OSGi CDI Integration Specification

Raymond Augé - Sr. Software Architect

@rotty3000

Liferay

Why CDI In OSGi?

- Reduce developer friction

- Important Java specification

- Benefit from extensive feature set

But Declarative Services (DS)?

Liferay loves DS!

99% of all **Liferay** bundles (jars) are DS and the vast majority will remain DS forever.

Liferay

WHY use anything else?

by design DS is...

ultra light weight, DS annotations are syntax sugar, **CLASS retention** and processed at build time, runtime overhead is extremely low, does not provide an integration SPI, does not provide intra-bundle dependency injection.

Liferay

CDI

as part of its feature set, is...

- **extensible** (CDI has a full fledged SPI)
- **annotation processing** engine
- **intra-bundle** dependency injection

Custom annotations!

@rotty3000   Liferay

CDI

allows for...

completely internal wiring.

Liferay

DS - Internal wiring: new

```
1 @Component
2 public class FooImpl {
3   private Pojo pojo;
4
5   public FooImpl() {
6     pojo = new PojoImpl();
7   }
8 }
```

Liferay

CDI - Internal wiring: @Inject

```java
public class FooImpl {
  private Pojo pojo;

  @Inject
  public FooImpl(Pojo pojo) {
    this.pojo = pojo;
  }
}
```

Liferay

DS - Services: singleton

```
1 @Component
2 public class FooImpl implements Function {
3   ...
4 }
```

Liferay

OSGi-CDI - Services: singleton

```java
@Service
public class FooImpl implements Function {
  ...
}
```

Liferay

DS - Services: prototype

```
1 @Component(scope = PROTOTYPE)
2 public class FooImpl implements Function {
3   ...
4 }
```

Liferay

OSGi-CDI - Services: prototype

```
1  @Service @ServiceInstance(PROTOTYPE)
2  public class FooImpl implements Function {
3    ...
4  }
```

Liferay

DS - References

```
1 @Reference
2 Pojo pojo;
```

Liferay

OSGi-CDI - References

```
1 @Inject @Reference
2 Pojo pojo;
```

Liferay

DS - Cardinality: mandatory

```
1 @Reference
2 Pojo pojo;
```

@rotty3000

Liferay

OSGi-CDI - Cardinality: mandatory

```
1 @Inject @Reference
2 Pojo pojo;
```

Liferay

DS - Cardinality: optional

```
1 @Reference(cardinality = OPTIONAL)
2 Pojo pojo;
```

Liferay

OSGi-CDI - Cardinality: optional

```
1 @Inject @Reference
2 Optional<Pojo> pojo;
```

@rotty3000   Liferay

DS - Cardinality: multiple

```
1 @Reference
2 List<Pojo> pojos;
```

Liferay

OSGi-CDI – Cardinality: multiple

```
1 @Inject @Reference
2 List<Pojo> pojos;
```

@rotty3000   Liferay

DS - Cardinality: at least one (or n)

```
1 @Reference(cardinality = AT_LEAST_ONE)
2 List<Pojo> pojos;
```

Liferay

OSGi-CDI – Cardinality: at least one (or n)

```
1 @Inject @Reference @MinimumCardinality(1)
2 List<Pojo> pojos;
```

Liferay

DS - Reference Policy: greedy

```
1 @Reference(policyOption = GREEDY)
2 Pojo pojo;
```

Liferay

OSGi-CDI - Reference Policy: reluctant

```
1  @Inject @Reference @Reluctant
2  Pojo pojo;
```

Liferay

DS - Dynamic: mandatory

```
1 @Reference(policy = DYNAMIC)
2 volatile Pojo pojo;
```

Liferay

OSGi-CDI - Dynamic: mandatory

```
1 @Inject @Reference
2 Provider<Pojo> pojo;
```

Liferay

DS - Dynamic: multiple

```
1 @Reference(policy = DYNAMIC)
2 volatile List<Pojo> pojos;
```

Liferay

OSGi-CDI - Dynamic: multiple

```
1 @Inject @Reference
2 Provider<List<Pojo>> pojos;
```

Liferay

DS - Dynamic: optional

```
1 @Reference(policy = DYNAMIC, cardinality = OPTIONAL)
2 volatile Pojo pojo;
```

Liferay

OSGi-CDI - Dynamic: optional

```
1 @Inject @Reference
2 Provider<Optional<Pojo>> pojo;
```

Liferay

DS - OSGi Logger

```
1 @Reference(service = LoggerFactory.class)
2 Logger logger;
```

OSGi-CDI - OSGi Logger

```
1 @Inject
2 Logger logger;
```

Liferay

DS - Configuration

```
1 @Activate
2 Map<String, Object> props;
```

OSGi-CDI - Configuration

```
1  @Inject @ComponentProperties
2  Map<String, Object> props;
```

Liferay

Configuration Types

```java
@Retention(RUNTIME)
@BeanPropertyType // OSGi-CDI
@ComponentPropertyType // DS
public @interface Config {
  String hostname() default "localhost";
  int port() default 8080;
  Config.Protocol protocol() default
  Config.Protocol.https;

  public enum Protocol {http, https}
}
```

Liferay

DS - Configuration: typed

```
1 @Activate
2 Config config;
```

Liferay

OSGi-CDI - Configuration: typed

```
1 @Inject @ComponentProperties
2 Config config;
```

Liferay

DS - Component

```
1 @Component(
2   configurationPid = {"foo", "bar"},
3   configurationPolicy = REQUIRE)
4 public class FooImpl {
5   ...
6 }
```

Liferay

OSGi-CDI - Single Component

```java
@SingleComponent
@PID("foo")
@PID(value = "bar", policy = REQUIRED)
public class FooImpl {
    ...
}
```

OSGi-CDI - Factory Component

```
1 @FactoryComponent("foo")
2 @PID("bar")
3 public class FooImpl {
4    ...
5 }
```

Liferay

The OSGi-CDI Spec

https://osgi.org/specification/osgi.enterprise/7.0.0/service.cdi.html

Liferay

The OSGi-CDI Reference Implementation

https://github.com/apache/aries-cdi

Liferay