Spring Boot Dev-Tools

Spring Boot DevTools makes the application development experience more easier.

Spring Boot Developer Tools

- **Spring Boot 1.3** introduced a new set of developer tools that make it even easier to work with Spring Boot at development time.
  - **Automatic restart** —Restarts a running application when files are changed in the class path
  - **Live Reload support** —Changes to resources trigger a browser refresh automatically
  - **Remote development** —Supports automatic restart and Live Reload when deployed remotely
  - **Development property defaults** —Provides sensible development defaults for some configuration properties.

Maven Dependency

```
<dependencies>
        <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-devtools</artifactId>
        <optional>true</optional>
        </dependency>
</dependencies>
```

Automatic Restart

- With the developer tools active, any changes to files on the class path will trigger an application restart.
- Application code that is being worked on will be loaded into a separate restart class loader.
- File on the class path change
- Two class loaders
  - Third party classes (base class loader)
  - Application classes (restart class loader)
  - Static resources do not need a restart

Automatic Restart

- Static resources in /static or /public likewise don't require an application restart, so Spring Boot developer tools exclude the following paths from restart consideration: **/META-INF/resources, /resources, /static, /public, /templates.**

- On the other hand, if you'd rather disable automatic restart completely, you can set

- **spring.devtools.restart.enabled =false**:

- The default set of restart path exclusions can be overridden by setting the **spring.devtools.restart.exclude** property.

- **spring.devtools.restart.exclude**=/static/**,/templates/**

Automatic Restart

- **Watching Additional Paths**

- You may want your application to be restarted or reloaded when you make changes to files that are not on the classpath.

- To do so, use the spring.devtools.restart.additional-paths property to configure additional paths to watch for changes.

- **spring.devtools.restart.additional-paths=/monitoring**

- **spring.devtools.restart.additional-exclude=**

Automatic Restart

- **Logging changes in condition evaluation**
- By default, each time your application restarts, a report showing the condition evaluation delta is logged. The report shows the changes to your application's auto-configuration as you make changes such as adding or removing beans and setting configuration properties.

- **spring.devtools.restart.log-condition-evaluation-delta=false**

Automatic Restart

- **Trigger File**
- If you work with an IDE that continuously compiles changed files, you might prefer to trigger restarts only at specific times.
- To do so, you can use a "trigger file", which is a special file that must be modified when you want to actually trigger a restart check.
- Changing the file only triggers the check and the restart only occurs if Devtools has detected it has to do something.
- To use a trigger file, set the **spring.devtools.restart.trigger-file** property to the path of your trigger file.

Automatic Restart

- **Global Settings**
- You can configure global devtools settings by adding a file named **.spring-boot-devtools.properties** to your $HOME folder (note that the filename starts with ".").
- Any properties added to this file apply to all Spring Boot applications on your machine that use devtools. For example, to configure restart to always use a trigger file, you would add the following property:
- ~/.spring-boot-devtools.properties.
- spring.devtools.reload.trigger-file=.reloadtrigger

Spring Boot: Property Default Sensible caching

- Several of the libraries supported by Spring Boot use caches to improve performance.
- Spring MVC can add HTTP caching headers to responses when serving static resources.
- While caching is very beneficial in production, it can be counter-productive during development
- spring-boot-devtools **disables the caching options by default**.
- Thymeleaf offers the **spring.thymeleaf.cache** property.
- Rather than needing to set these properties manually, the spring-boot-devtools module automatically applies sensible development-time configuration.

Spring Boot: Property Default Sensible caching

- Because you need more information about web requests while developing Spring MVC and Spring WebFlux applications, developer tools will enable DEBUG logging for the web logging group.

- This will give you information about the incoming request, which handler is processing it, the response outcome, etc.

- If you wish to log all request details (including potentially sensitive information), you can turn on the **spring.http.log-request-details** configuration property.

- If you don't want property defaults to be applied you can set **spring.devtools.add-properties** to false in your application.properties.

Live Reload

- The spring-boot-devtools module includes an embedded LiveReload server that can be used to trigger a browser refresh when a resource is changed.

- LiveReload browser extensions are freely available for Chrome, Firefox and Safari from livereload.com.

- If you do not want to start the LiveReload server when your application runs, you can set the **spring.devtools.livereload.enabled** property to false.

- **spring.devtools.livereload.enabled = false**:

Dev tools property Default

- Developer tools are automatically disabled when running a fully packaged application. If your application is launched from java -jar or if it is started from a special classloader, then it is considered a "production application".

- Property Defaults
  - Several of the libraries supported by Spring Boot use caches to improve performance. For example, template engines cache compiled templates to avoid repeatedly parsing template files. Also, Spring MVC can add HTTP caching headers to responses when serving static resources.

- For a complete list of the properties that are applied by the devtools,
  see DevToolsPropertyDefaultsPostProcessor.

Q & A

Thank You!