Configurare la DispatcherServlet

Cosa imparerai

- Come configurare la web application per utilizzare la DispatcherServlet





Configurare la DispatcherServlet

A partire dalle Servlet API 3.0, è possibile utilizzare il **web.xml** o il **codice Java** per configurare una servlet.

Pertanto, possiamo utilizzare una delle due notazioni per configurare la DispatcherServlet.

Configurazione XML nel web.xml

```
<web-app>
   <listener>
      <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
   </listener>

   <context-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>/WEB-INF/my-context.xml</param-value>
   </context-param>

   <servlet>
      <servlet-name>myapp</servlet-name>
      <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
      <init-param>
         <param-name>contextConfigLocation</param-name>
         <param-value></param-value>
      </init-param>
      <load-on-startup>1</load-on-startup>
   </servlet>

   <servlet-mapping>
      <servlet-name>myapp</servlet-name>
      <url-pattern>/*</url-pattern>
   </servlet-mapping>
</web-app>
```

Configurazione tramite codice Java

```
import org.springframework.web.context.WebApplicationContext;
import org.springframework.web.servlet.DispatcherServlet;
import org.springframework.web.servlet.FrameworkServlet;
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class WebAppInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {
   @Override
   protected Class<?>[] getServletConfigClasses() {
       return new Class<?>[] { WebConfig.class };
   }

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

   @Override
   protected FrameworkServlet createDispatcherServlet(WebApplicationContext servletAppContext) {
      DispatcherServlet ds = new DispatcherServlet(servletAppContext);
      ds.setThrowExceptionIfNoHandlerFound(true);
      return ds;
   }

   …
}
```

paolopreite

Configurare la DispatcherServlet

Per configurare la DispatcherServlet via codice Java è necessario seguire questi passi:

- Si crea una classe che estende la classe astratta **AbstractAnnotationConfigDispatcherServletInitializer (package org.springframework.web.servlet.support)**. Questa classe implementa l'interfaccia *WebApplicationInitializer* messa a disposizione da Spring MVC che garantisce che la classe create venga rilevata e utilizzata per inizializzare qualsiasi Servlet container versione 3;
- Nella classe si implementano i metodi richiesti dalla classe astratta e si fa l'override del metod *createDispatcherServlet(…)*;

Associamo la classe che contiene le configurazioni

Definiamo il path al quale risponderà la DispatcherServlet

Creiamo un oggetto di tipo DispatcherServlet e lo passiamo come parametro di ritorno al metodo.
NOTA: se impostiamo a true il setThrowExceptionIfNoHandlerFound(true), la web app genererà un'eccezione NoHandlerFoundException al posto dell'errore 404

```
public class WebAppInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {
   @Override
   protected Class<?>[] getServletConfigClasses() {
      return new Class<?>[] { WebConfig.class };
   }

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

   @Override
   protected FrameworkServlet createDispatcherServlet(WebApplicationContext servletAppContext) {
      DispatcherServlet ds = new DispatcherServlet(servletAppContext);
      ds.setThrowExceptionIfNoHandlerFound(true);
      return ds;
   }

   …
}
```

paolopreite

Configurare la DispatcherServlet

NOTA: se usiamo la notazione XML, possiamo usare la classe astratta **AbstractDispatcherServletInitializer** (al possto di AbstractAnnotationConfigDispatcherServletInitializer) per creare l'oggetto che definirà la DispatcherServlet.

NOTA: queste classi astratte hanno due metodi interessanti per definire i context:

- *getRootConfigClasses()* utilizzato per definire un application context che contiene configurazioni che non riguardano la componente web dell'applicazione.
- *getServletConfigClasses()* utilizzato per definire il context che conterrà le configurazioni di Spring MVC e verrà passato come argomento alla DispatcherServlet.

paolopreite

Di cosa abbiamo parlato in questa lezione

- Come configurare la web application per utilizzare la DispatcherServlet