

Allgemeine Infos

Projekt Konfigurator:

<http://start.spring.io/>

Allgemeine Blogeinträge:

<http://blog.codeleak.pl/2014/04/how-to-spring-boot-and-thymeleaf-with-maven.html>

Properties und Filterung

* Erbt das Projekt von spring-boot-parent so ist im Mavenbuild bereits die Filterung überschrieben:

** In der application.properties werden Ausdrücke @xxx@ durch maven Werte ersetzt.

* Thymleaf ersetzt Ausdrücke der Art \${xxx} und #{xxx} zur Laufzeit mit Werten

applications.properties

spring.thymeleaf.cache: false Schaltet den Cache für Thymleaf aus. Damit wird bei jedem Request das Template neu übersetzt und bei der Entwicklung braucht der Server für HTML Anpassungen nicht neu gestartet werden.

endpoints.shutdown.enabled Aktiviert den Endpoint zum stoppen des Servers per POST Request auf den Management Port

Das Herunterfahren des Servers lässt sich bei der Konfiguration des Managementports auf **8080** und eingeschaltetem Endpoint wie folgt von der Console aus auslösen: **curl -X POST http://localhost:8080/shutdown**

Bootstrap - Definition Liste horizontal

```
<dl class="dl-horizontal">
<dt>
<span class=" glyphicon glyphicon-trash ">
</span>
</dt>
<dd>Buch löschen</dd>
<dt>
<span class=" glyphicon glyphicon-warninng-sign">
</span>
</dt>
<dd>ISBN ungültig</dd>
</dl>
```

Interceptors

Interceptoren in Spring Boot können genutzt werden um:

* Requests an den Controller zu modifizieren (Header etc.)

* Responses an den Client zu modifizieren (Header etc.)

Ein Interceptor wird erstellt durch eine Klasse mit:

* @Component

* implements HandlerInterceptor

Folgende Lifecycle Hooks sind verfügbar:

* preHandle: Modifikation des Request vor Erreichen des Controllers. Nach dem Aufruf wird ein Response erzeugt.

* postHandle: Modifikation des Response vor Herausgabe an den Client

* afterCompletion: Modifikation wenn Request und Response fertig bearbeitet sind

Jeder Interceptor muss bei der InterceptorRegistry angemeldet werden

@Component

```
public class ProductServiceInterceptor implements HandlerInterceptor {
```

```
    WebMvcConfigurerAdapter {
```

```
        @Autowired
```

```
        ProductServiceInterceptor productServiceInterceptor;
```

```
        @Override
```

```
        public void addInterceptors (InterceptorRegistry registry) {
```

```
            registry.addInterceptors (productServiceInterceptor);
```

```
        }
```

```
}
```

Quelle: https://www.tutorialspoint.com/spring_boot/spring_boot_interceptor



By Huluhu424242
(FunThomas424242)

Published 17th February, 2019.

Last updated 14th July, 2019.

Page 1 of 1.

Sponsored by **CrosswordCheats.com**

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>