7/7 일 오전

출처: goddaehee.tistory.com

스프링에 대해서

1.스프링이란? ->스프링 프레임워크

2.자바 엔터프라이즈 개발을 위한 오픈소스 애플리케이션 프레임워크

프레임워크?

➔ 개발할 대 설계 기본이 되는 뼈대나 구조 / 환경(문제 영역을 해결한 재사용, 확장 가능한 라이브러리.)

다음 개념(3,4)은 DI, AOP 등 스프링을 좀 더 이해 한 후 생각하기.

3.종속 객체를 생성 해주고, 조립할 수 있는 프레임 워크

4.자바 SE 로 된 자바 객체(POJO)를 자바 EE 에 의존적이지 않게 연결해준다.

POJO -> Plain Old Java Object)란 단순히 평범한 자바빈즈(Javabeans) 객체를 의미한다.

5.우리나라의 공공기관의 웹 서비스 개발 시 사용을 권장하고 있는 "전자정부 표준프레임워크"의 기반 기술로서 쓰이고 있다.

스프링 특징

1."경량 컨테이너" (크기와 부하의 측면)로서 자바 객체를 직접 관리

- 각각의 객체 생성, 소멸과 같은 라이프 사이클을 관리하며 스프링으로부터 필요한 객체를 얻어올 수 있다.

2.제어 역행(IoC : Inversion of Control)

- 애플리케이션의 느슨한 결합을 도모.
- 컨트롤의 제어권이 사용자가 아니라 프레임워크에 있어 필요에 따라 스프

링에서 사용자의 코드를 호출한다.

3.의존성 주입(DI : Dependency Injection)
- 각각의 계층이나 서비스들 간에 의존성이 존재할 경우 프레임워크가 서로 연결시켜준다.

4.관점지향 프로그래밍(AOP : Aspect-Oriented Programming)
- 트랜잭션이나 로깅, 보안과 같이 여러 모듈에서 공통적으로 사용하는 기능의 경우 해당 기능을 분리하여 관리 할 수 있다.

AOP 를 공부하려면 Filter, Interceptor, AOP 를 비교하면서 공부하면 이해가 더 빠를 것이다.

출처: http://goddaehee.tistory.com/154

5. 애플리케이션 객체의 생명 주기와 설정을 포함하고 관리한다는 점에서 일종의 "컨테이너"(Container)라고 할 수 있다.
- iBatis, myBatis 나 Hibernate 등 완성도가 높은 데이터 베이스처리 라이브러리와 연결할 수 있는 인터페이스를 제공한다.

6. 트랜잭션 관리 프레임워크
- 추상화된 트랜잭션 관리를 지원하며 설정파일(xml, java, property 등)을 이용한 선언적인 방식 및 프로그래밍을 통한 방식을 모두 지원한다.

7. 모델-뷰-컨트롤러 패턴
- 웹 프로그램밍 개발 시 거의 표준적인 방식인 "Spring MVC"라 불리는 모델-뷰-컨트롤러(MVC) 패턴을 사용한다.

- DispatcherServlet 이 Controller 역할을 담당하여 각종 요청을 적절한 서비스에 분산시켜주며 이를 각 서비스들이 처리를 하여 결과를 생성하고 그 결과는 다양한 형식의 View 서비스들로 화면에 표시될 수 있다.

8.배치 프레임워크

- 스프링은 특정 시간대에 실행하거나 대용량의 자료를 처리하는데 쓰이는 일괄 처리(Batch Processing)을 지원하는 배치 프레임워크를 제공한다. 기본적으로 스프링 배치는 Quartz 기반으로 동작한다.

9.즉 공통 부분의 소스 코딩이 용이하며 확장성도 매우 높다.

스프링 모듈

1.Spring Core
- Spring 프레임워크의 근간이 되는 요소. IoC(또는 DI) 긴으을 지원하는 영역을 담당.
- BeanFactory 를 기반으로 Bean 클래스들을 제어할 수 있는 기능을 지원

2.Spring Context
- Spring Core 바로 위에 있으면서 Spring Core 에서 지원하는 기능외에 추가적인 기능들과 좀 더 쉬운 개발이 가능하도록 지원
- 또한 JNDI, EJB 등을 위한 Adapter 들을 포함

3.Spring DAO
- 지금까지 우리들이 일반적으로 많이 사용해왔던 JDBC 기반하의 DAO 개발을 좀 더 쉽고, 일관된 방법으로 개발하는 것이 가능하도록 지원
- Spring DAO 를 이용할 경우 지금까지 개발하던 DAO 보다 적은 코드와 쉬운 방법으로 DAO 를 개발하는 것이 가능하다.

4.Spring ORM
- Object Relation Mapping 프레임워크인 Hibernate, IBatis, JDO 와의 결합을 지원하기 위한 기능

- Spring ORM 을 이용할 경우 Hibernate, IBatis, JDO 프레임워크와 쉽게 통합하는 것이 가능

5.Spring AOP

- Spring 프레임워크에 Aspect Oriented Programming 을 지원하는 기능이다. 이 기능은 AOP Alliance 기반하에서 개발

6.Spring Web

- Web Application 개발에 필요한 Web Application Context 와 Multipart Request 등의 기능을 지원

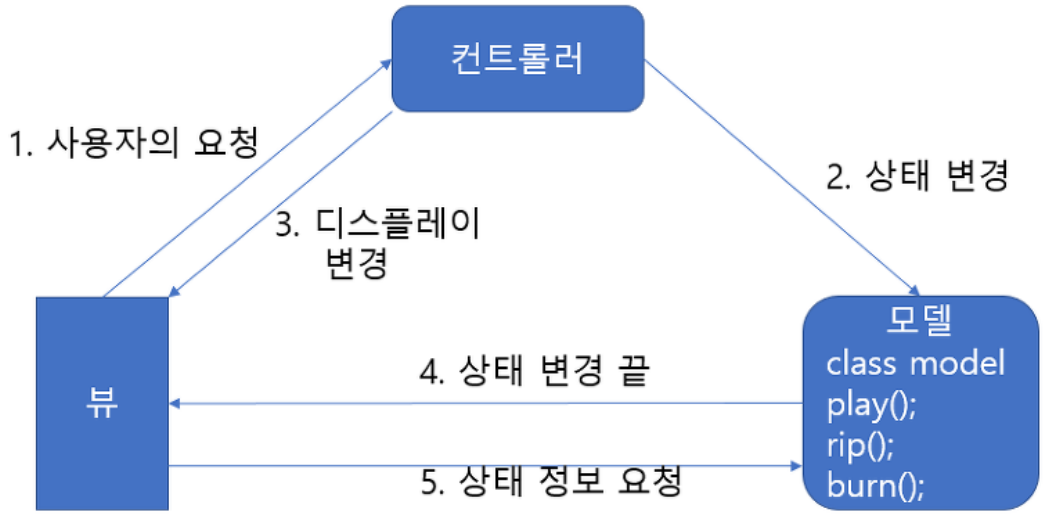
- 또한 Struts, Webwork 와 같은 프레임워크의 통합을 지원하는 부분을 담당

7.Spring Web MVC

- Spring 프레임워크에서 독립적으로 Web UI Layer 에 Model-View-Controller 를 지원하기 위한 기능

지금까지 Struts, Webwork 가 담당했던 기능들을 Spring Web MVC 를 이용하여 대체하는 것이 가능하다. 또한 Velocity, Excel, PDF 와 같은 다양한 UI 기술들을 사용하기 위한 API 를 제공

MVC 패턴



(1) View

- 모델을 표현하는 방법을 제공하는 사용자 인터페이스.
  일반적으로 화면에 표시하기 위해 필요한 상태 및 데이터를 모델에서 직접 가져온다.

(2) Model

- 모든 데이터, 상태 및 어플리케이션 로직이 들어있다.
  뷰와 컨트롤러에서 모델의 상태를 조작하거나 가져오기 위한 인터페이스를 제공하고
  모델에서 자신의 상태 변화에 대해서 옵저버들에게 알려주긴 하지만
  기본적으로 모델은 뷰와 컨트롤러에 별 관심이 없다.

(3) Controller

- 뷰와 모델 사이에서 위치하며 사용자로부터 입력을 받아서 그것이 모델에게 어떤 의미가 있는지 파악한다.
  단순히 모델한테 전달하는 역할만 하는것이 아니라, 사용자가 입력한 것을 해석해서
  그것을 바탕으로 모델을 조작하는 임무를 맡고있다.

- 뷰에서 컨트롤러의 역할을 직접 맡아도 되지만, 그렇게 하지 않는 것이 좋다.
  첫 번째로 뷰가 두 가지 역할을 맡기 때문에 코드가 복잡해지고 유지보수에 좋지 않다.
  두 번째로 뷰를 모델에 너무 밀접하게 연관시켜야 한다는 문제가 있다.
  때문에 뷰를 다른 모델하고 연결해서 재사용하기가 어려워진다.

- 컨트롤러는 뷰와 모델의 결합을 끊어주는 역할을 한다.

뷰와 컨트롤러를 느슨하게 결합시켜 놓으면 더 유연하고 확장하기 좋은 디자인을 만들 수 있다.

*** 사용자는 뷰하고만 접촉할 수 있다.**

- 뷰는 모델을 보여주는 창이라고 할 수 있다.
  재생 버튼을 누른다든가 하는 식으로 뷰에 대해서 어떠한 행동을 하면
  뷰에서 컨트롤러한테 사용자가 어떤 일을 했는지 알려준다.

*** 컨트롤러에서 모델한테 상태를 변경하라는 요청을 한다.**

- 컨트롤러에서는 사용자의 행동을 받아서 해석한다.
  사용자가 버튼을 클릭한다든가 하는 어떤 행동을 하면 컨트롤러에서는 그것이 무엇을 의미하는지 분석하고
  그 행동에 따라서 모델을 어떤식으로 조작해야 하는지 결정한다.

*** 컨트롤러에서 뷰를 변경해달라고 요청할 수 있다.**

- 컨트롤러에서 뷰로부터 어떤 행동을 받았을 때, 그 행동의 결과로 뷰를 바꿔달라고 할 수 있다.
  예를 들어, 컨트롤러에서 인터페이스에 있는 어떤 버튼이나 메뉴를 활성화 또는 비활성화 시킬 수 있다.

*** 상태가 변경되면 모델에서 뷰한테 그 사실을 알린다.**

- 사용자의 행동 때문이든 다른 내부적인 변화 때문이든, 모델에서 무언가가 변경되면
  모델에서 뷰한테 상태가 변경되었음을 알린다.

*** 뷰에서 모델한테 상태를 요청한다.**

- 뷰에서 화면에 표시할 상태는 모델로부터 직접 가져온다.
  예를 들어, 모델에서 뷰한테 새로운 곡이 재생되기 시작했다고 알려주면
  뷰에서는 모델한테 곡 제목을 요청하고, 그것을 받아서 화면에 표시한다.
  컨트롤러에서 뷰를 바꾸라는 요청을 했을 때도 뷰에서 모델한테 상태를 알려달라고 요청할 수 있다.