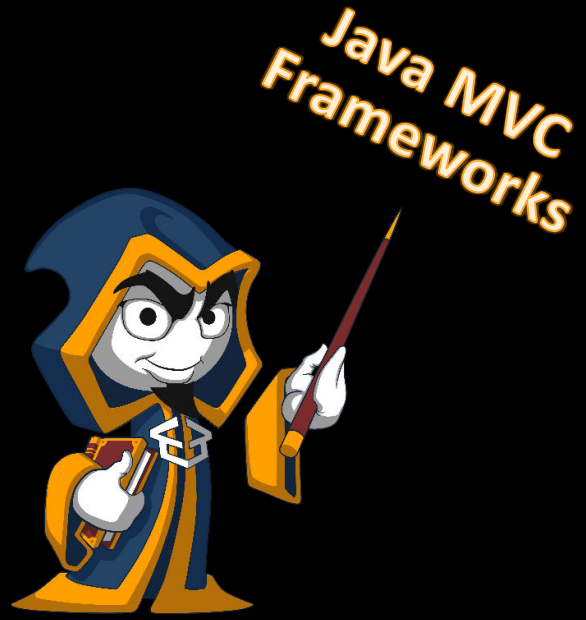
Spring Filters

Filters & Interceptors



SoftUni Team

Technical Trainers

Software University

http://softuni.bg

Table of Contents

Questions

SoftUni Foundation

sli.do

#java-web


SoftUni
Foundation
spring
Spring Interceptors
Pre/Post Handlers

SoftUni Foundation

Problem

Problem: Application Title

SoftUni Foundation

- Place the application name in front of every page title:

```
Add Game
Register
Home
->
SoftUni Store - Add Game
SoftUni Store - Register
SoftUni Store - Home
```

SoftUni Foundation

Solution

What We Need to Do?

SoftUni Foundation

- We need to modify the Model attributes that we're sending to the view
- That means we need to intercept the request before the view
- Pre-Handle will be executed before the Action is executed
- Post-Handle will be executed after the Action, but before the view is rendered
- After-Completion will be executed after the view is rendered

Creating Interceptor

SoftUni Foundation

- Class extending **HandlerInterceptorAdapter**:

```
@Component
public class TitleInterceptor extends
HandlerInterceptorAdapter {

}
```

Creating Interceptor (2)

SoftUni Foundation

- Override the **postHandle()**:

```
@Override
public void postHandle(HttpServletRequest request,
HttpServletResponse response, Object handler,
ModelAndView modelAndView) throws Exception {

  String title = "SoftUni Store - " +
modelAndView.getModelMap().get("title");

  modelAndView.addObject("title", title);
}
```

Creating MVC Config

SoftUni Foundation

- Class extending **`WebMvcConfigurerAdapter`**:

```
@Configuration
public class WebMvcConfig extends
                        WebMvcConfigurerAdapter {
  private final TitleInterceptor titleInterceptor;
  @Autowired
  public WebMvcConfig(TitleInterceptor titleInterceptor) {
    this.titleInterceptor = titleInterceptor;
  }
}
```

Creating MVC Config (2)

SoftUni Foundation

- Override the `addInterceptors()`:

```
@Override
public void addInterceptors(InterceptorRegistry registry) {
  registry.addInterceptor(this.titleInterceptor);
}
```

- Test it with your application and see if it works

SoftUni Foundation

Problem

Problem: Log the Application Activity

SoftUni Foundation

- Log in the database when anyone accesses any possible route. The log should contain date and message in the following format:

```
[Controller - Action] Action Execute Time: %d ms, Overall Execute Time: %d ms
```

id	date	text
1	2017-03-15 23:16:58	[class softuni.controllers.HomeController - index] Action Execute Time: 15 ms, Overall Execute Time: 71 ms
2	2017-03-15 23:17:13	[class softuni.controllers.UserController - register] Action Execute Time: 0 ms, Overall Execute Time: 19 ms
3	2017-03-15 23:17:14	[class softuni.controllers.UserController - login] Action Execute Time: 1 ms, Overall Execute Time: 9 ms
4	2017-03-15 23:17:24	[class softuni.controllers.HomeController - index] Action Execute Time: 10 ms, Overall Execute Time: 34 ms
5	2017-03-15 23:17:26	[class softuni.controllers.GameController - details] Action Execute Time: 5 ms, Overall Execute Time: 16 ms
6	2017-03-15 23:17:28	[class softuni.controllers.HomeController - index] Action Execute Time: 13 ms, Overall Execute Time: 35 ms

SoftUni Foundation

Solution

Hints: Log the Application Activity

SoftUni Foundation

- You can get the current time with:

```
System.currentTimeMillis();
```

- Your interceptor should override the **HandlerInterceptor** interface, because we want all of the methods:

```
public class LogInterceptor implements HandlerInterceptor
```

- You will need new entity with corresponding repository and service

Interceptor Pre-Handle

SoftUni Foundation

- The `preHandle()` should only get the current time:

```
@Override
public boolean preHandle(HttpServletRequest req,
HttpServletResponse resp, Object o) throws Exception {

  req.setAttribute("preHandleTime",
       System.currentTimeMillis());

  return true;
}
```

Interceptor Post-Handle

SoftUni Foundation

- The `postHandle()` should also only get the current time:

```
@Override
public boolean postHandle(HttpServletRequest req,
HttpServletResponse resp, Object o, ModelAndView mav)
throws Exception {

  req.setAttribute("postHandleTime",
       System.currentTimeMillis());
}
```

Interceptor After-Completion

SoftUni Foundation

- The `afterCompletion()` should calculate the time and create the log:

```
@Override
public void afterCompletion(HttpServletRequest req,
HttpServletResponse resp, Object handler, Exception e)
throws Exception {
   …
}
```

Interceptor After-Completion (2)

SoftUni Foundation

```
//TODO: Calculate the time
  HandlerMethod handlerMethod = (HandlerMethod) handler;

  String message = String.format("[%s - %s] Action Execute
Time: %d ms, Overall Execute Time: %d ms",
handlerMethod.getBean().getClass(),
                handlerMethod.getMethod().getName(),
                actionTime, overallTime);

  Log log = new Log(message, new Date());
  this.logService.create(log);
```

Register the Interceptor

SoftUni Foundation

- You only need to register the interceptor in the configuration class:

```
@Override
public void addInterceptors(InterceptorRegistry registry) {
  registry.addInterceptor(this.titleInterceptor);
  registry.addInterceptor(this.logInterceptor);
}
```

- Test it to see if it works

SoftUni Foundation

Problem

Problem: Log Only Register/Login/Logout

- Use the DB from the previous problem. Log only the activity regarding Register/Login/Logout functionality.

SoftUni Foundation

Solution

Modify the Interceptor Registration

- We only need to modify the paths that our interceptor will be listening to:

```
@Override
public void addInterceptors(InterceptorRegistry registry) {
  registry.addInterceptor(this.titleInterceptor);
  registry.addInterceptor(this.logInterceptor)
        .addPathPatterns("/register", "/logout", "/login");
}
```

SoftUni Foundation

Problem

Problem: Log Specific Actions

SoftUni Foundation

- Use the DB from the previous problem. Implement a way to log only specific actions, without using the path patterns. You should work with the `Object` that you are given in the interceptor.

SoftUni Foundation

Solution

Solution: Log Specific Actions

SoftUni Foundation

- Create **@Log** annotation.
- It should only be applied to actions.
- If an action has this annotation, you should create a log in the database when someone is accessing it.
- Actions without the annotation mustn't be logged

Interceptor Modifications

SoftUni Foundation

- All of your overridden methods should check if the annotation is present on the handler object:

```
HandlerMethod handlerMethod = (HandlerMethod) handler;
if(!handlerMethod.getMethod()
                 .isAnnotationPresent(Log.class)) {
  return;
}
```

- Remove the path patterns from the configuration and check if it works

Summary

- Spring Interceptors can be used to modify your response or request in any given moment
  - `preHandle()` is executed before the action
  - `postHandle()` is executed after the action, but before rendering the view
  - `afterCompletion()` is executed after the view is rendered
- The `Object` received in the interceptor methods is the `HandlerMethod` (action) that will serve your request

Java MVC Frameworks – Spring Filters

SoftUni Foundation

Questions?

https://softuni.bg/courses/java-mvc-frameworks-spring

SoftUni Diamond Partners

SoftUni Foundation

INDEAVR
Serving the high achievers

INFRAGISTICS®

SoftwareGroup
doing it right

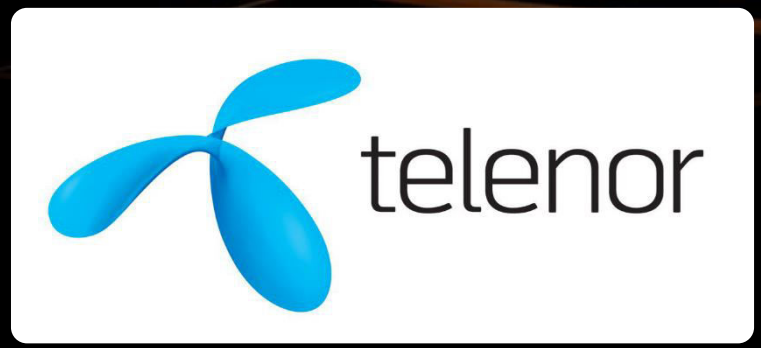
xssoftware

NETPEAK
SEO and PPC for Business

SUPER HOSTING .BG®

SoftUni Diamond Partners

SoftUni Foundation



License

SoftUni Foundation

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

Software University

SoftUni Foundation

FORUM