**WebApplicationContext? ApplicationContext? ServletContext?**

## Context?

* Context 란 용어는 어떤 한글번역본에는 "문맥"이란 용어로 직역됩니다만 사실 적절한 의미전달은 안되고 있습니다.
* C++ 언어에서 보면, Print을 하기 위해 PrintContext 객체를 사용하고, Java Applet에서 Applet의 AppletContext를 얻기 위해 Applet.getAppletContext( )를 사용합니다.
* 결국 "Context"란 "어떤 객체를 핸들링하기 위한 접근 수단"의 의미를 가집니다.
* 위에서 해당 서블릿을 수행하려면 "서블릿 Context"를 획득해야 그것을 통해 그 서블릿을 수행할 수 있는 것이지요.
* 서블릿은 CGI와는 다르게 멀티쓰레드(Multi-thread)로 동작합니다.
* 한 순간에 하나의 수행만 있는 것이 아니라, 같은 서블릿이 여러 client의 호출에 대해 동시다발적으로 같이 수행되는 구조이지요.

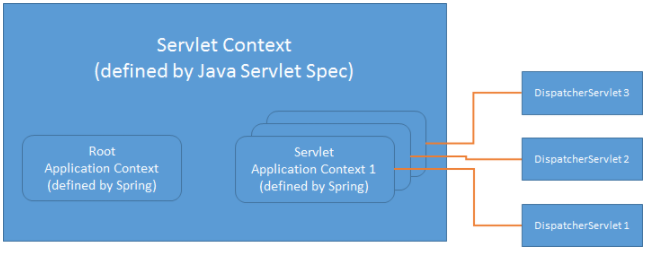
## ApplicationContext?

* **ApplicationContext**는 Spring에서 만든 **인터페이스**로 말그대로 애플리케이션에 대한 **context**를 가지고 있다.

## WebApplicationContext?

* **WebApplicationContext란 Spring의 ApplicationContext를 확장**한 **인터페이스**로, 웹 애플리케이션에서 필요한 몇 가지 기능을 추가한 인터페이스다.
* 예를 들면 **WebApplicationContext**의 구현체는 getServletContext라는 메소드를 통해 **ServletContext**를 얻을 수 있다.

## ServletContext?

* ServletContext는 Servlet API에서 제공하는 context로 모든 servlet이 공유하는 **context**이다.
* Spring Web MVC에서는 **ServletContext**가 **WebApplicationContext**를 가지고 있다.
* 아래의 그림과 같이 Serlvet Context가 WebApplication을 감싸고 있다.
* 아래에서 Root Applicaition Context와 Servlet Application Context 1~3 모두 WebApplicationContext이다.  
  

## ServletContext 클래스?

* **ServletContext 클래스는 톰캣 컨테이너 실행 시 각 컨텍스트(웹 애플리케이션)마다 한 개의 ServletContext 객체를 생성**합니다.
* **톰캣 컨테이너가 종료되면 ServletContext 객체 역시 소멸**됩니다.
* ServletContext 객체는 웹 애플리케이션이 실행되면서 애플리케이션 전체의 공통 자원이나 정보를 미리 바인딩(binding)해서 서블릿들이 공유하여 사용합니다.

### **ServletContext클래스의 특징**

* javax.servlet.ServletContext 로 정의 되어 있다.
* **서블릿과 컨테이너 간의 연동**을 위해 사용한다.
* 컨텍스트(웹 어플리케이션)마다 하나의 ServletContext가 생성된다.
* 서블릿끼리 자원(데이터)을 공유하는 데 사용한다.
* 컨테이너 실행시 생성되고 컨테이너 종료 시 소멸된다.

### ServletContext가 제공하는 기능

* 서블릿에서 파일 접근 기능
* 자원 바인딩 기능
* 로그 파일 기능
* 컨텍스트에서 제공하는 설정 정보 제공 기능

## 웹 애플리케이션 기본 구조

* 기본 구조

Web Application Name

│

└─────── WEB-INF

│

├─────── classes

│

├─────── lib

│

└─────── web.xml

* 웹 컨테이너(톰캣 등)에서 실행하는 웹 애플리케이션의 기본 디렉터리 구조
  + 이런 구조를 갖추지 않고 컨테이너에서 웹 어플리케이션 실행시 오류 발생
  + 기본구조 이외의 다른 기능이 추가되면 디렉터리를 추가해서 사용

## DispatcherServlet

* DispatcherServlet도 결국 Servlet이다.
* 따라서 다른 Servlet과 마찬가지로 자바 설정이나 web.xml에 있는 설정에 따라 정의되어야 하고 매핑되어야 한다.
* DispatcherServlet은 WebApplicationContext를 이용하여 자신을 설정한다.

## Context Hierarchy

* Context는 계층구조(부모 자식 관계, 상속 관계)를 가질 수 있다.
* 다시 말해 하나의 root WebApplicationContext 밑에 여러 개의 child WebApplicationContext를 갖는 것도 가능하다.