

Web.xml'siz Java Tabanlı ApplicationContext Konfigürasyonu



Web.xml'siz ApplicationContext Konfigürasyonu

- web.xml'siz ApplicationContext konfigürasyonu **Servlet 3.0+ spesifikasyonu** ile uyumlu uygulama sunucular için geçerlidir
- **WebApplicationInitializer** arayüzünü implement ederek web uygulamasında **ApplicationContext bootstrap işlemi tamamen web.xml'siz** gerçekleştirilebilir

WebApplicationInitializer

- Web uygulamalarında **ServletContext** nesnesini programatik olarak konfigüre etmek için kullanılan **callback interface**'tir
- Classpath'deki **WebApplicationInitializer** sınıflarını tespit edip yükleyen **SpringServletContainerInitializer** sınıfıdır
- SpringServletContainerInitializer sınıfı ise **javax.servlet.ServletContainerInitializer** SPI arayüzünden türer


SpringServletContainer Initializer

- ServletContainerInitializer implemantasyonları /META-INF/services/javax.servlet.ServletContainerInitializer dosyaları içerisinde bulunup yüklenir
- SpringServletContainerInitializer'da **spring-web.jar** içerisindeki ilgili dosya içerisinde tanımlıdır

Web.xml'siz ApplicationContext Konfigürasyonu

```
public class PetClinicWebApplicationInitializer
    extends AbstractContextLoaderInitializer {

    @Override
    protected WebApplicationContext createRootApplicationContext() {
        AnnotationConfigWebApplicationContext wac =
            new AnnotationConfigWebApplicationContext();
        wac.register(DaoBeansConfig.class, ServiceBeansConfig.class);
        wac.refresh();
        return wac;
    }
}
```



WebApplicationInitializer arayüzünden türer. ServletContext'de ContextLoaderListener'ın konfigürasyonunu sağlar

Birden fazla WebApplicationInitializer olması durumunda **Ordered** arayüzü ile aralarında sıralama yapmak mümkündür

Web.xml'siz DispatcherServlet Konfigürasyonu

```
public class PetClinicSpringMvcWebApplicationInitializer
    extends AbstractDispatcherServletInitializer {

    @Override
    protected WebApplicationContext createServletApplicationContext() {
        AnnotationConfigWebApplicationContext wac =
            new AnnotationConfigWebApplicationContext();
        wac.setEnvironment(new StandardServletEnvironment());
        wac.register(ControllerBeansConfig.class);
        return wac;
    }

    @Override
    protected String[] getServletMappings() {
        return new String[]{"/mvc/*"};
    }

    @Override
    protected WebApplicationContext createRootApplicationContext() {
        AnnotationConfigWebApplicationContext wac =
            new AnnotationConfigWebApplicationContext();
        wac.register(DaoBeansConfig.class, ServiceBeansConfig.class);
        return wac;
    }
}
```

Web.xml'siz DispatcherServlet Konfigürasyonu

- DispatcherServlet ayrı bir WAC oluşturmak yerine parent WAC'ı da kullanabilir

```
public class PetClinicSpringMvcWebApplicationInitializer
    extends AbstractDispatcherServletInitializer {

    private AnnotationConfigWebApplicationContext wac;

    @Override
    protected WebApplicationContext createRootApplicationContext() {
        wac = new AnnotationConfigWebApplicationContext();
        wac.register(DaoBeansConfig.class, ServiceBeansConfig.class,
            ControllerBeansConfig.class);
        return wac;
    }

    @Override
    protected WebApplicationContext createServletApplicationContext() {
        return wac;
    }

    @Override
    protected String[] getServletMappings() {
        return new String[]{"/mvc/*"};
    }
}
```

İletişim

- **Harezmi** Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- <http://www.java-egitimleri.com>
- info@java-egitimleri.com

