Flash Back to XML config (the old way)

File: web.xml

```
<web-app>









</web-app>
```

Flash Back to XML config (the old way)

File: web.xml

Just an FYI

```
<web-app>









</web-app>
```

Flash Back to XML config (the old way)

Just an FYI

File: web.xml

```
<web-app>

    <servlet>
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-mvc-demo-servlet.xml</param-value>
        </init-param>

        <load-on-startup>1</load-on-startup>
    </servlet>









</web-app>
```

Flash Back to XML config (the old way)

File: web.xml

Just an FYI

```
<web-app>

    <servlet>
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-mvc-demo-servlet.xml</param-value>
        </init-param>

        <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>

</web-app>
```

Web App Initializer

Web App Initializer

- Spring MVC provides support for web app initialization

Web App Initializer

- Spring MVC provides support for web app initialization
- Makes sure your code is automatically detected

Web App Initializer

- Spring MVC provides support for web app initialization
- Makes sure your code is automatically detected
- Your code is used to initialize the servlet container

Web App Initializer

- Spring MVC provides support for web app initialization
- Makes sure your code is automatically detected
- Your code is used to initialize the servlet container

AbstractAnnotationConfigDispatcherServletInitializer

Web App Initializer (more info)

AbstractAnnotationConfigDispatcherServletInitializer

Web App Initializer (more info)

AbstractAnnotationConfigDispatcherServletInitializer

- Your TO DO list

Web App Initializer (more info)

AbstractAnnotationConfigDispatcherServletInitializer

- Your TO DO list
  - Extend this abstract base class

Web App Initializer (more info)

AbstractAnnotationConfigDispatcherServletInitializer

- Your TO DO list
  - Extend this abstract base class
  - Override required methods

Web App Initializer (more info)

AbstractAnnotationConfigDispatcherServletInitializer

- Your TO DO list
  - Extend this abstract base class
  - Override required methods
  - Specify servlet mapping and location of your app config

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {












}
```

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```java
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }









}
```

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { DemoAppConfig.class };
    }






}
```

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { DemoAppConfig.class };
    }



}
```

Our config class from Step 2

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { DemoAppConfig.class };
    }




}
```

Step 3: Create Spring Dispatcher Servlet Initializer

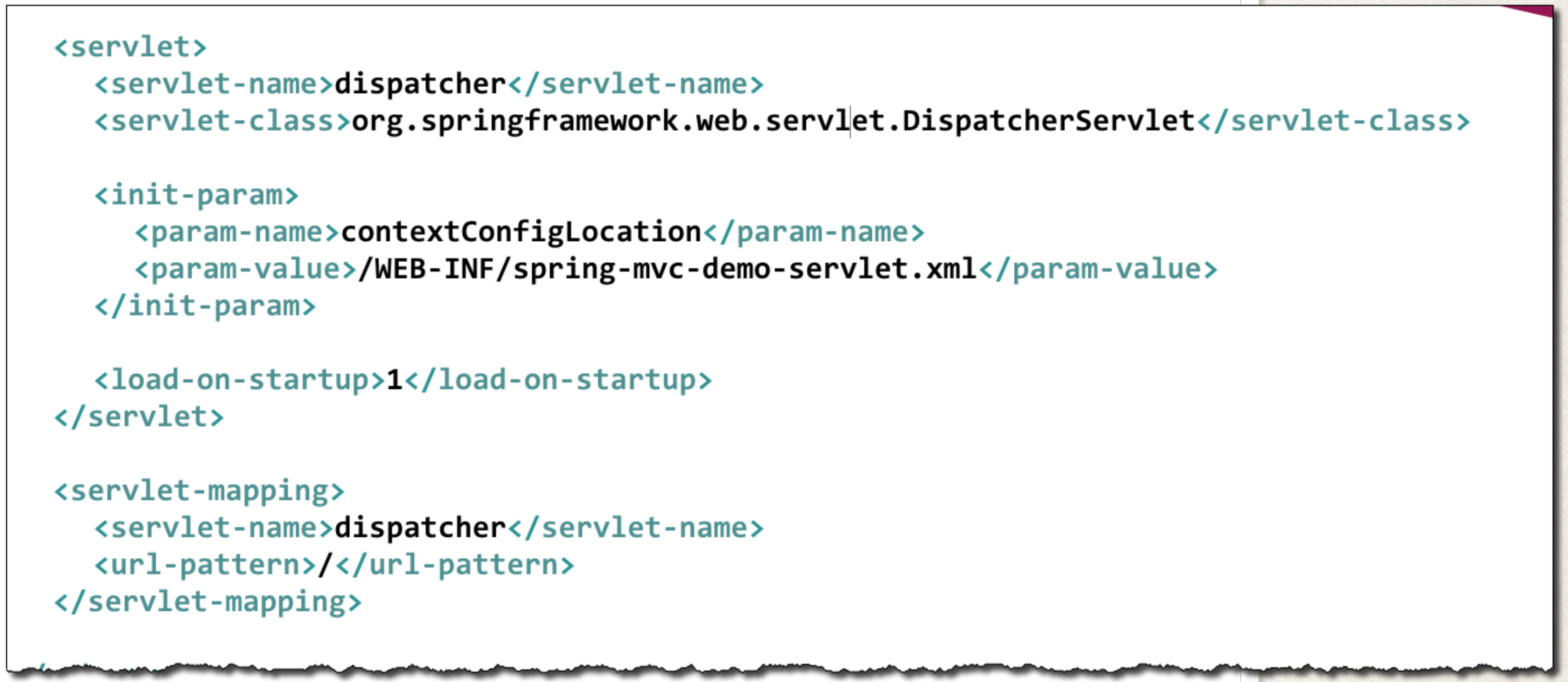
File:MySpringMvcDispatcherServletInitializer.java

```
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { DemoAppConfig.class };
    }

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

}
```

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```java
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { DemoAppConfig.class };
    }

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

}
```

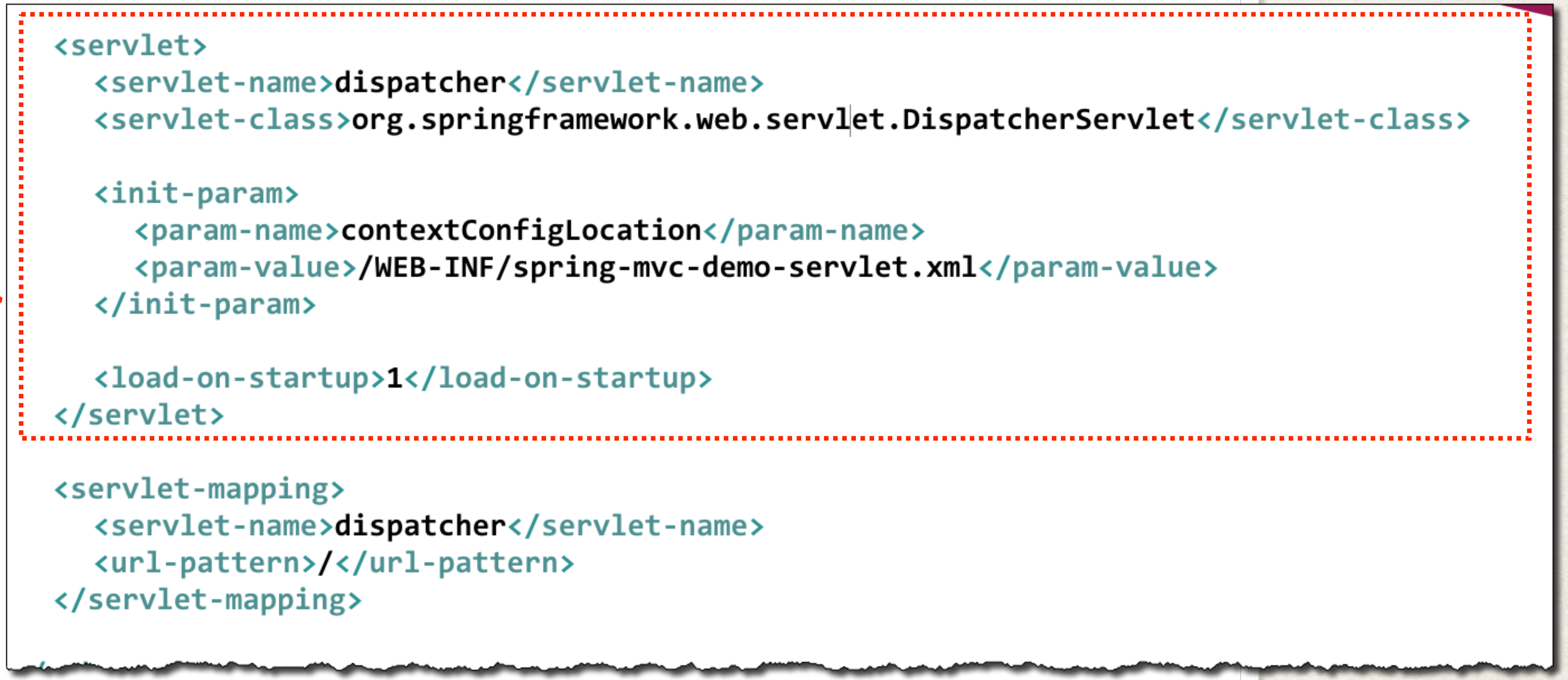


```xml
<servlet>
  <servlet-name>dispatcher</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

  <init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring-mvc-demo-servlet.xml</param-value>
  </init-param>

  <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>dispatcher</servlet-name>
  <url-pattern>/</url-pattern>
</servlet-mapping>
```

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```java
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { DemoAppConfig.class };
    }

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

}
```

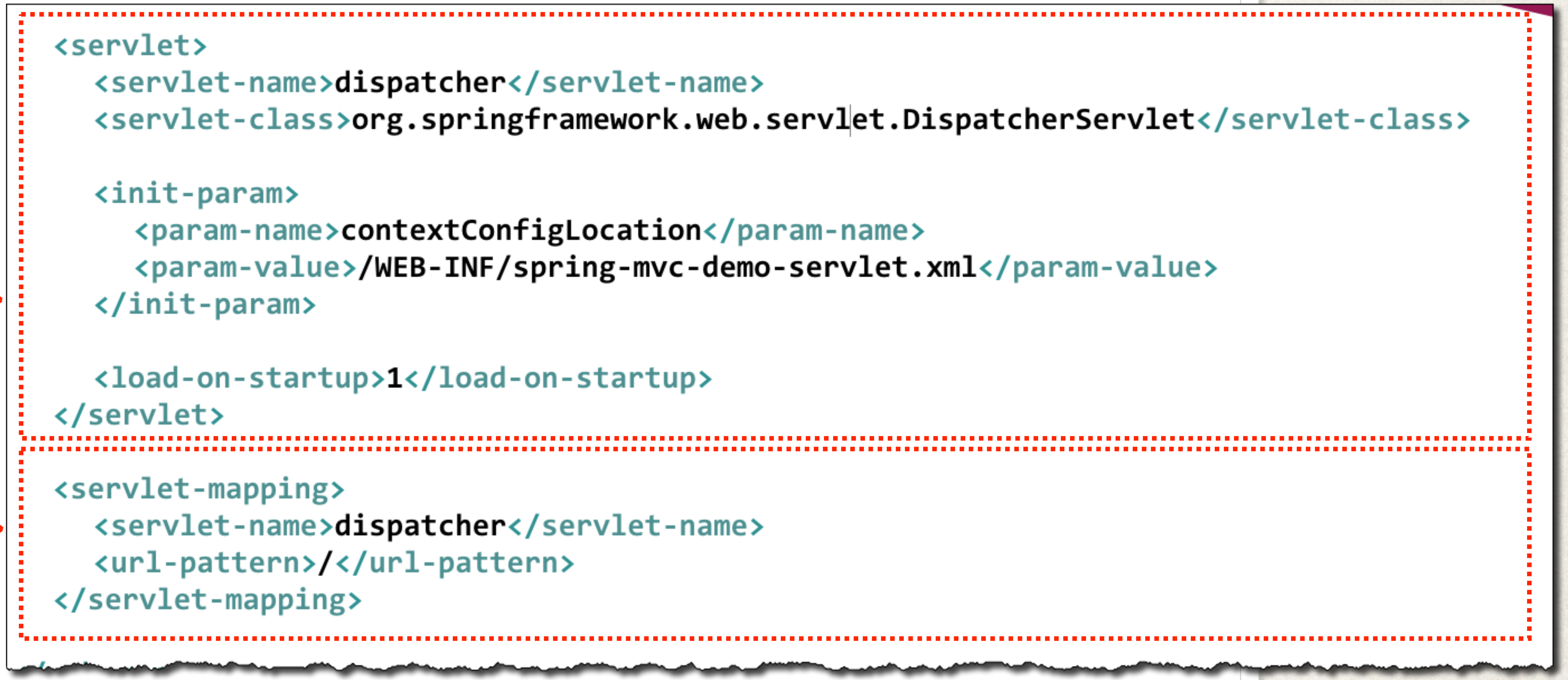


```xml
<servlet>
  <servlet-name>dispatcher</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

  <init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring-mvc-demo-servlet.xml</param-value>
  </init-param>

  <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>dispatcher</servlet-name>
  <url-pattern>/</url-pattern>
</servlet-mapping>
```

Step 3: Create Spring Dispatcher Servlet Initializer

File:MySpringMvcDispatcherServletInitializer.java

```java
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {

    @Override
    protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return new Class[] { DemoAppConfig.class };
    }

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

}
```



```xml
<servlet>
   <servlet-name>dispatcher</servlet-name>
   <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

   <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>/WEB-INF/spring-mvc-demo-servlet.xml</param-value>
   </init-param>

   <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
   <servlet-name>dispatcher</servlet-name>
   <url-pattern>/</url-pattern>
</servlet-mapping>
```

Whew!!!!

Now for the easy stuff

Create Spring MVC Controller and view page

easy peazy …

Step 4: Develop our Spring Controller

File: DemoController.java

```
@Controller
public class DemoController {





}
```

Step 4: Develop our Spring Controller

File: DemoController.java

```
@Controller
public class DemoController {

    @GetMapping("/")
    public String showHome() {

        return "home";
    }

}
```

Step 4: Develop our Spring Controller

File: DemoController.java

```
@Controller
public class DemoController {

    @GetMapping("/")
    public String showHome() {

        return "home";
    }

}
```

/WEB-INF/view/ home .jsp

Step 4: Develop our Spring Controller

File: DemoController.java

```
@Controller
public class DemoController {

    @GetMapping("/")
    public String showHome() {

        return "home";
    }

}
```

View name

/WEB-INF/view/ home .jsp

Step 5: Develop our JSP view page

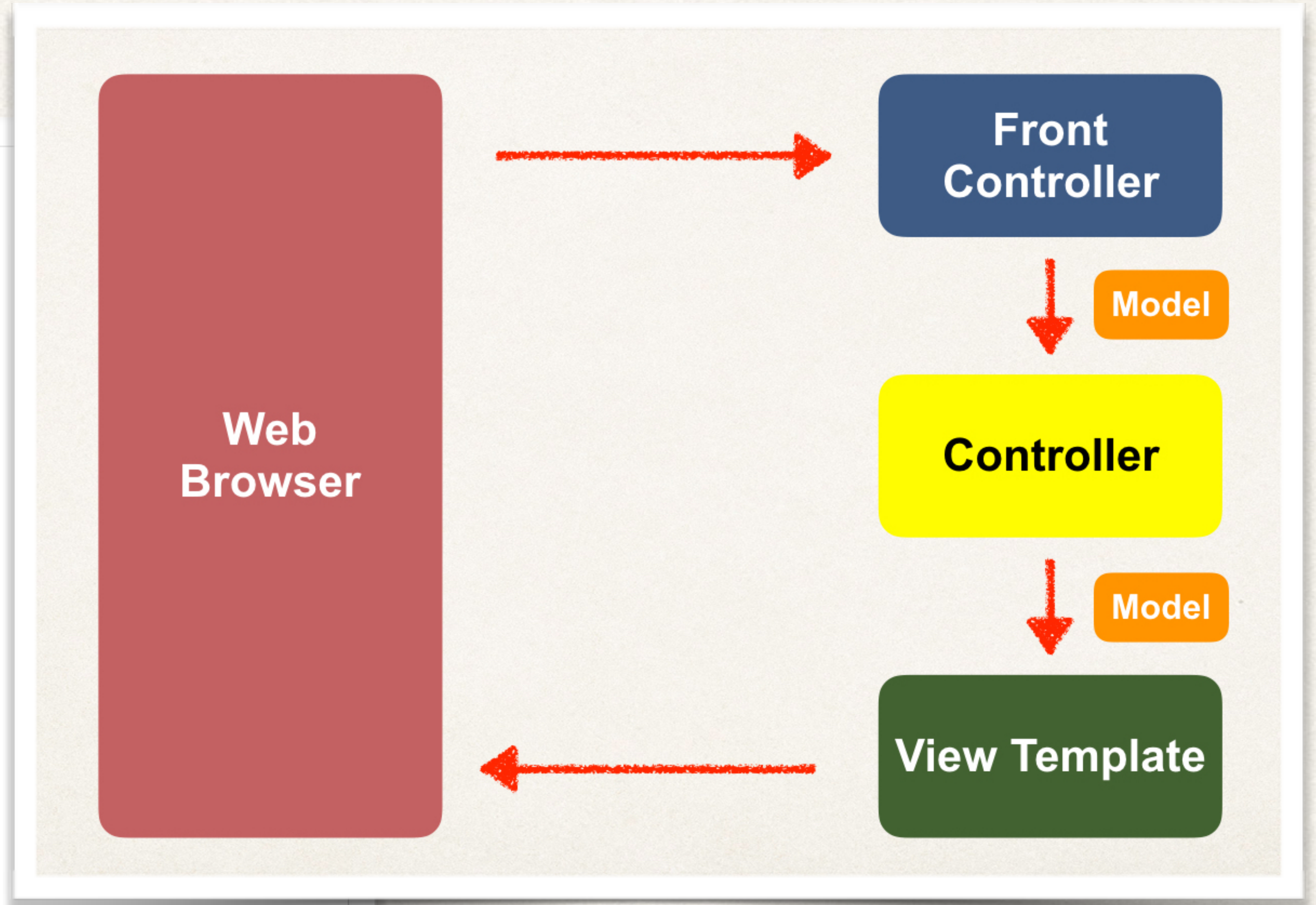
File: /WEB-INF/view/home.jsp

```
<html>

<body>

    Welcome to the luv2code company home page!

</body>

</html>
```

Step 5: Develop our JSP view page

File: /WEB-INF/view/home.jsp

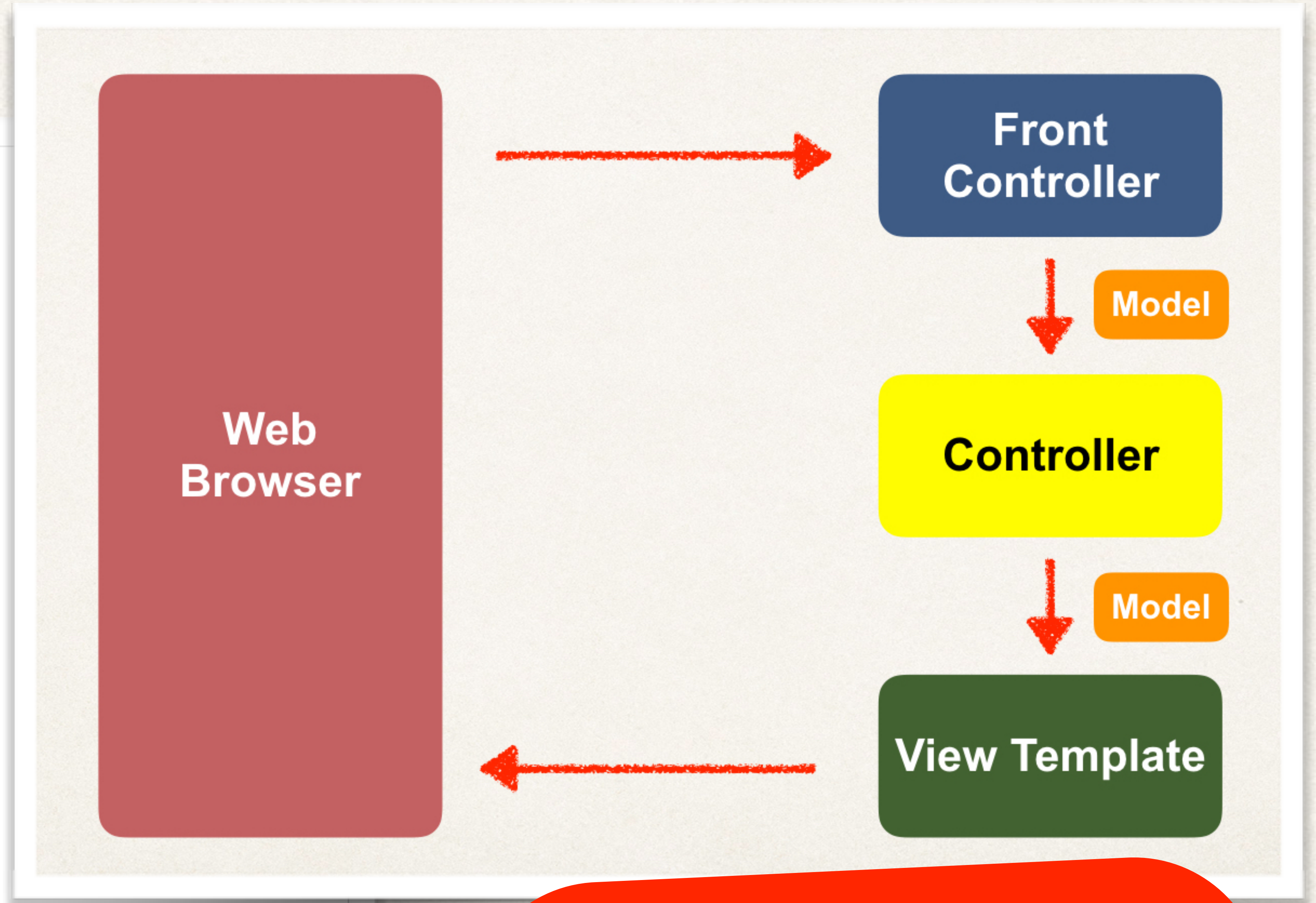
```
<html>

<body>

    Welcome to the luv2code company home page!

</body>

</html>
```



Step 5: Develop our JSP view page

File: /WEB-INF/view/home.jsp

```
<html>

<body>

   Welcome to the luv2code company home page!

</body>

</html>
```



No XML!