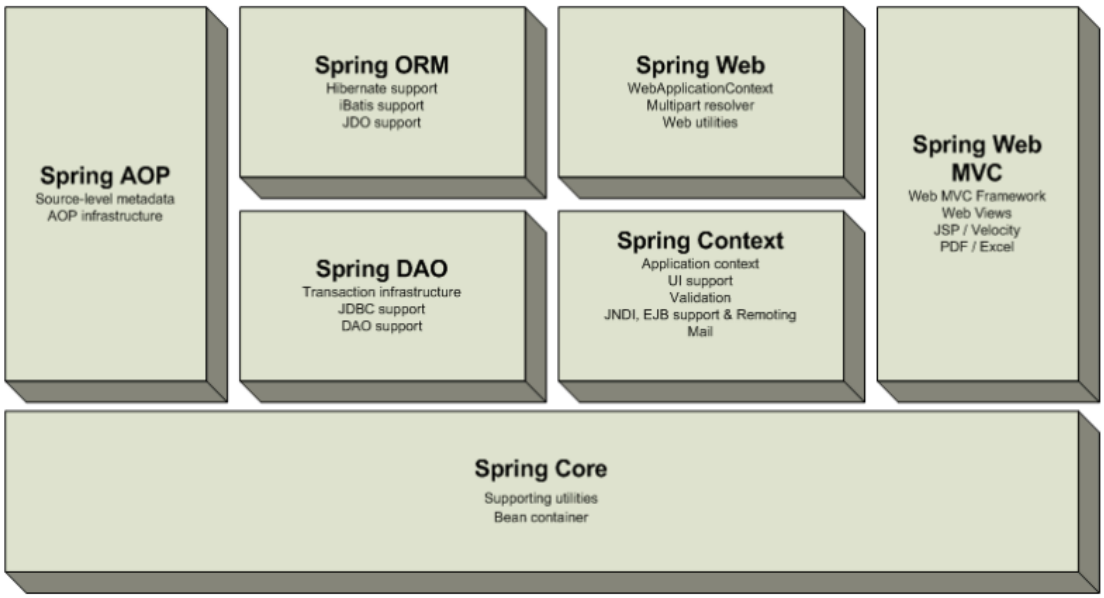
SpringFramework 개요

- **Enterprise Application에서 필요로 하는 기능을 제공하는 프레임워크**
- **JEE가 제공하는 다수의 기능을 지원하는 Lightweight Application Framework**



- **Spring Core** : Spring 프레임워크의 근간이 되는 IoC(또는 DI) 기능을 지원하는 영역을 담당하고 있다. BeanFactory를 기반으로 Bean 클래스들을 제어할 수 있는 기능을 지원한다.
- **Spring Context** : Spring Core 바로 위에 있으면서 Spring Core에서 지원하는 기능외에 추가적인 기능들과 좀 더 쉬운 개발이 가능하도록 지원하고 있다. 또한 JNDI, EJB등을 위한 Adaptor들을 포함하고 있다.
- **Spring DAO** : JDBC 기반하의 DAO개발을 좀 더 쉽고, 일관된 방법으로 개발하는 것이 가능하도록 지원하고 있다. Spring DAO를 이용할 경우 지금까지 개발하던 DAO보다 적은 코드와 쉬운 방법으로 DAO를 개발하는 것이 가능하다.
- **Spring ORM** : Object Relation Mapping 프레임워크인 Hibernate, IBatis, JDO와의 결합을 지원하기 위한 기능이다. Spring ORM을 이용할 경우 Hibernate, IBatis, JDO 프레임워크와 쉽게 통합하는 것이 가능하다.
- **Spring AOP** : Spring 프레임워크에 Aspect Oriented Programming을 지원하는 기능이다. 이 기능은 AOP Alliance 기반하에서 개발되었다.

- **Spring Web** : Web Application 개발에 필요한 Web Application Context와 Multipart Request등의 기능을 지원한다. 또한 Struts, Webwork와 같은 프레임워크의 통합을 지원하는 부분을 담당한다.
- **Spring Web MVC** : Spring 프레임워크에서 독립적으로 Web UI Layer에 Model-View-Controller를 지원하기 위한 기능이다. 지금까지 Struts, Webwork가 담당했던 기능들을 Spring Web MVC를 이용하여 대체하는 것이 가능하다. 또한 Velocity, Excel, PDF와 같은 다양한 UI 기술들을 사용하기 위한 API를 제공하고 있다.

DI(Dependency Injection)

- DI는「Dependency Injection」의 약어로 의존성 주입을 의미
- 객체 사이의 의존 관계가 자기 자신이 아닌 외부(스프링)에 의해서 설정
- 스프링은 각 클래스 간의 의존 관계를 관리하기 위한 2가지 방법을 제공

  ● Constructor Injection

  •의존하는 빈 객체를 컨테이너로부터 생성자의 파라미터를 통해서 전달받는 방식

  •클래스를 초기화 할 때 컨테이너로부터 의존 관계에 있는 특정 리소스인 빈 객체를 생성자를 통해서 할당 받는 방법

◈ Foo.java

```
public class Foo{
  private Bar bar;
  public Foo(Bar bar){
    this.bar = bar;
  }
}
```

◈ applicationContext.xml

```
〈bean id = "foo" class = "Foo" 〉
  〈constructor-arg〉
    〈ref bean = "bar" /〉
  〈/constructor-arg〉
〈/bean〉
〈bean id = "bar" class = "Bar" /〉
```

스프링의 설정에서 생성된 Bar 인스턴스를 <bean> 요소의 자식 요소인 <constructor-arg> 요소를 연결

● Setter Injection

•클래스 사이의 의존 관계를 연결시키기 위해서 setter 메소드를 이용하는 방법

◈ Foo.java

```
public class Foo{
  private Bar bar;
  public setBar(Bar bar){
    this.bar = bar;
  }
}
```

```
<bean id = "foo" class = "Foo" >
  <property name = "bar" ref = "bar" > </property> ····· ①
</bean>

<bean id = "bar" class = "Bar" /> ····· ②
```