Spring Framework

Olivier Liechti & Laurent Prévost
COMEM Web Services

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

What is Spring Framework?

- It is a bunch development libraries: it provides various modules to develop software components.
- Unlike Java EE, it is **not** an execution platform: it **does not** provides an environment to deploy and bring these components “to live”.
- It is not exactly an entreprise platform, but it provides support for distributed transactions, security, integration, etc. **Same as J2EE.**
- Separation of concerns: "The developer takes care of the business logic. Spring modules take care of the systemic qualities”. **Same as J2EE.**

http://flickr.com/photos/decade_null/427124229/sizes/m/#cc_license

Spring Framework

- Spring Framework is not a single library. It's a bunch of libraries where you choose the ones you need (transitive dependencies are managed for you by tools like Maven).

- It brings the **Dependency Injection** also known as **Inversion of Control** (**IoC**)

> *"The question is, what aspect of control are [they] inverting?" Martin Fowler posed this question about **Inversion of Control (IoC)** on his site in 2004. Fowler suggested renaming the principle to make it more self-explanatory and came up wit**h Dependency Injection**.*

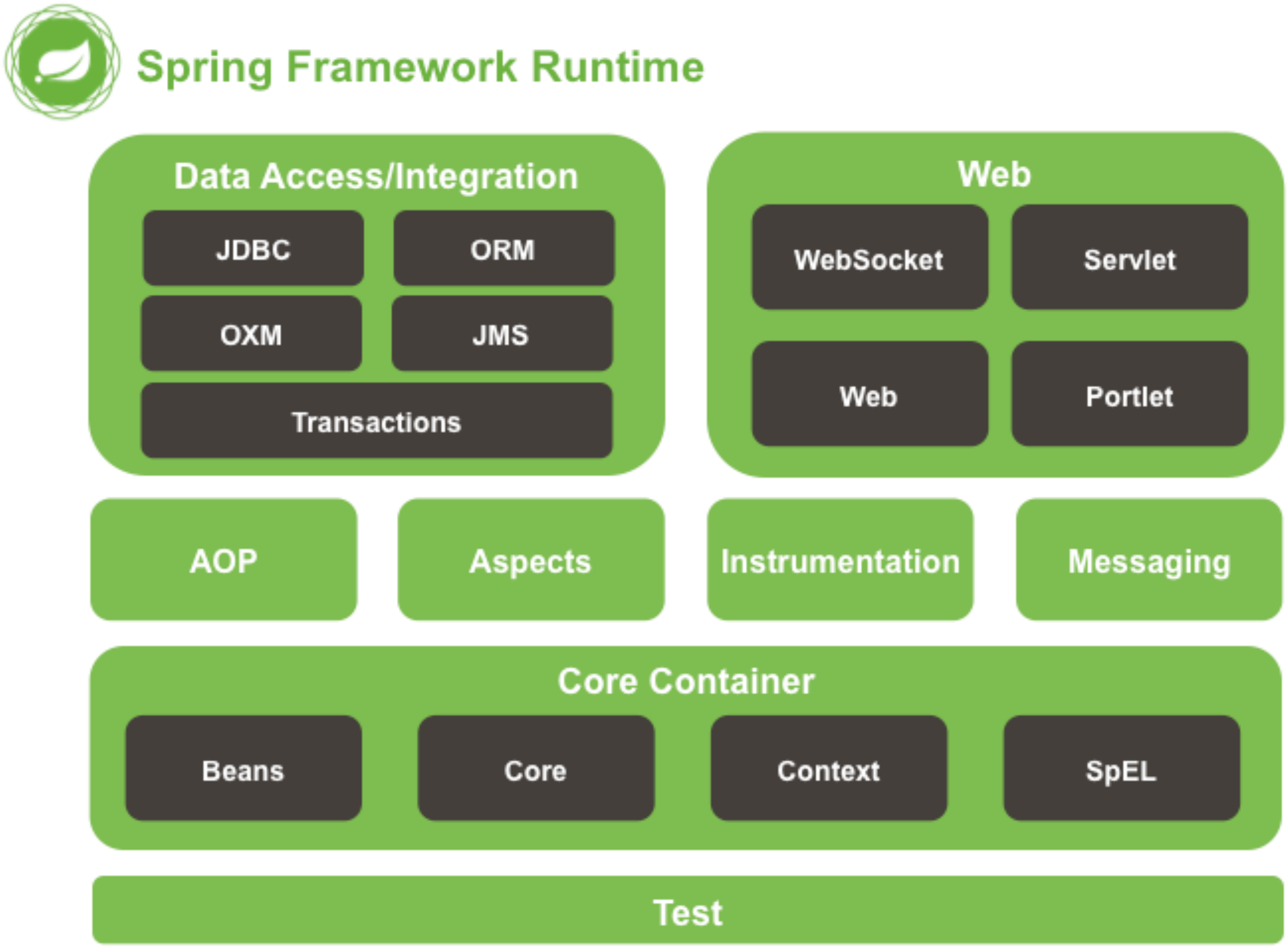
http://martinfowler.com/articles/injection.html

- Provide various default implementations to solve problems without having to code them yourself. For example:
  - Spring Security offers Basic Authentication filter to allow your users to authenticate to your service via HTTP basic authentication.

Architecture

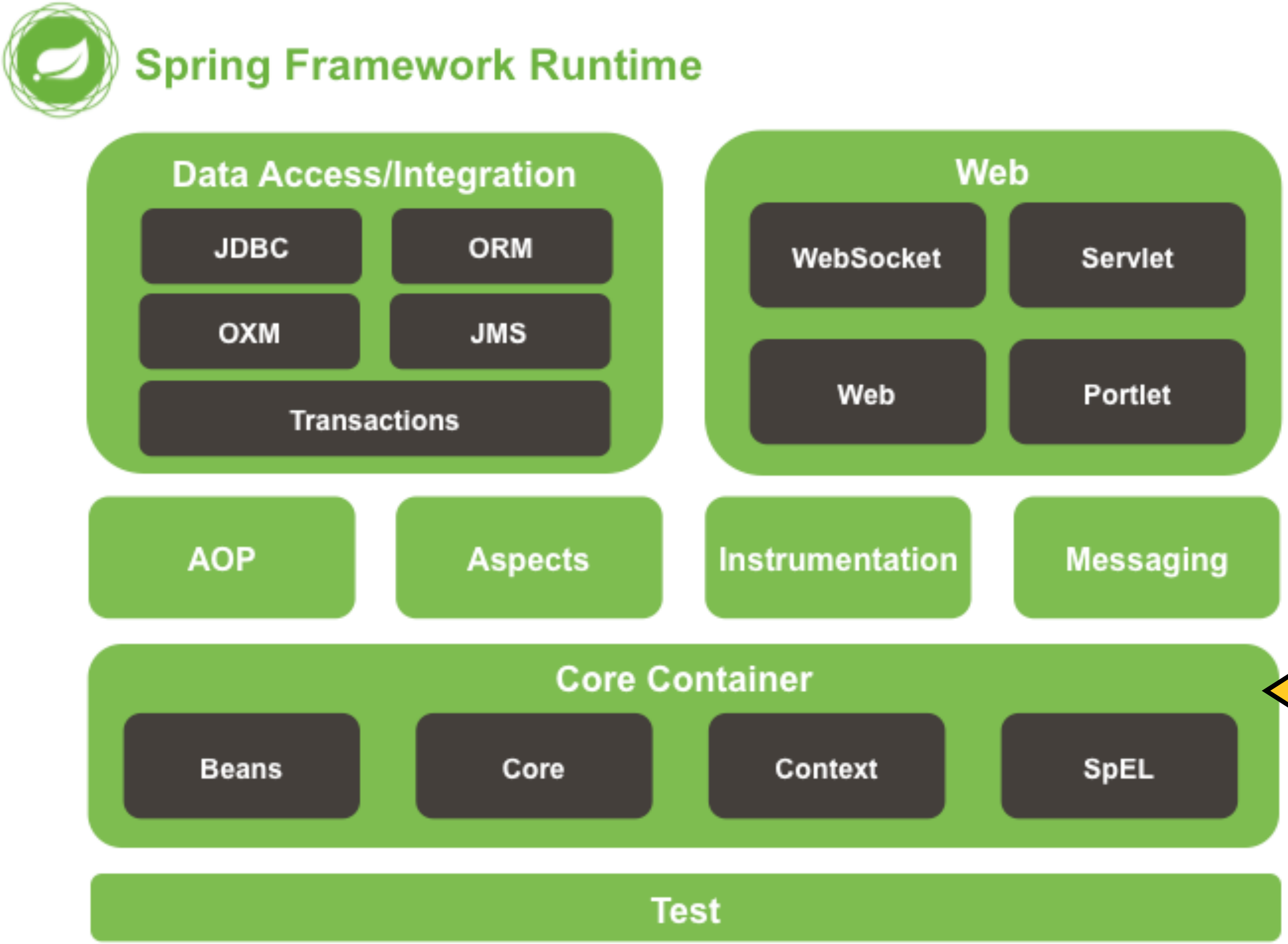
- Spring Framework modules are libraries like any other library. It means that you can add one or more to your classpath and develop your app like any others.
  - Spring Framework modules provide bindings and integrations with most of Java EE Specs:
    - JPA, JMS, ...
- Out of the box, Spring Framework cannot run on its own. The application developed with Spring Framework modules must be deployed in an application server.
  - the container is the **environment** in which we run components;
  - the container **provides services** (transactions, security, etc.) that a Spring Framework can use through the binding modules;
  - the different containers present in Java EE are also available for Spring Framework. In fact, as you develop an app for a container with Spring libraries, you choose which container you want to use.

Spring Framework Modules



http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules

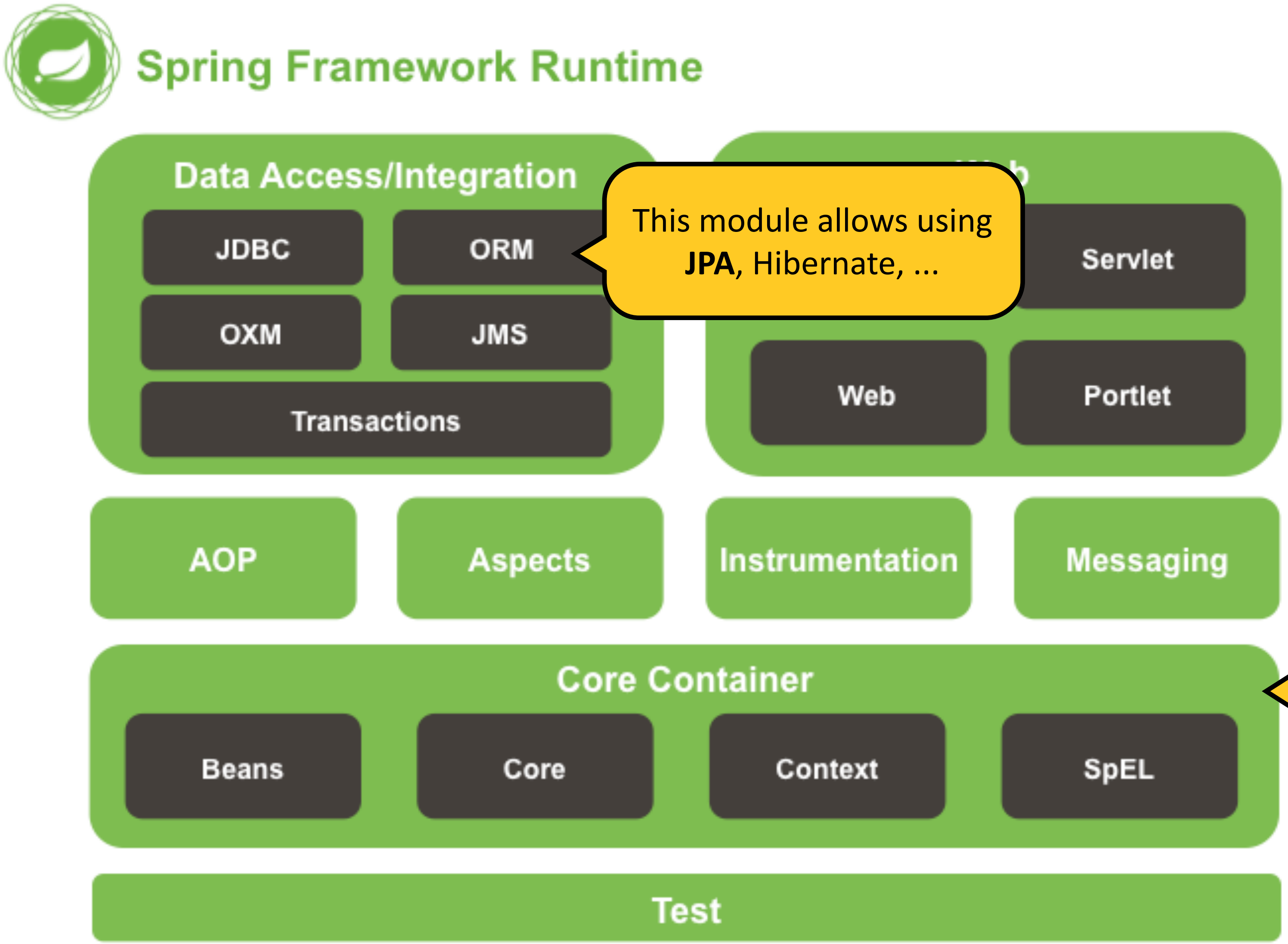
Spring Framework Modules



Main features that brings you **IoC** and **Dependency Injection**.

http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules

Spring Framework Modules

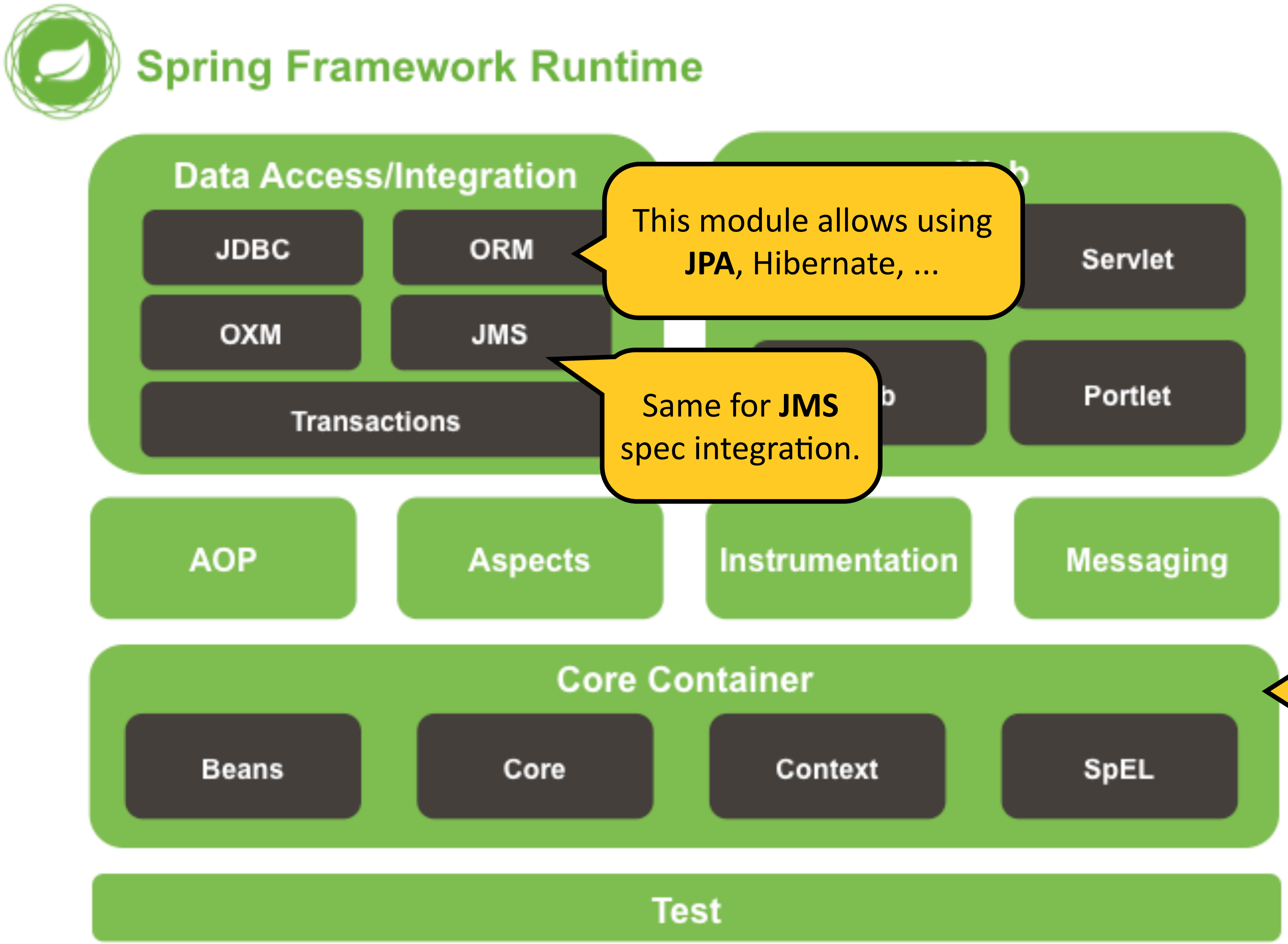


This module allows using **JPA**, Hibernate, ...

Main features that brings you **IoC** and **Dependency Injection**.

http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules

Spring Framework Modules



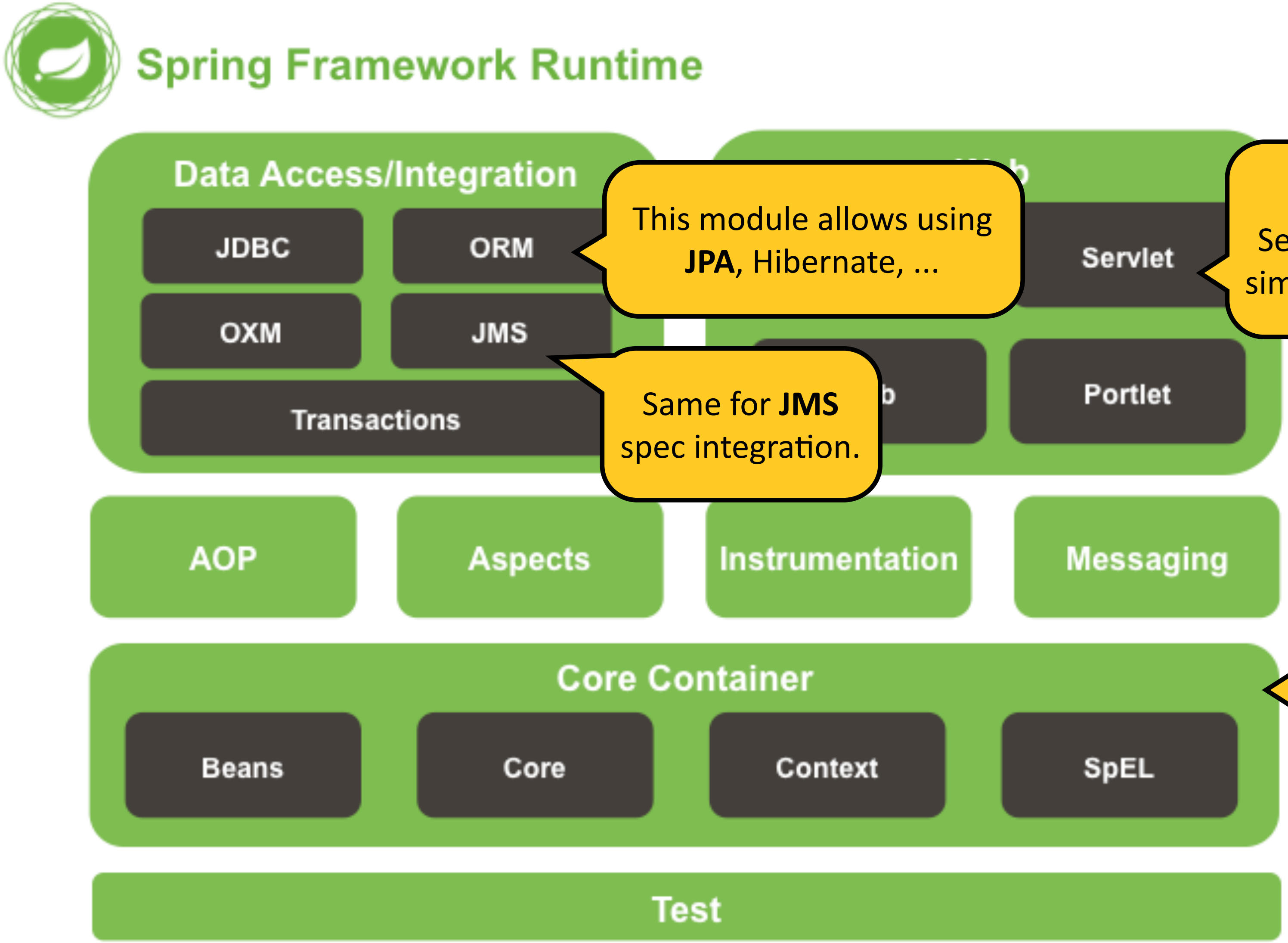
This module allows using **JPA**, Hibernate, ...

Same for **JMS** spec integration.

Main features that brings you **IoC** and **Dependency Injection**.

http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules

Spring Framework Modules



This module allows using **JPA**, Hibernate, ...

Same for **JMS** spec integration.

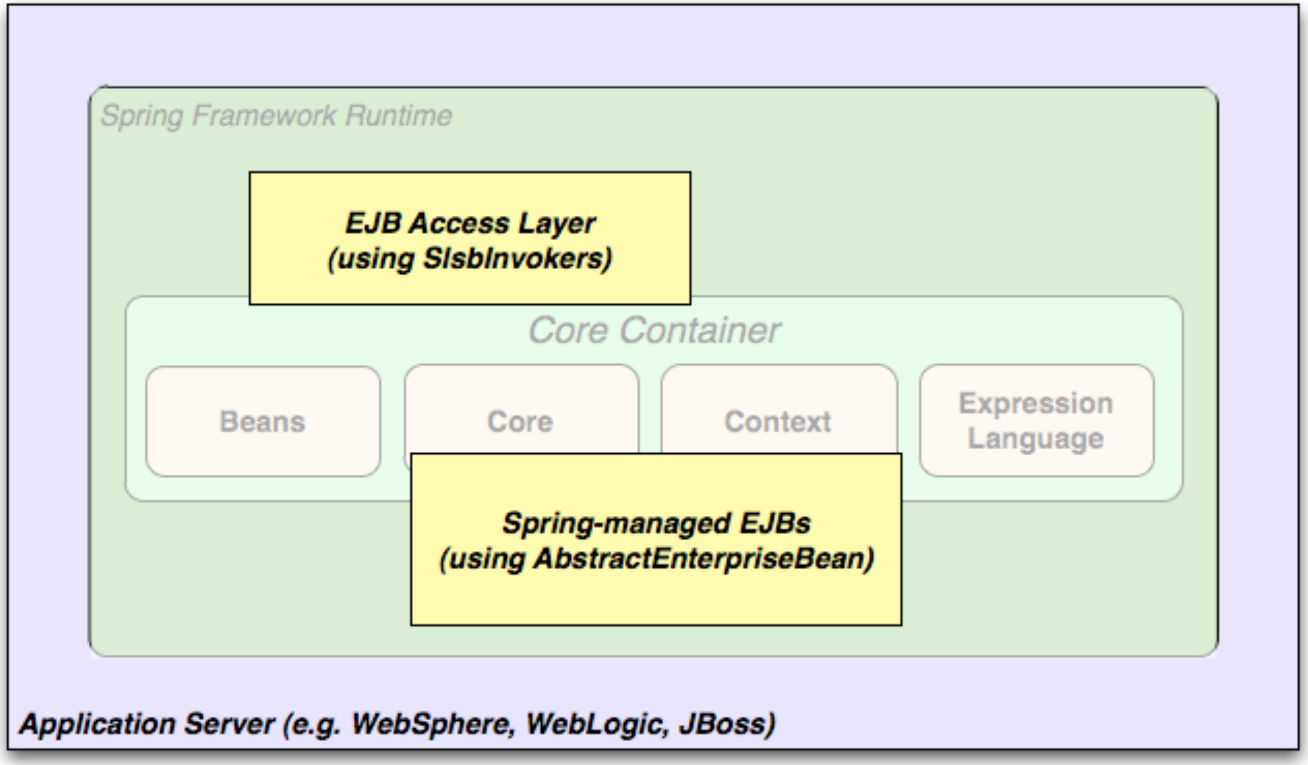
Complies to the spec for Servlets. Therefore, we can do similar things like we do in J2EE.

Main features that brings you **IoC** and **Dependency Injection**.

http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules

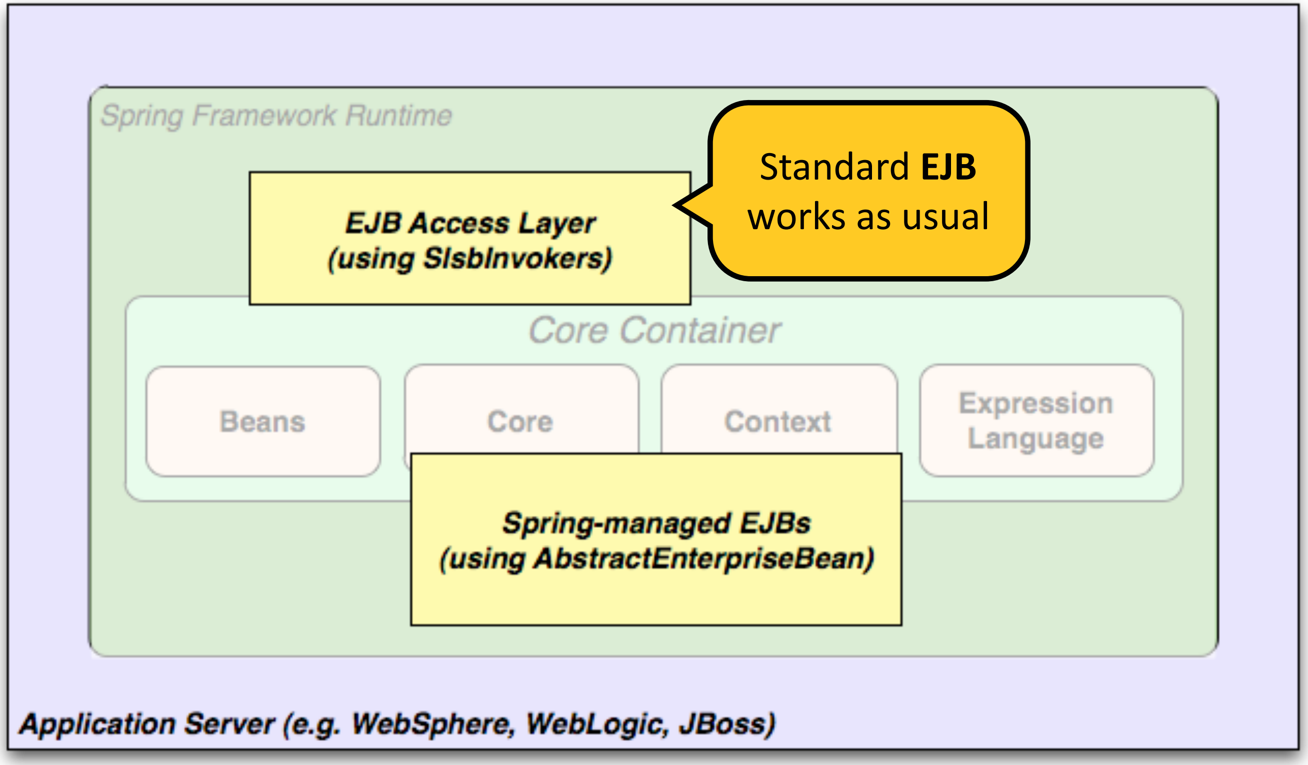
Spring Framework - Example of Integration

http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules


Spring Framework Runtime
EJB Access Layer (using SlsbInvokers)
Core Container
Beans
Core
Context
Expression Language
Spring-managed EJBs (using AbstractEnterpriseBean)
Application Server (e.g. WebSphere, WebLogic, JBoss)

Spring Framework - Example of Integration

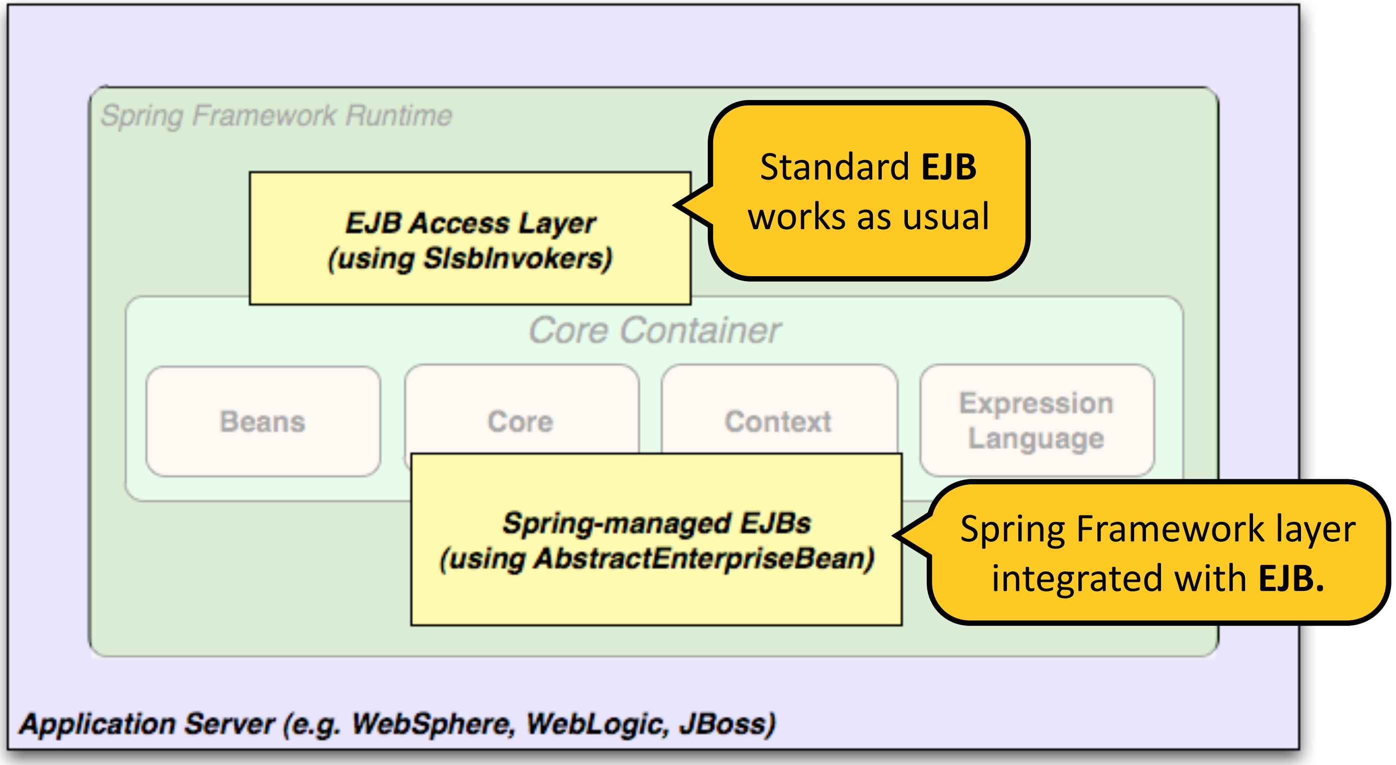
http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules


Spring Framework Runtime
EJB Access Layer (using SlsbInvokers)
Core Container
Beans
Core
Context
Expression Language
Spring-managed EJBs (using AbstractEnterpriseBean)
Application Server (e.g. WebSphere, WebLogic, JBoss)

Standard **EJB** works as usual

Spring Framework - Example of Integration

http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules


Spring Framework Runtime
EJB Access Layer (using SlsbInvokers)
Core Container
Beans
Core
Context
Expression Language
Spring-managed EJBs (using AbstractEnterpriseBean)
Application Server (e.g. WebSphere, WebLogic, JBoss)

Standard **EJB** works as usual

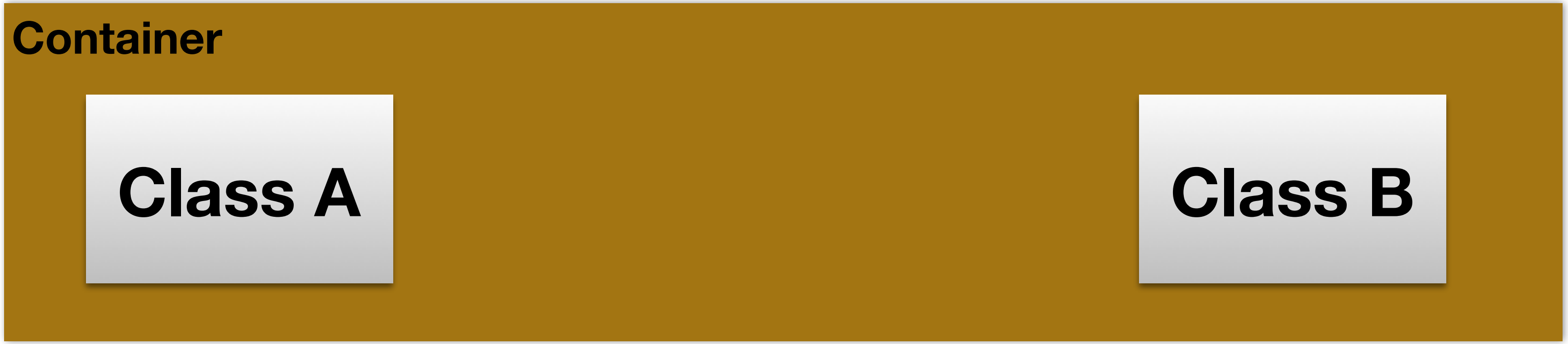
Spring Framework layer integrated with **EJB.**

IoC - Dependency Injection

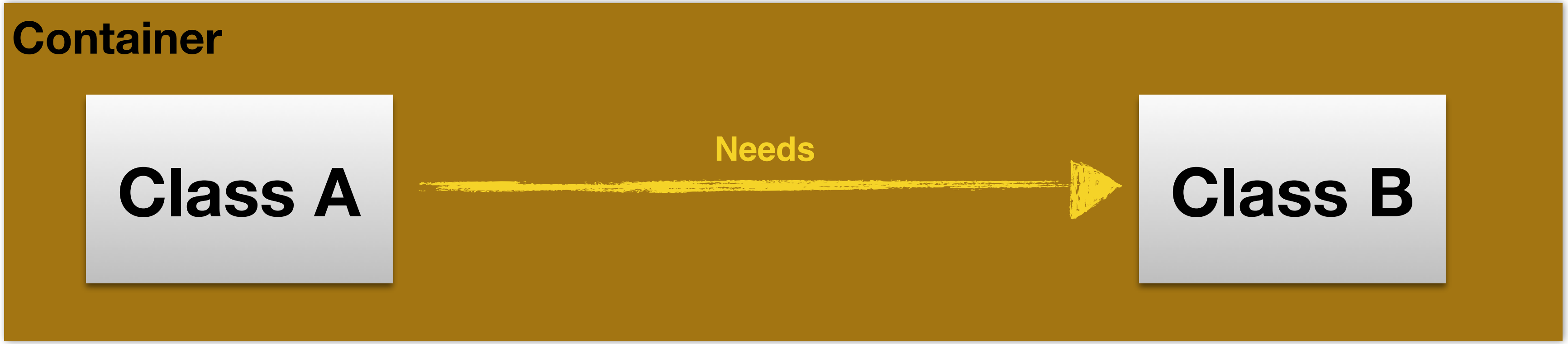
*In software engineering, **dependency injection** is a software **design pattern** that implements **inversion of control** for software libraries, where the caller **delegates** to an **external framework** the **control flow** of discovering and importing a service or software module.*

http://en.wikipedia.org/wiki/Dependency_injection

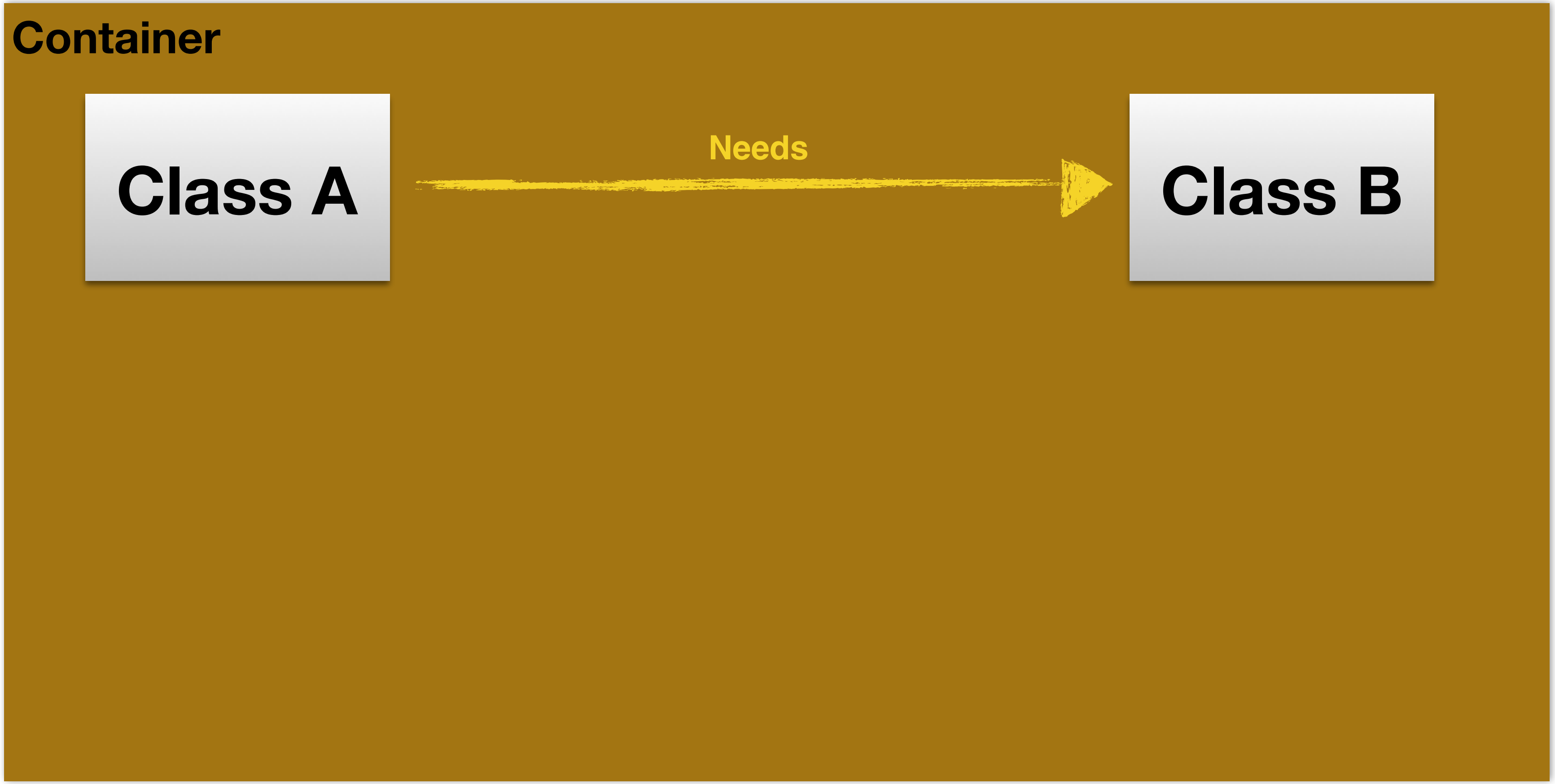
IoC - Dependency Injection



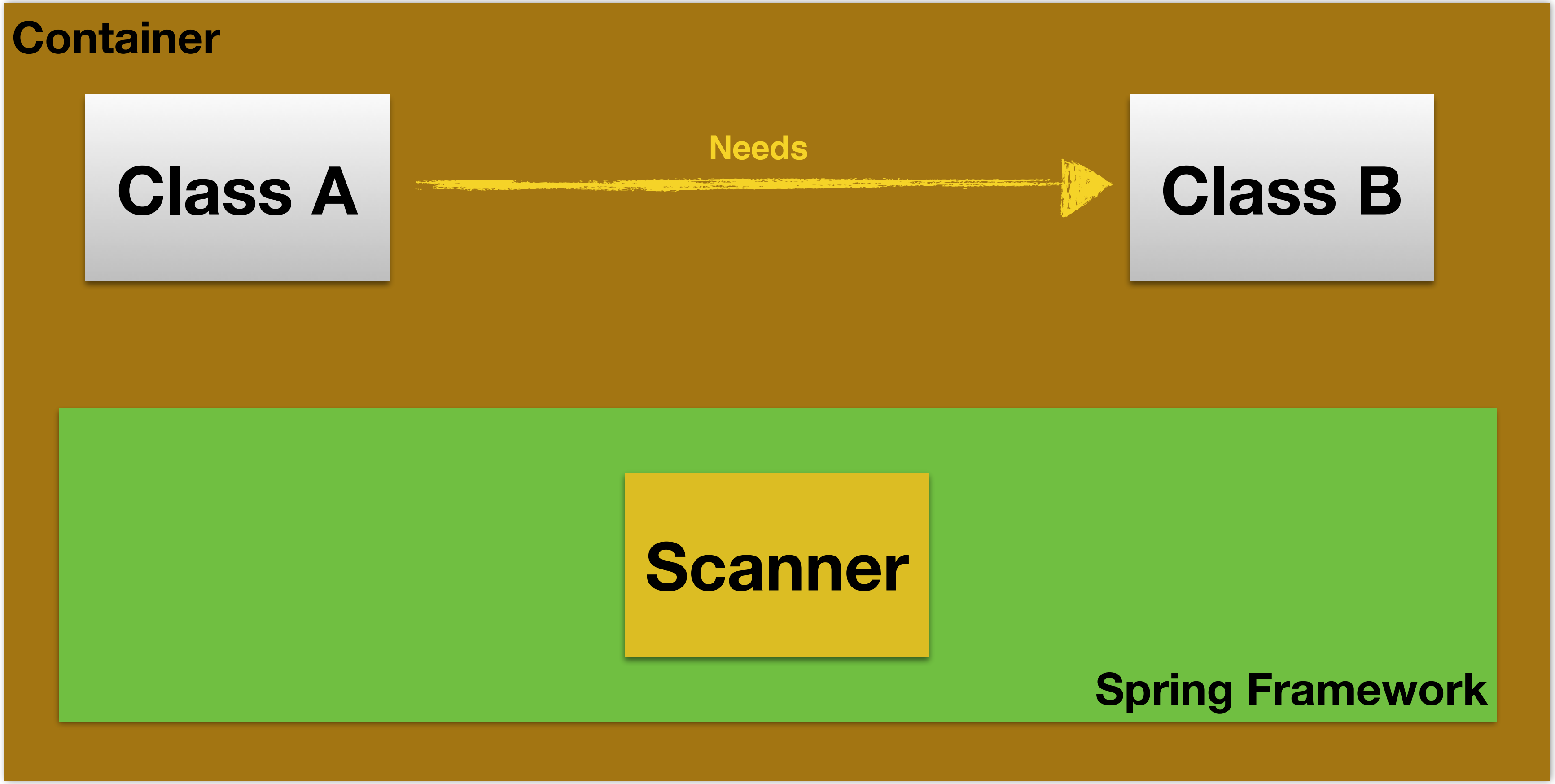
IoC - Dependency Injection



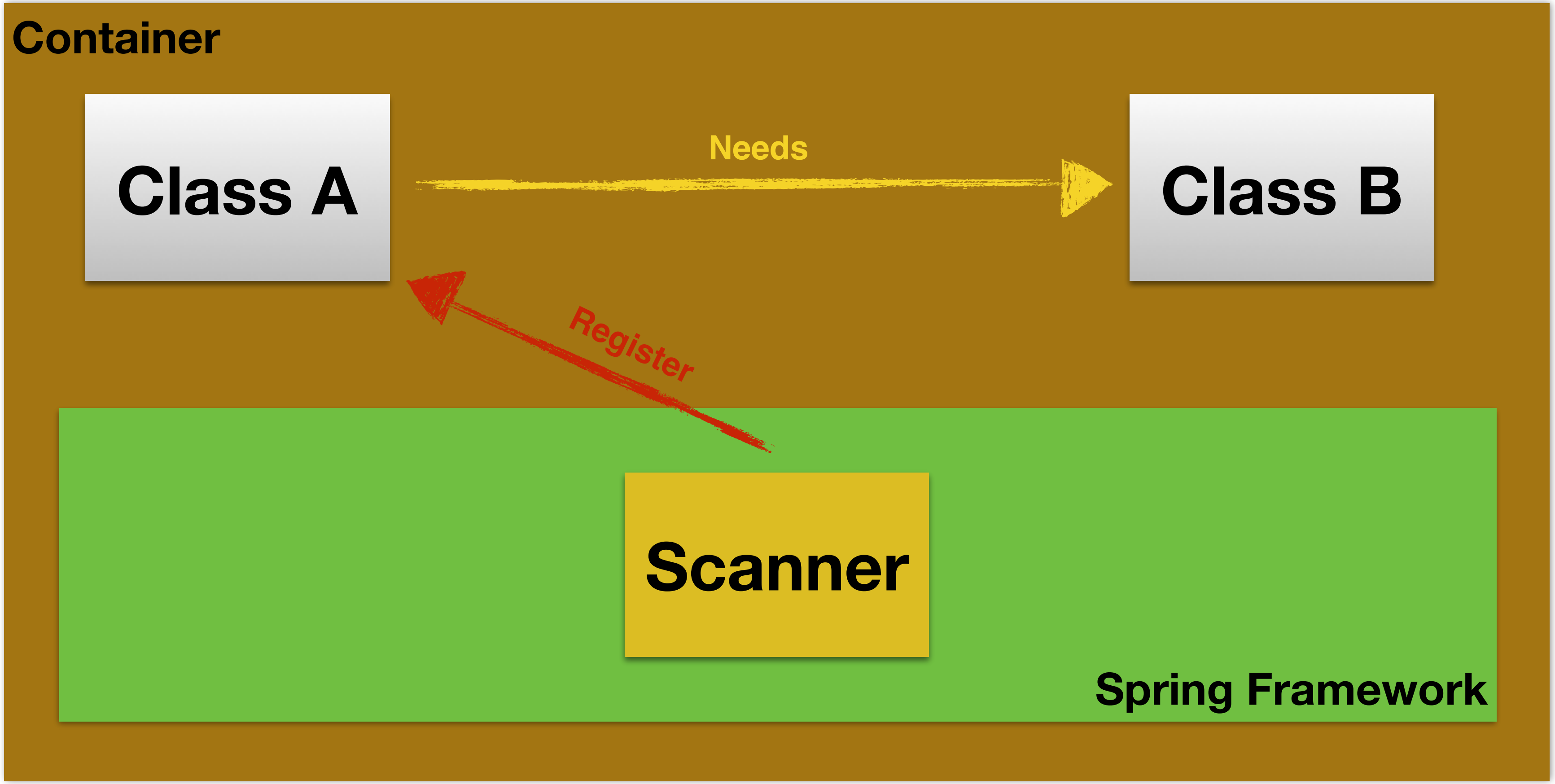
IoC - Dependency Injection



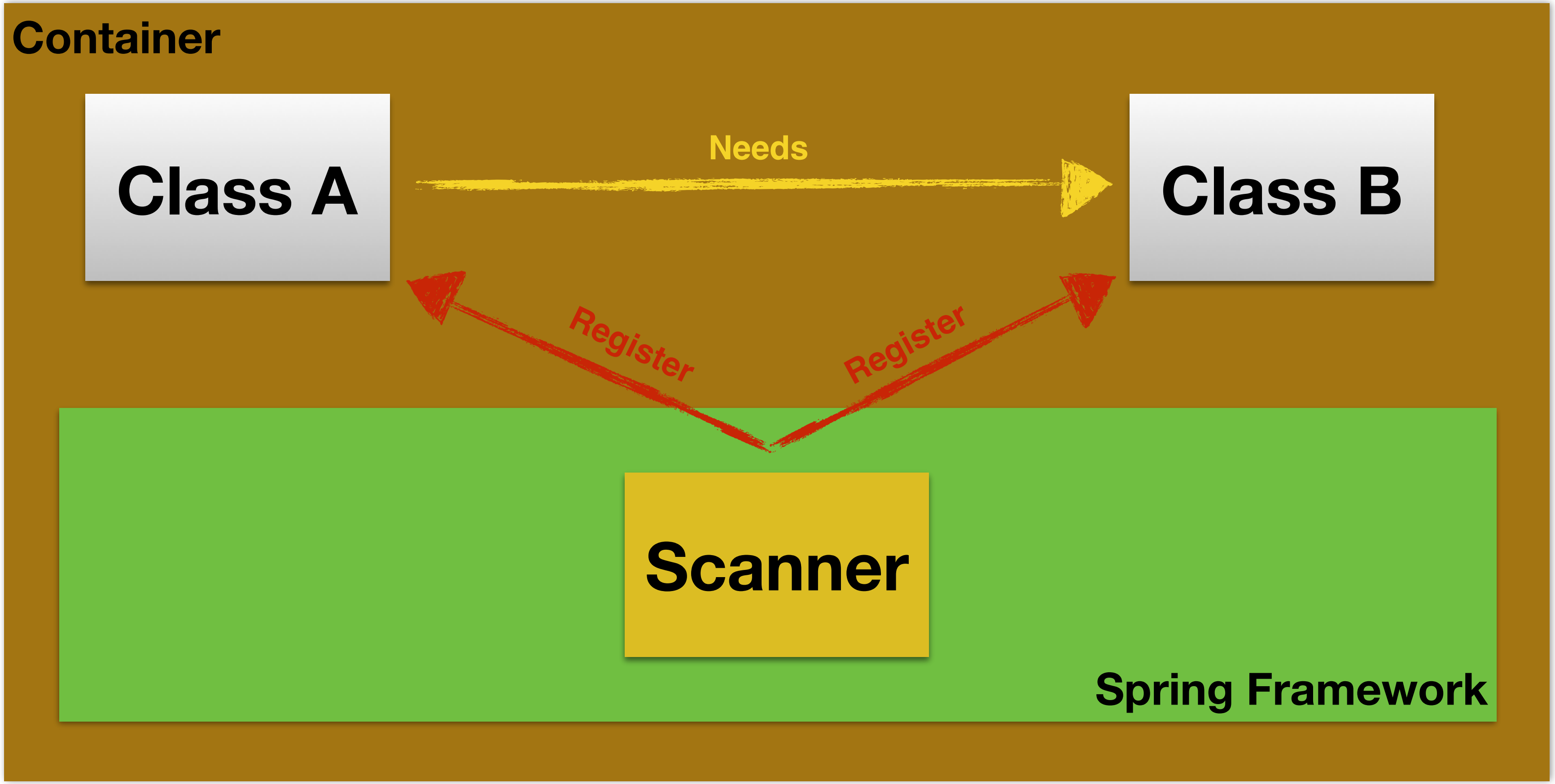
IoC - Dependency Injection



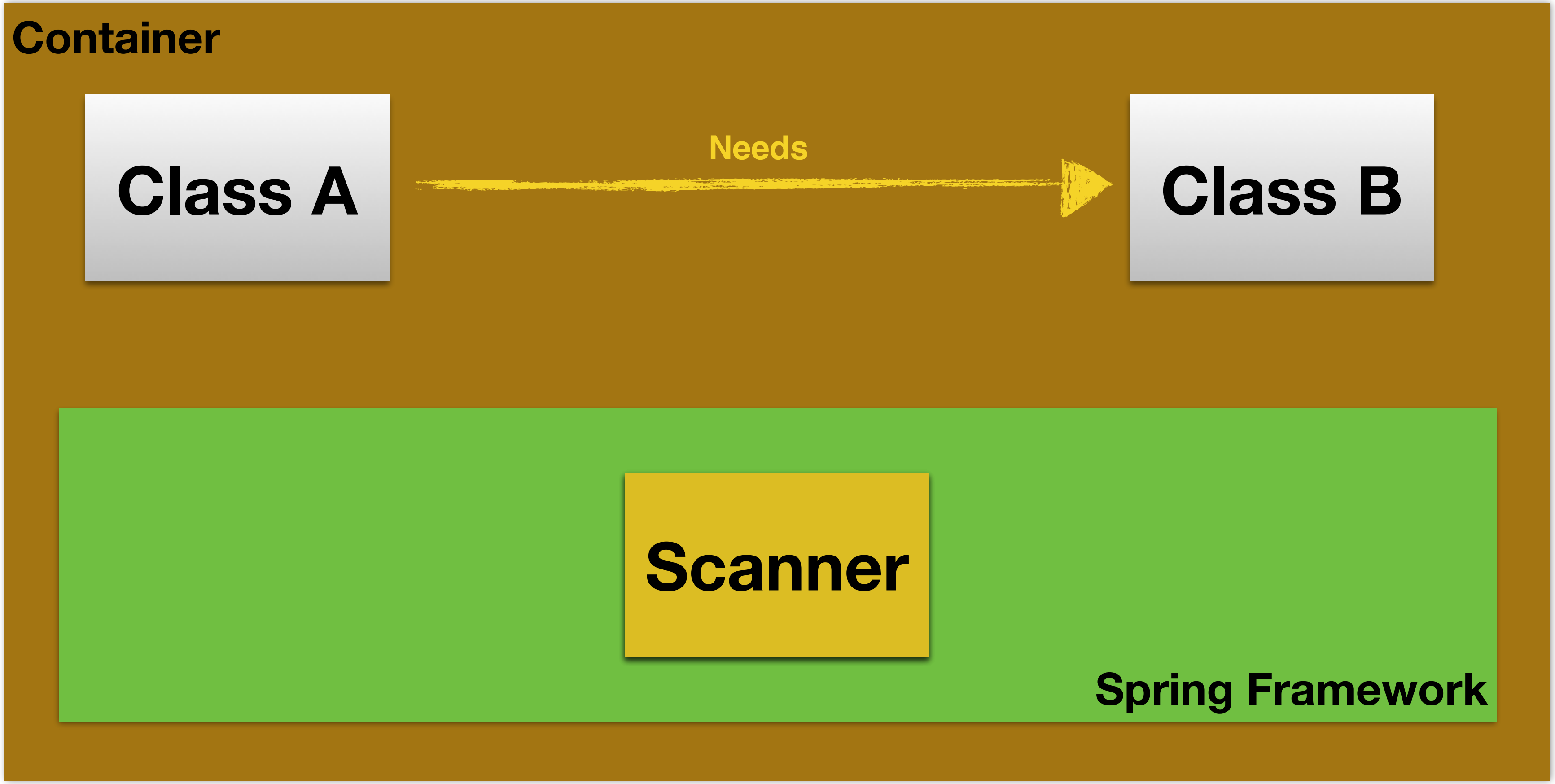
IoC - Dependency Injection



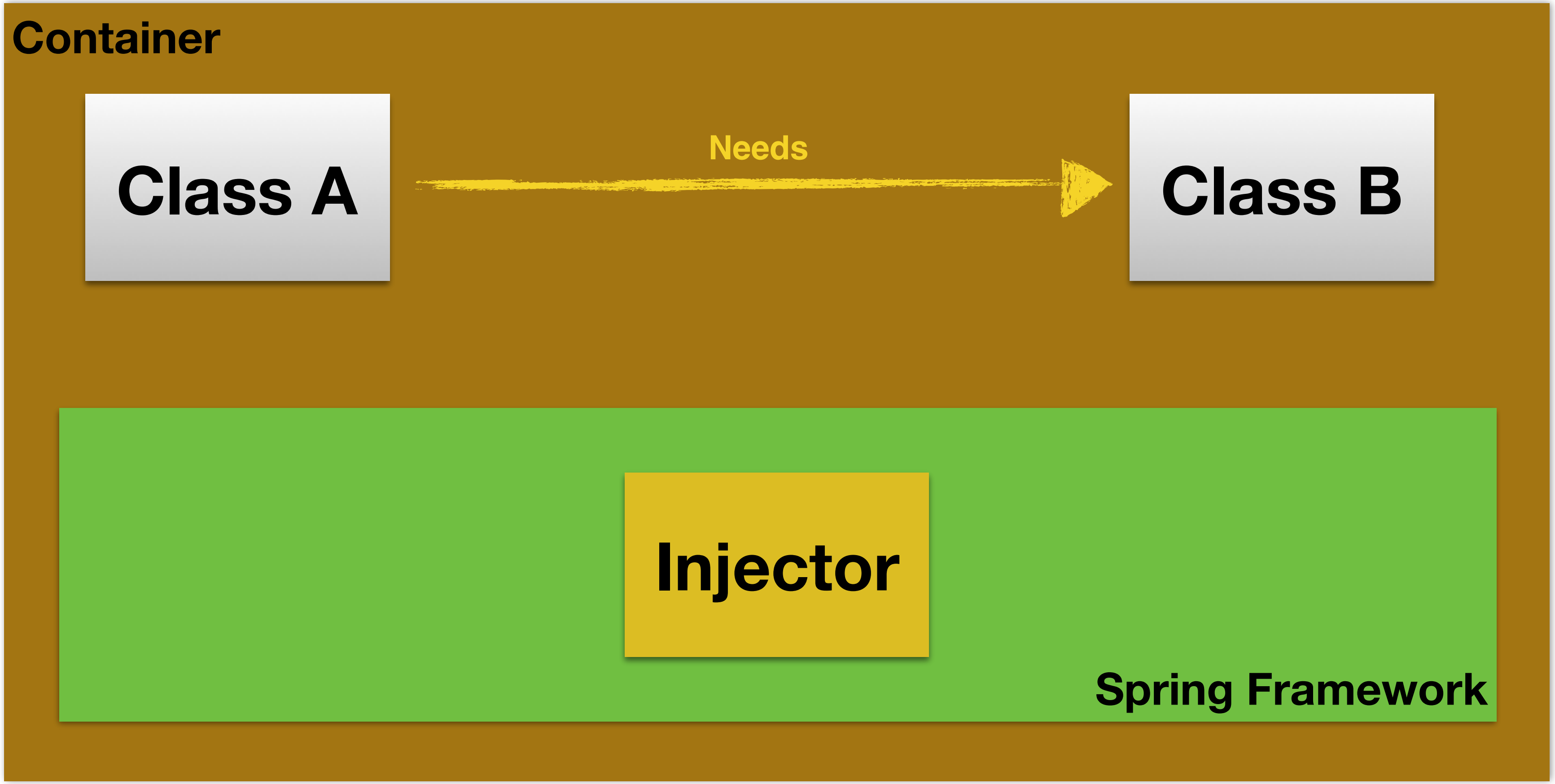
IoC - Dependency Injection



IoC - Dependency Injection



IoC - Dependency Injection



Container

Class A

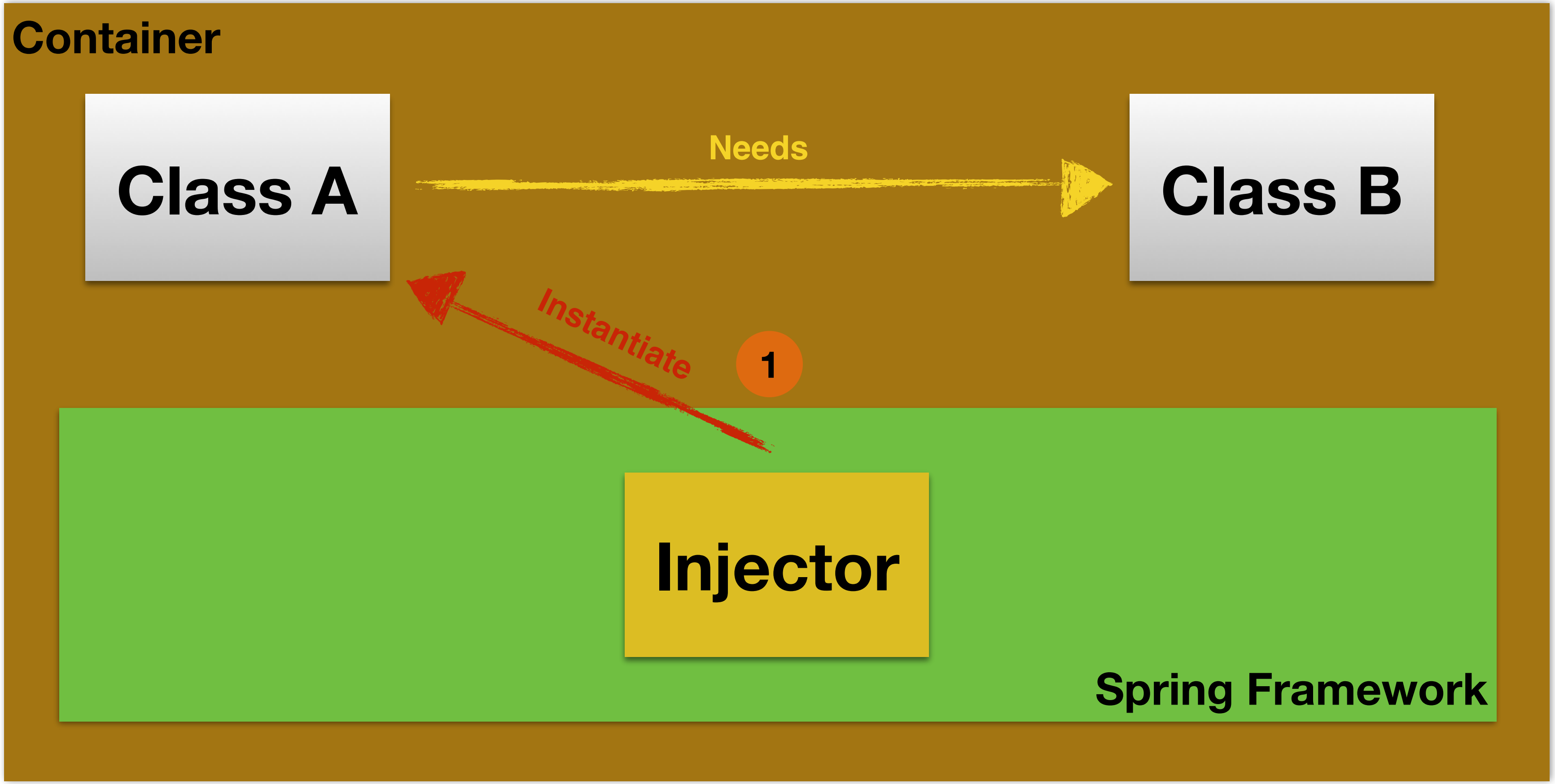
Needs

Class B

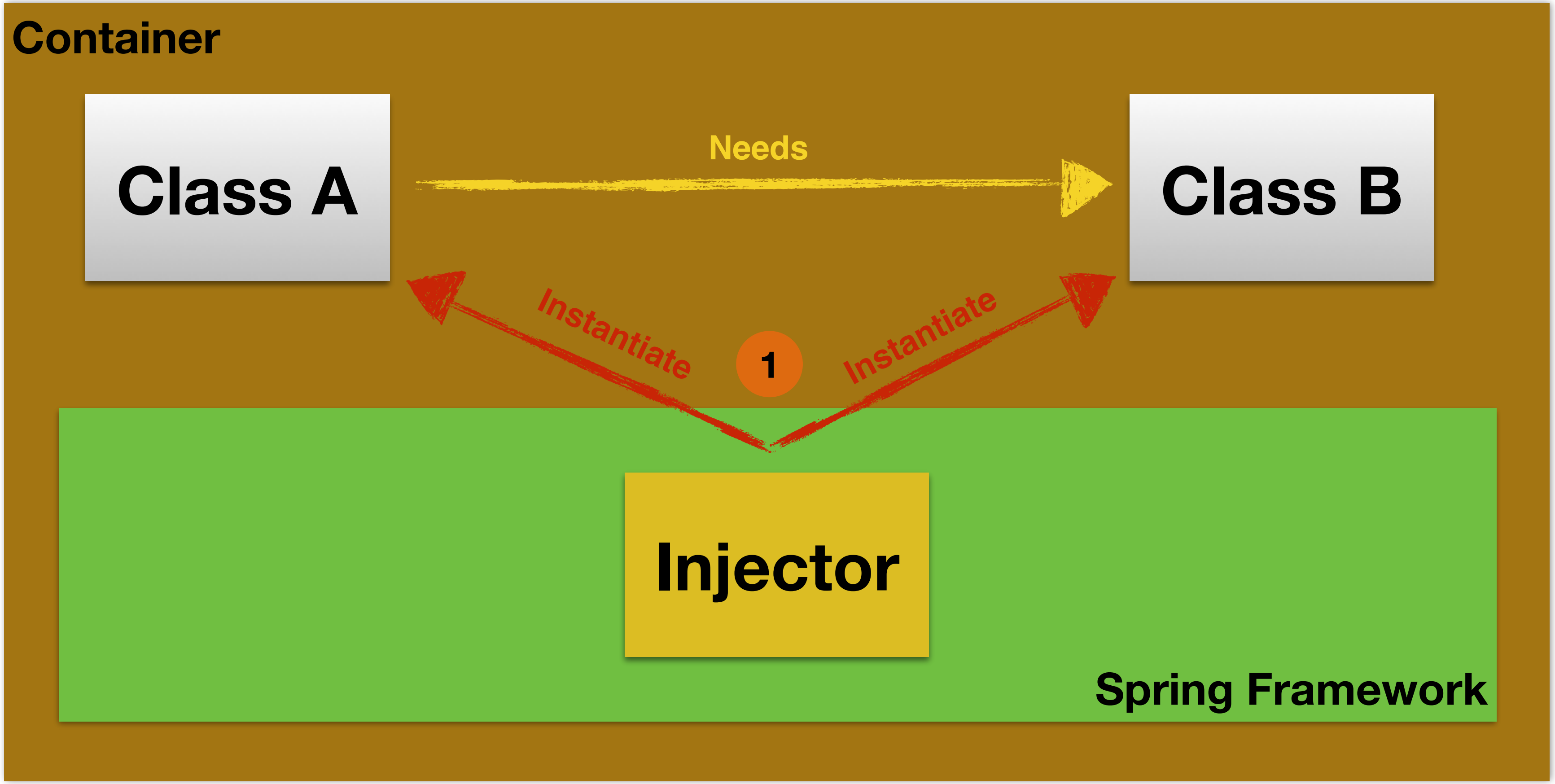
Injector

Spring Framework

IoC - Dependency Injection



IoC - Dependency Injection



Container

Class A

Needs

Class B

Instantiate

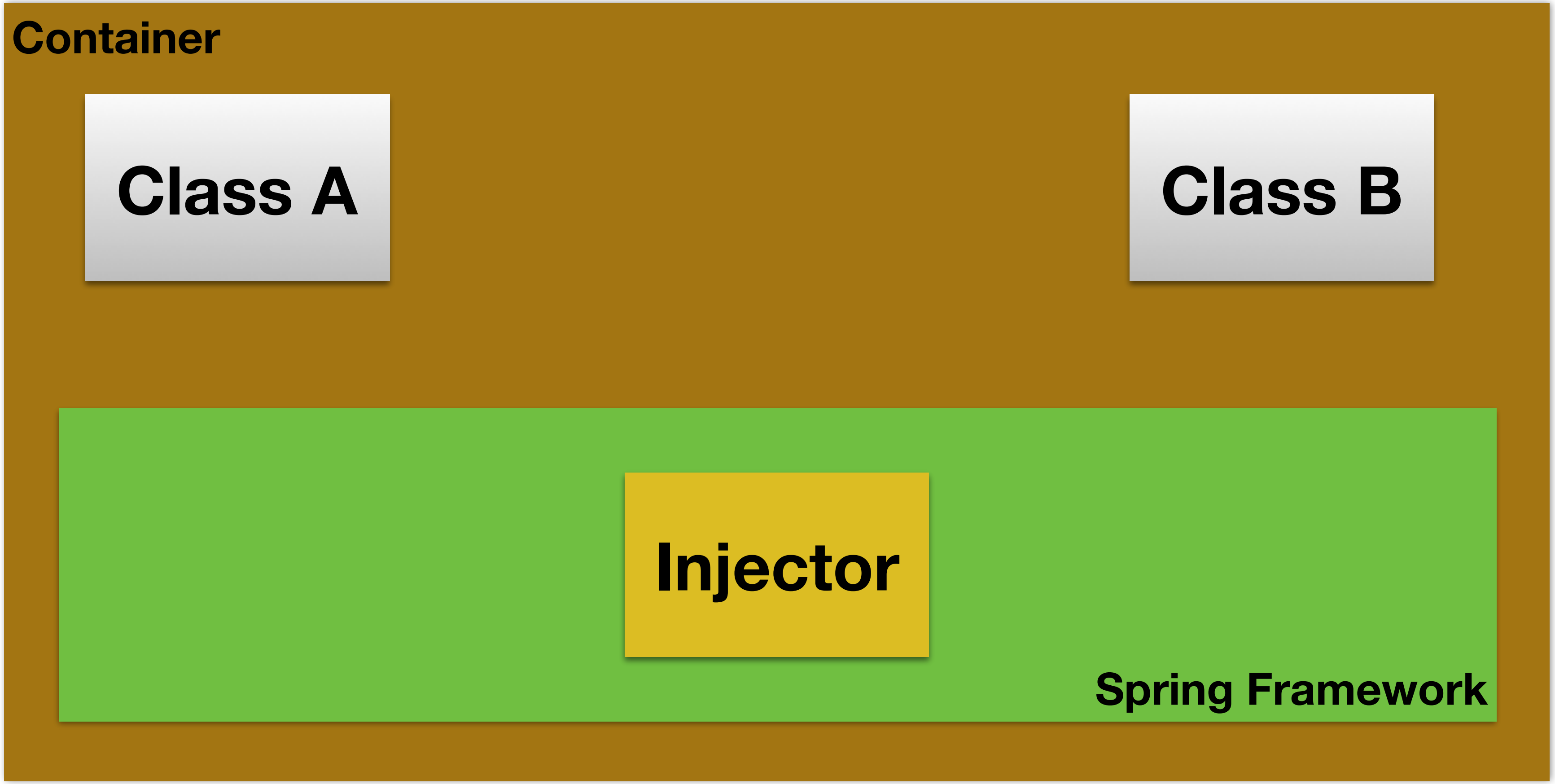
1

Instantiate

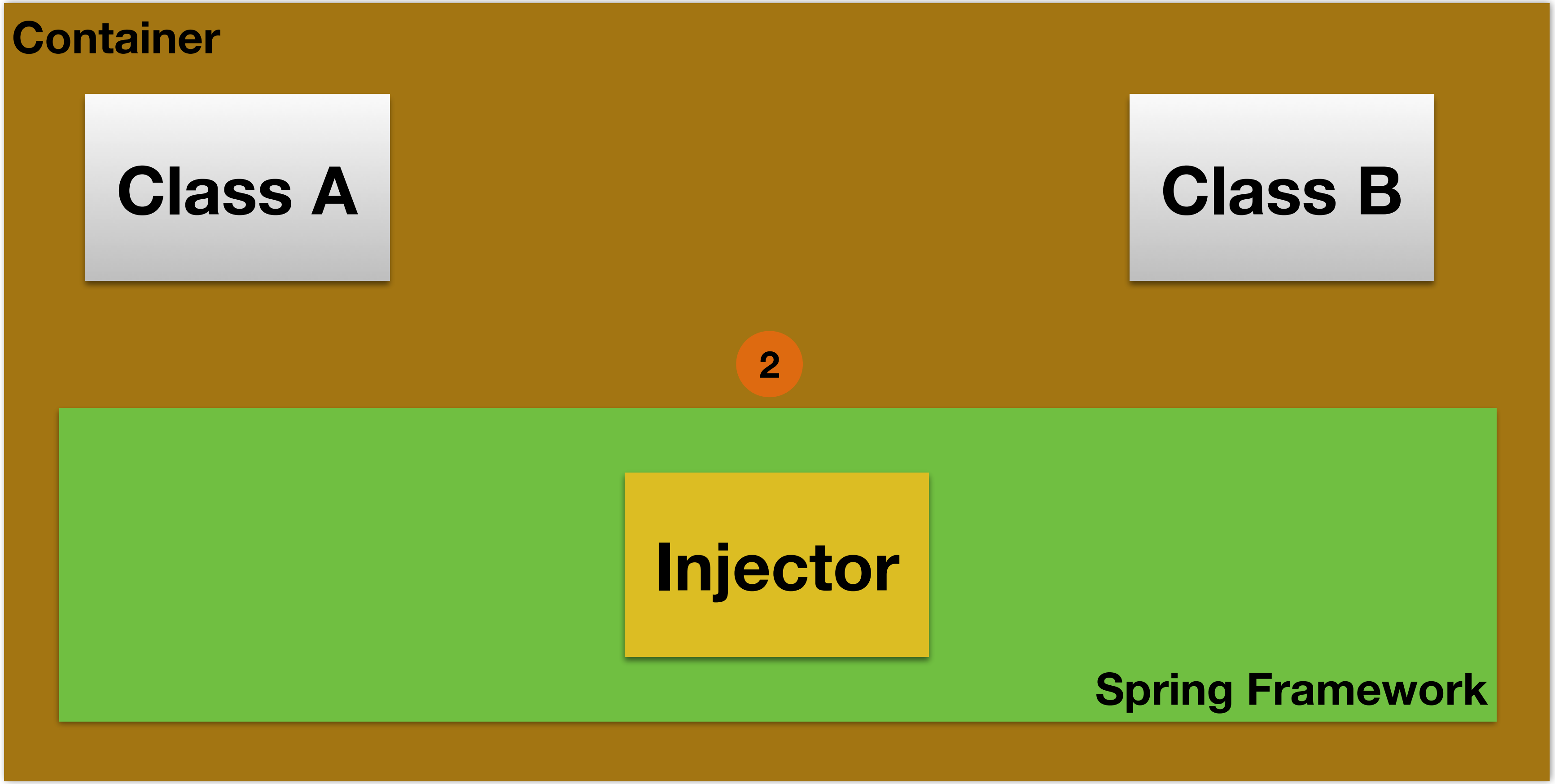
Injector

Spring Framework

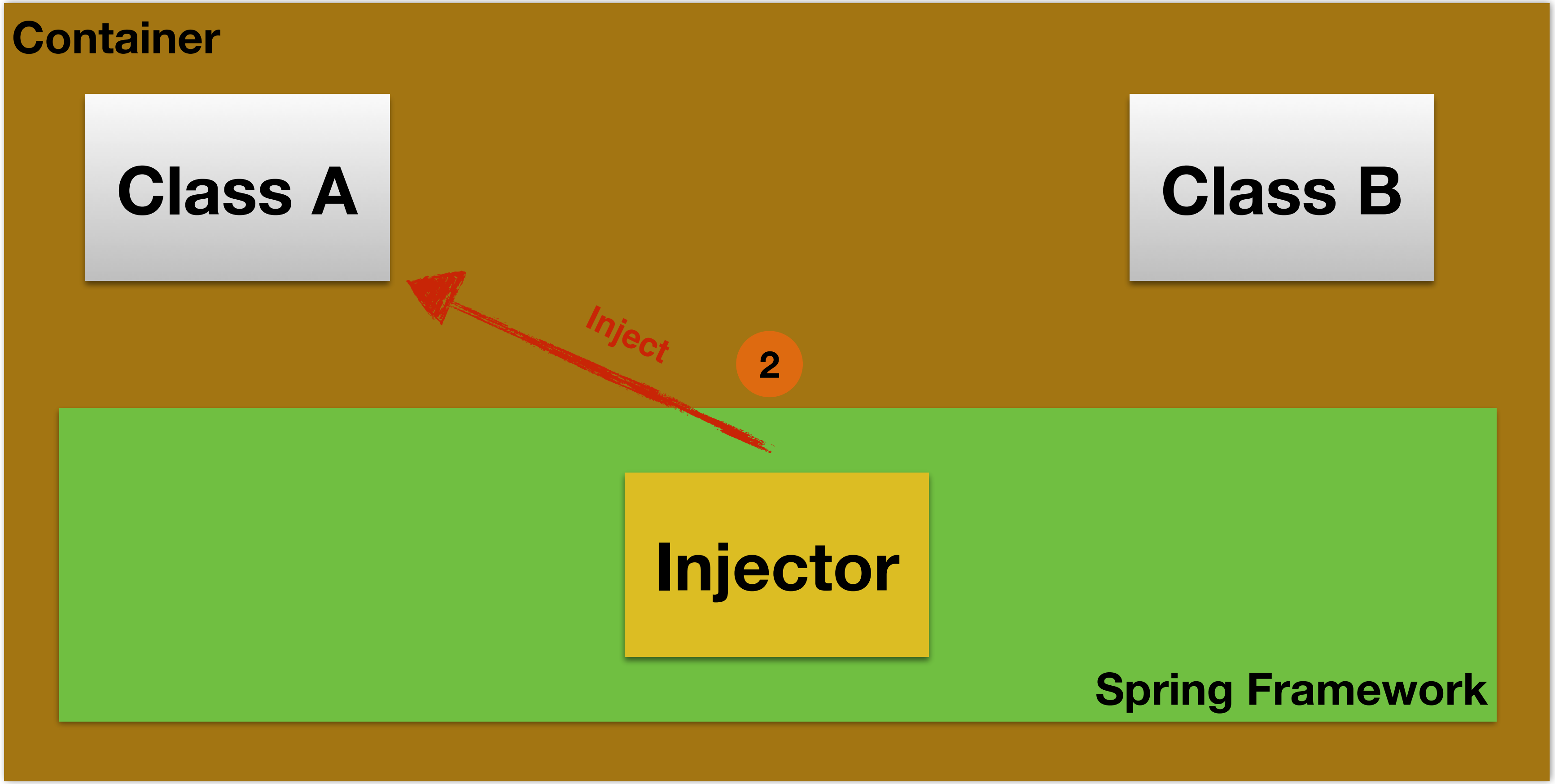
IoC - Dependency Injection



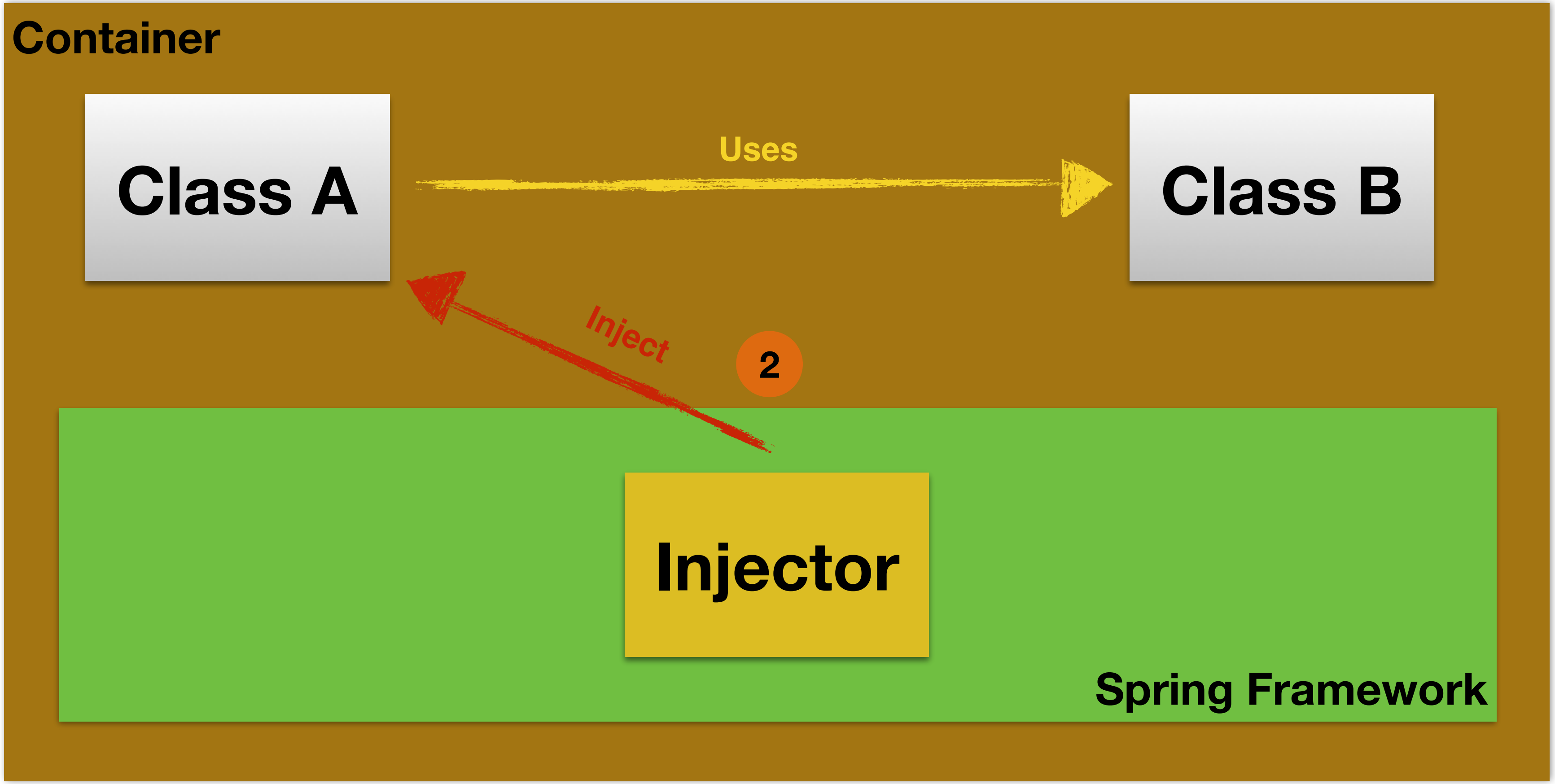
IoC - Dependency Injection



IoC - Dependency Injection



IoC - Dependency Injection



Dependency Injection in code with SF

```
package ch.heigvd.ptl.sample.ioc.sf;

import org.springframework.stereotype.Service;

@Service
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Dependency Injection in code with SF

```
package ch                      sf;

import org                      otype.Service;

@Service
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the framework that is a **service**.

Dependency Injection in code with SF

```
package ch.heigvd.ptl.sample.ioc.sf;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class ServiceAImpl {
    @Autowired
    private ServiceBImpl serviceB;

    public String hello() {
        return "Hello " + serviceB.world();
    }
}
```

```
package ch                        sf;

import org                        otype.Service;

@Service
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the framework that is a **service**.

Dependency Injection in code with SF

Tell the framework that is also a **service**.

```java
package ch.                           sf;

import org.                           factory.annotation.Autowired;
import org                           otype.Service;

@Service
public class ServiceAImpl {
    @Autowired
    private ServiceBImpl serviceB;

    public String hello() {
        return "Hello " + serviceB.world();
    }
}
```

Tell the framework that is a **service**.

```java
package ch                           sf;

import org                           otype.Service;

@Service
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Dependency Injection in code with SF

```
package ch.                    sf;

import org.                    factory.annotation.Autowired;
import org                     otype.Service;

@Service
public class ServiceAImpl {
    @Autowired
    private ServiceBImpl serviceB;

    public Stri
        return                         d();
    }
}
```

Tell the framework that is also a **service**.

Tell the framework that Service A requires an instance of Service B.

```
package ch                     sf;

import org                     otype.Service;

@Service
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the framework that is a **service**.

Dependency Injection in code with SF

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Tell the framework that is also a **service**.

```
package ch.                      sf;

import org.                      factory.annotation.Autowired;
import org                      otype.Service;

@Service
public class ServiceAImpl {
    @Autowired
    private ServiceBImpl serviceB;

    public Stri
        return                      d();
    }
}
```

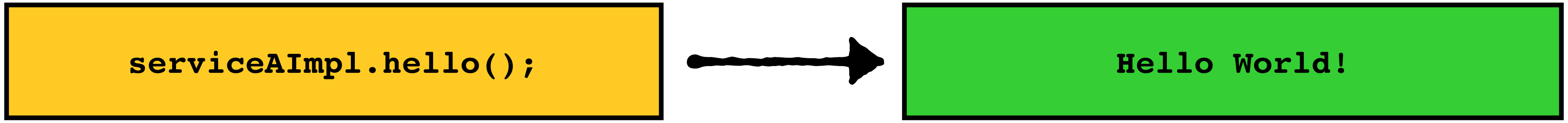
Tell the framework that Service A requires an instance of Service B.

Tell the framework that is a **service**.

```
package ch                      sf;

import org                      otype.Service;

@Service
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```



`serviceAImpl.hello();`

`Hello World!`

Dependency Injection in code for Java EE

```java
package ch.heigvd.ptl.sample.ioc.j2ee;

import javax.ejb.LocalBean;
import javax.ejb.Stateless;

@Stateless
@LocalBean
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Dependency Injection in code for Java EE

```
package ch.h                          c.j2ee;

import javax
import javax

@Stateless
@LocalBean
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the container that is a **local stateless** session bean.

Dependency Injection in code for Java EE

```java
package ch.heigvd.ptl.sample.ioc.j2ee;

import javax.ejb.EJB;
import javax.ejb.LocalBean;
import javax.ejb.Stateless;

@Stateless
@LocalBean
public class ServiceAImpl {
    @EJB
    private ServiceBImpl serviceB;

    public String hello() {
        return "Hello " + serviceB.world();
    }
}
```

```java
package ch.h                         c.j2ee;

import javax
import javax

@Stateless
@LocalBean
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the container that is a **local stateless** session bean.

Dependency Injection in code for Java EE

```
package ch.heigvd.ptl.sample.ioc.j2ee;

import java
import java
import java

@Stateless
@LocalBean
public class ServiceAImpl {
    @EJB
    private ServiceBImpl serviceB;

    public String hello() {
        return "Hello " + serviceB.world();
    }
}
```

Tell the container that is also a **local stateless** session bean.

```
package ch.h                              c.j2ee;

import javax
import javax

@Stateless
@LocalBean
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the container that is a **local stateless** session bean.

Dependency Injection in code for Java EE

```
package ch.heigvd.ptl.sample.ioc.j2ee;

import java
import java
import java

@Stateless
@LocalBean
public class ServiceAImpl {
    @EJB
    private ServiceBImpl serviceB;

    pu                          ) {
                              serviceB.world();
    }
}
```

Tell the container that is also a **local stateless** session bean.

Tell the container that Service A requires an instance of Service B.

```
package ch.h                              c.j2ee;

import javax
import javax

@Stateless
@LocalBean
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the container that is a **local stateless** session bean.

Dependency Injection in code for Java EE

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

```
package ch.heigvd.ptl.sample.ioc.j2ee;

import javax
import javax
import javax

@Stateless
@LocalBean
public class ServiceAImpl {
    @EJB
    private ServiceBImpl serviceB;

    pu                  ) {
                   serviceB.world();
    }
}
```

Tell the container that is also a **local stateless** session bean.

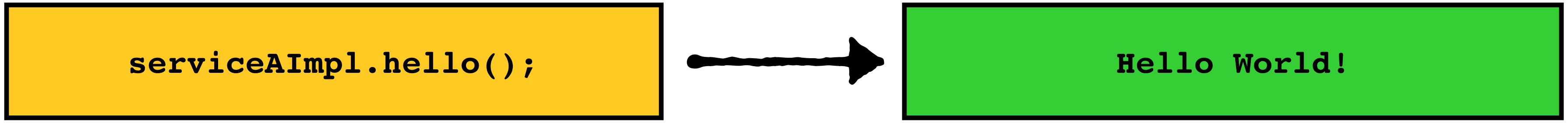
Tell the container that Service A requires an instance of Service B.

```
package ch.h                        c.j2ee;

import javax
import javax

@Stateless
@LocalBean
public class ServiceBImpl {
    public String world() {
        return "World!";
    }
}
```

Tell the container that is a **local stateless** session bean.



`serviceAImpl.hello();`

`Hello World!`

Why Spring Framework?

- Java EE need new ideas to enrich his specs
- Developers love alternatives
- Spring Source became a reference for various tools and libraries like Spring Security.
- Offers enrichment to develop web applications or enterprise applications.
- Allow integration with Java EE technologies (JPA, JMS, ...) in addition of the libraries used in the app.
- Can be deployed in application servers.
- Big community behind the various libraries that compose the Spring Framework.

Spring Framework vs. Java EE

- Are they concurrent or complementary?
  - No answer like: “Yes” or “No”
  - In fact, they play well together but that really depends what you are building.
- It’s really common to use Java EE and one or more libraries from Spring Framework (Ex: Spring Security, Spring MVC, ...)
- Big difference:
  - Java EE is a full package where you choose a profile type to develop your apps;
  - Spring Framework, you pick the libs you need to build what you need and no more.