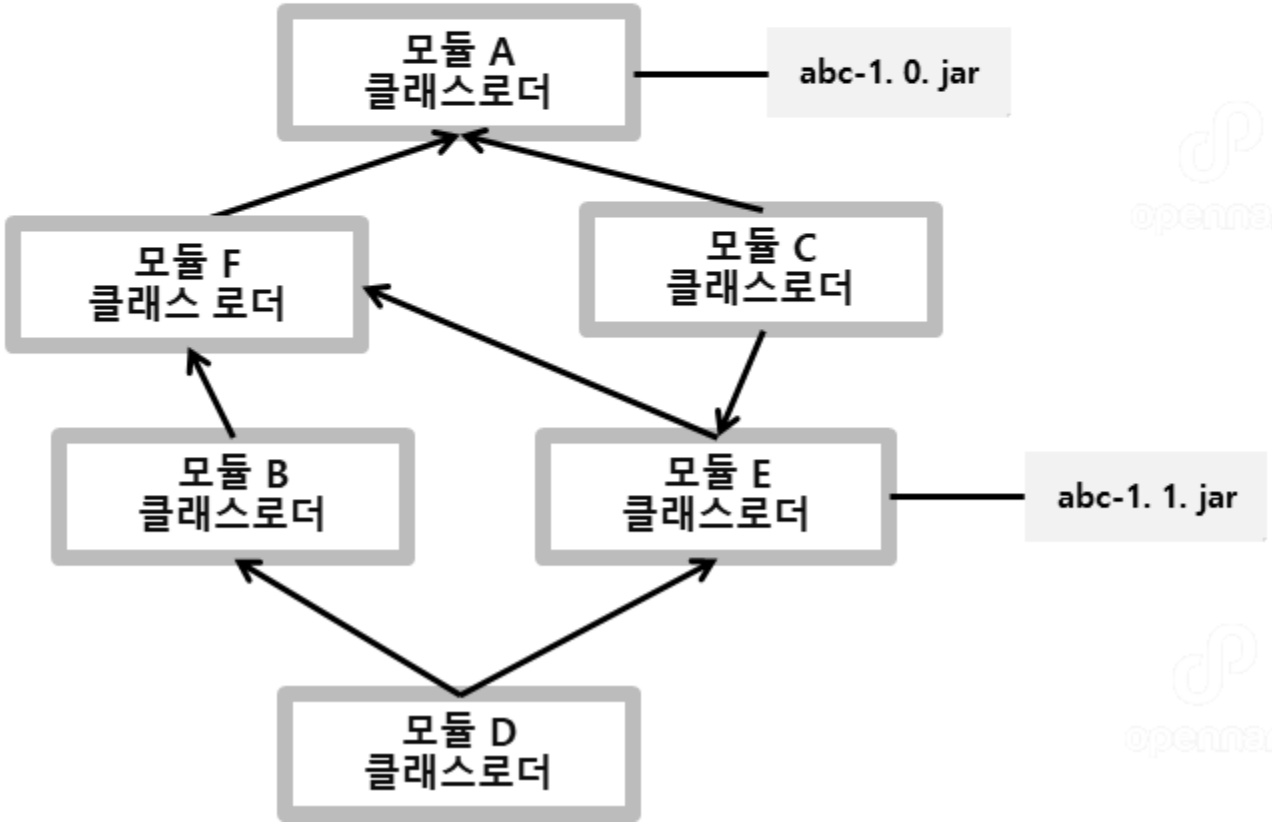


부트스트랩 클래스 로더	• JVM이 실행될 때 맨 처음 실행되는 클래스 로더 • 코어 패키지(rt.jar, i18n.jar) 등의 클래스 로딩
확장 클래스 로더	• Bootstrap 로딩 후 기본적으로 로딩되는 클래스 • $JAVA_HOME/lib/ext에 있는 Class들이 로딩
시스템 클래스 로더	• java.class.path 즉 CLASSPATH 변수에 클래스를 로딩 • -cp, -classpath에 지정된 Class들이 로딩
애플리케이션 클래스 로더	• 사용자 정의 클래스 로더

거침없이 배우는 JBoss, 163 page

계층형 클래스로더의 문제점을 해결

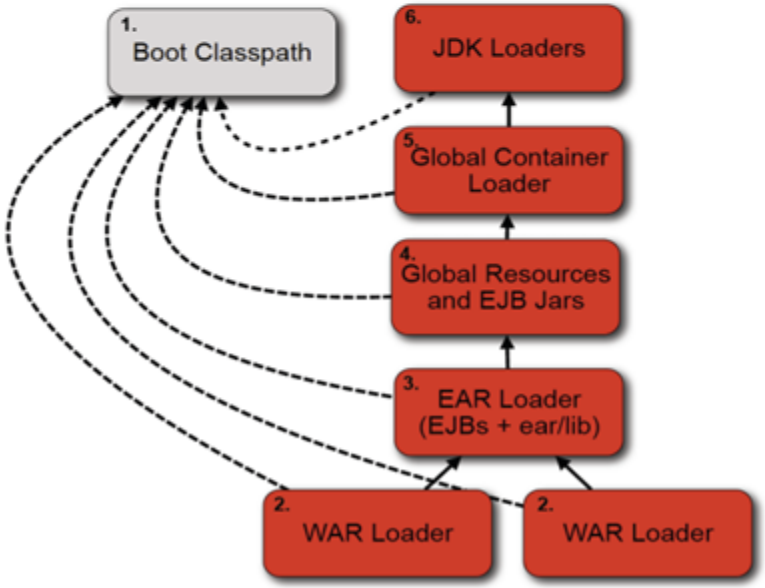
- 클래스 로딩 시 자신(클래스로더)과 의존하는 모듈의 클래스로더 만 검색하기 때문에 클래스 검색이 빠름
- 필요한 모듈이 필요한 때에 로드 되기 때문에, 실행 속도가 빠르며 메모리 사용량이 낮음
- 다른 버전의 라이브러리 로딩도 가능



JBoss EAP 7 Architecture : 선진화된 클래스로더

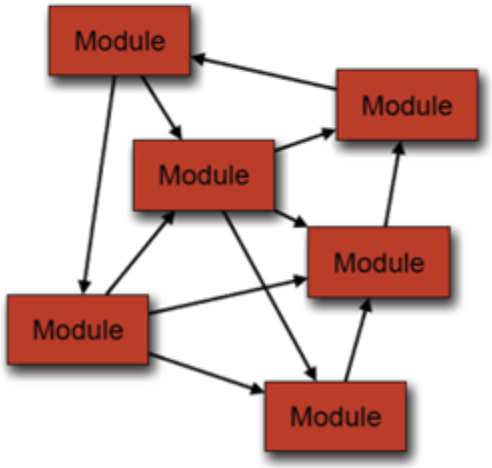
계층형 클래스로더

- 중복 배포
- 로드 순서에 의한 교착 상태 발생
- 복잡/ 클래스 검색이 늦음
- 중복 배포에 의한 오류 발생
- 클래스 공유
- 문제를 회피하기 위한 구조가 더 복잡도를 높여 악순환



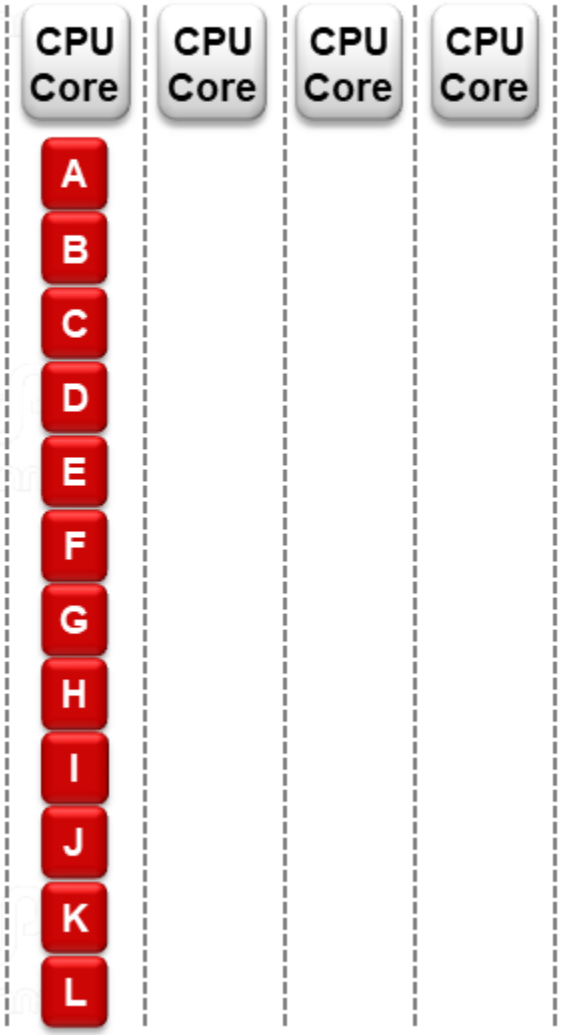
모듈형 클래스로더

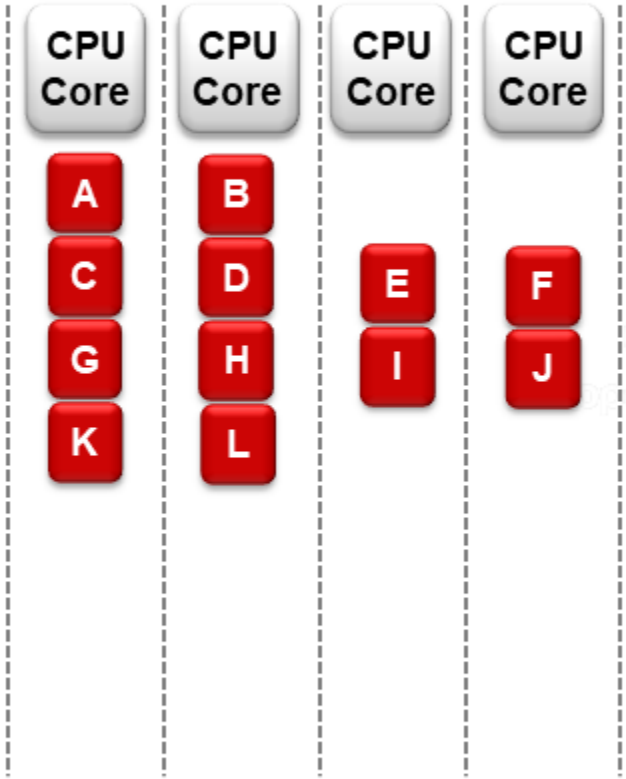
- 계층형 클래스 로더의 문제점 해결
- 모듈 하나에 대해서 하나의 클래스로더
- 각 모듈은 런타임으로 필요로 하는 모듈의 의존성을 정의
- 계층형이 아닌 그래프 구조
- 「클래스 패스」는 사라짐
- 단순하여 초고속




모듈 - 주요 특징

Serial DAG vs Concurrent DAG실행 방법

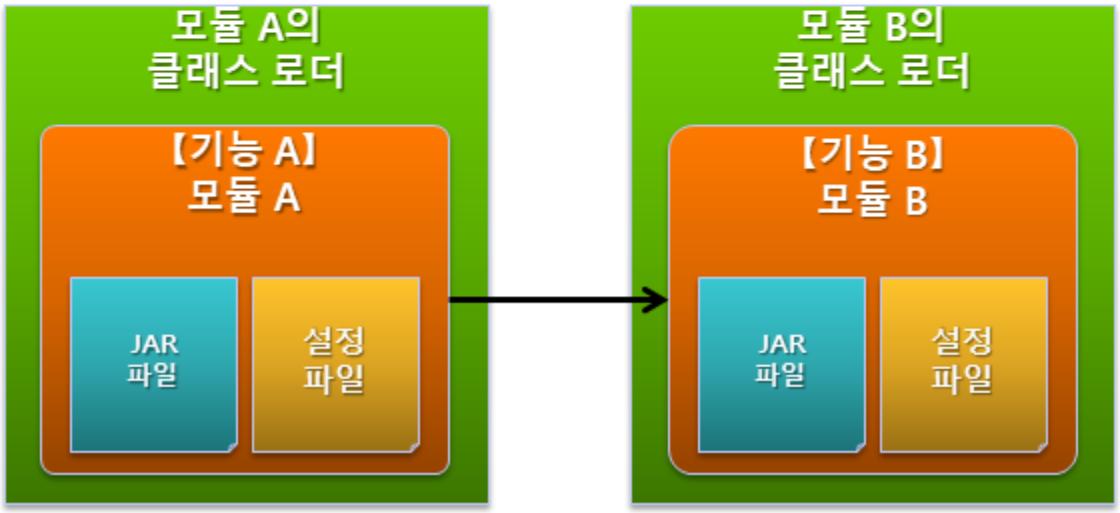




거침없이 배우는 JBoss, 175 page

모듈이란?

- EAP 7의 개별 기능으로 분할된 프로그램 단위
  - EAP 7는 모듈들을 모아서 구성
- 모듈은 물리적으로 JAR 파일과 설정 파일
- 모듈간의 의존성을 명시적으로 지정
- 모듈마다 클래스로더를 할당



모듈의 종류 – module.xml

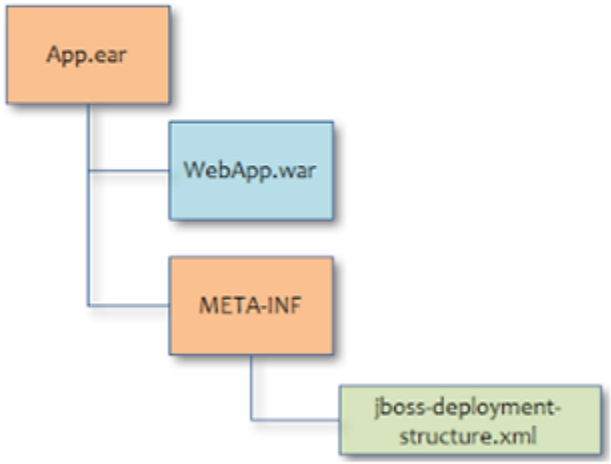
- 모듈은 클래스 로딩 및 종속성 관리에 사용되는 클래스의 논리적 그룹
- module.xml 파일을 통해 정의
- 정적모듈과 동적모듈이라는 두 가지 유형이 있으며 기능은 동일함

정적 모듈

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.0" name="com.
mysql">
  <resources>
    <resource-root path="mysql-connector-java-5.1.1
5.jar"/>
  </resources>
  <dependencies>
    <module name="javax.api"/>
    <module name="javax.transaction.api"/>
  </dependencies>
</module>
```

- 정적 모듈은 EAP_HOME/modules/ 디렉터리에 사전 정의
- 각 서브 디렉터리는 하나의 모듈이고 1 개 이상의 JAR 파일과 설정 파일 (module.xml)이 포함

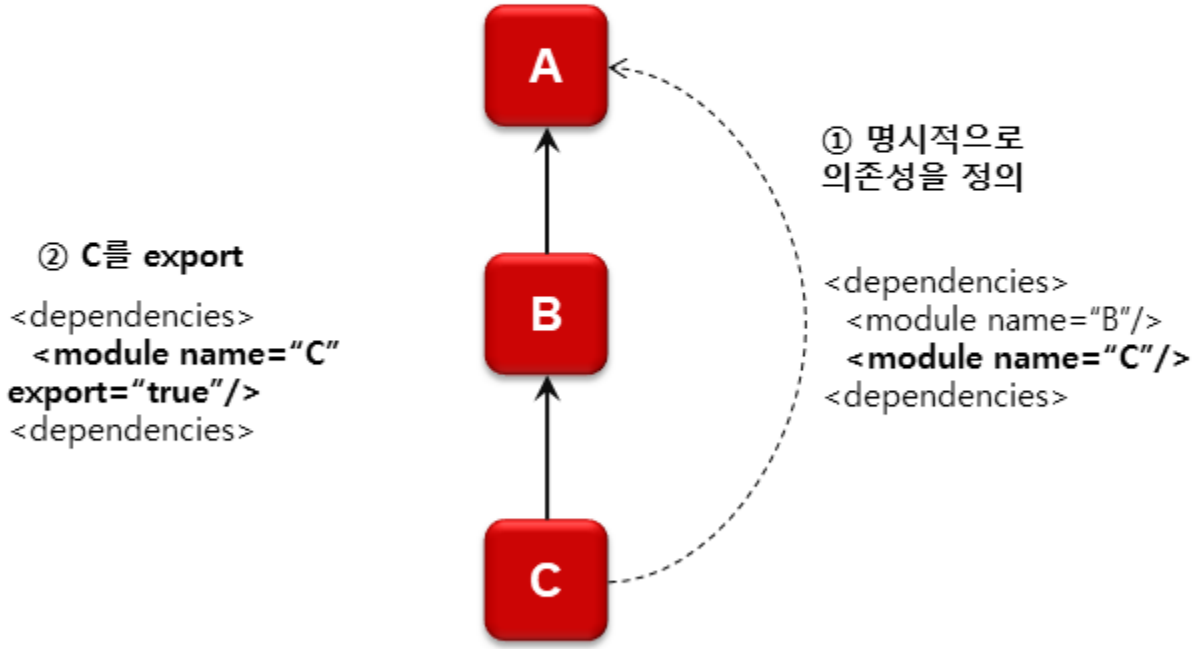
동적 모듈



- 동적 모듈은 각 JAR 또는 WAR 배포되어 Jboss에서 모듈로 자동생성되어 로드
- 동적 모듈의 이름은 배포 된 아카이브 이름에서 명명
- 배포 시에 모듈로 로드되므로 종속성을 설정

JBoss Module에서 의존성 예제

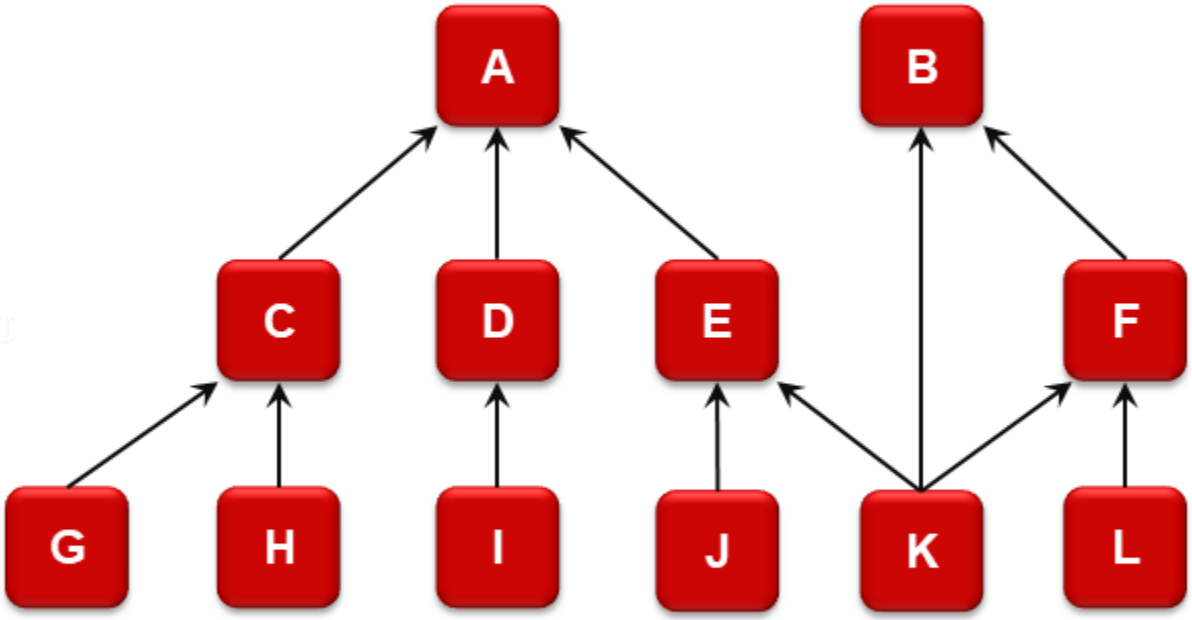
- 모듈 A는 모듈 B에 의존하며, 모듈 B는 모듈 C에 의존한다.
- 모듈 A는 모듈 B 클래스에 액세스 할 수 모듈 B는 모듈 C 클래스에 액세스 할 수 있다.
- 아래의 경우를 제외하고 모듈 A는 모듈 C 클래스에 액세스 할 수 없습니다.



거침없이 배우는 JBoss, 162 page

Module – 주요 설정 방법

모듈의 의존 관계



거침없이 배우는 JBoss, 174 page

사용자 모듈 디렉터리 설정 방법

- layers.conf 파일 작성하여, 사용자가 추가하는 모듈에 대해서
- modules/system/layers/ext 디렉토리를 사용할 수 있도록 설정한다

```
$ mkdir $JBOSS_HOME/modules/system/layers/ext

$ vi $JBOSS_HOME/modules/layers.conf
```

```
layers=ext
```

거침없이 배우는 JBoss, 571 page

모듈의 의존성 정의 - jboss-deployment-structure.xml

- dependency 설정

```
<?xml version="1.0" encoding="UTF-8"?>
<jboss-deployment-structure>
   <deployment>
     <dependencies>
       <module name="org.javassist" />
       <module name="org.apache.velocity" export="TRUE" />
     </dependencies>
   </deployment>
 </jboss-deployment-structure>
```

- exclusion 설정

```
<?xml version="1.0" encoding="UTF-8"?>

<jboss-deployment-structure>
   <deployment>
     <exclusions>
       <module name="org.javassist" />
       <module name="org.dom4j" />
     </exclusions>
   </deployment>
</jboss-deployment-structure>
```

거침없이 배우는 JBoss, 170 page

스프링 프레임워크를 모듈로 만들어 쓸 수 있을까?

- Spring Framework을 모듈로 만들기

```
<?xml version="1.0" encoding="UTF-8"?>
<jboss-deployment-structure>
    <deployment>
        <dependencies>
            <module name="org.springframework.spring" slot="3.1.2">
                <imports>
                    <include path="META-INF**"/>
                    <include path="org**"/>
                </imports>
            </module>
            <module name="org.springframework.security" slot="main"/>
            <module name="org.mybatis" slot="main"/>

            <module name="org.slf4j"/>
            <module name="org.apache.log4j"/>

             <module name="javax.validation.api"/>

        </dependencies>
    </deployment>
</jboss-deployment-structure>
```

거침없이 배우는 JBoss, 6장

JDK 클래스에 접근하기

- JDK 클래스 에 접근하기

sun.jdk 모듈에 JDK에서 제공하는 클래스들을 정의하고 있음
이 모듈에 정의되지 않은 클래스를 로딩하려면 다음과 같이 설정한다.

```
<jboss-deployment-structure xmlns="urn:jboss:deployment-structure:1.1">
    <deployment>
        <dependencies>
            <system>
                <paths>
                    <path name="sun/security/action"/>
                    <path name="javax/crypto"/>
                </paths>
            </system>
        </dependencies>
    </deployment>
</jboss-deployment-structure>
```

https://access.redhat.com/site/solutions/343033

거침없이 배우는 JBoss, 6장

JBoss에서 Jersey 사용시 모듈 설정 방법

- JBoss에서 Jersey(RESTful Web Services 모듈)

JBoss는 RESTEasy 모듈을 기본으로 제공하고 있음
만약 Jersey를 사용하고 싶다면 아래와 같이 모듈을 설정

```
<jboss-deployment-structure xmlns="urn:jboss:deployment-structure:1.1">
    <deployment>
        <dependencies>
            <system>
                <paths>
                    <path name="com/sun/xml/bind/v2/runtime/JAXBContextImpl"/>
                </paths>
            </system>
        </dependencies>
    </deployment>
</jboss-deployment-structure>
```

https://access.redhat.com/site/solutions/330173

거침없이 배우는 JBoss, 6장

Global 모듈

- **WAS 시작시 CLASSPATH에 설정하여야만 하는 JAR를 사용하려면**

일부 암호화 모듈이 이런 경우가 있음

- **module.xml**

```
<?xml version="1.0" encoding="UTF-8"?>

<module xmlns="urn:jboss:module:1.1" name="com.opennaru">

    <resources>
        <resource-root path="abc.jar"/>
        <resource-root path="def.jar"/>
    </resources>

    <dependencies>
    </dependencies>
</module>
```

- **standalone*.xml / domain.xml**

```
<subsystem xmlns="urn:jboss:domain:ee:1.1">
    <spec-descriptor-property-replacement>false</spec-descriptor-property-replacement>
    <jboss-descriptor-property-replacement>true</jboss-descriptor-property-replacement>
    <global-modules>
        <module name="com.opennaru" slot="main"/>
    </global-modules>
</subsystem>
```

거침없이 배우는 JBoss, 6장

제품이나 서비스에 관한 문의

콜 센터 : 02-469-5426 (휴대폰 : 010-2243-3394)

전자메일 : sales@opennaru.com