Debugging and Testing

Martin Ledvinka

martin.ledvinka@fel.cvut.cz

Winter Term 2017

Contents

Debugging

Debugging

- No application is perfect,
- When an issue occurs, it is necessary to be able to:
    - Identify it,
    - Reproduce it,
    - Determine its cause,
    - Fix it.

Debugging

Multiple ways of identifying an issue:

1. Error in browser (Developer console/tools – *F12*),
2. Log entry,
3. Exception stack trace,
   - It is necessary to be able to determine which part of a stack trace is relevant and which is just the framework infrastructure,
   - Enterprise applications → long stack traces.

Stack trace

Stack trace II

```
java.lang.NullPointerException: null
    at cz.cvut.kbss.inbas.reporting.persistence.dao.BaseDao.persist(BaseDao.java:70)
    at cz.cvut.kbss.inbas.reporting.service.repository.BaseRepositoryService.persist(BaseRepositoryService.java:36)
    at cz.cvut.kbss.inbas.reporting.service.MainReportService.persist(MainReportService.java:80)
    at cz.cvut.kbss.inbas.reporting.service.CachingReportBusinessService.persist(CachingReportBusinessService.java:39)
    at cz.cvut.kbss.inbas.reporting.rest.ReportController.createReport(ReportController.java:43)
    at cz.cvut.kbss.inbas.reporting.rest.ReportController$$FastClassBySpringCGLIB$$87936B8a.invoke(<generated>)
    at org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:204)
    at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.invokeJoinpoint(CglibAopProxy.java:717)
    at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:157)
    at org.springframework.security.access.intercept.aopalliance.MethodSecurityInterceptor.invoke(MethodSecurityInterceptor.java:68)
    at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:179)
    at org.springframework.aop.framework.CglibAopProxy$DynamicAdvisedInterceptor.intercept(CglibAopProxy.java:653)
    at cz.cvut.kbss.inbas.reporting.rest.ReportController$$EnhancerBySpringCGLIB$$87936B8a.createReport(<generated>) <14 internal calls>
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:648) <1 internal calls>
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:729)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:291)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
    at org.springframework.security.web.access.intercept.FilterSecurityInterceptor.invoke(FilterSecurityInterceptor.java:126)
    at org.springframework.security.web.access.intercept.FilterSecurityInterceptor.doFilter(FilterSecurityInterceptor.java:90)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.access.ExceptionTranslationFilter.doFilter(ExceptionTranslationFilter.java:114)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.session.SessionManagementFilter.doFilter(SessionManagementFilter.java:122)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.authentication.AnonymousAuthenticationFilter.doFilter(AnonymousAuthenticationFilter.java:111)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.servletapi.SecurityContextHolderAwareRequestFilter.doFilter(SecurityContextHolderAwareRequestFilter.java:169)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.savedrequest.RequestCacheAwareFilter.doFilter(RequestCacheAwareFilter.java:48)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.session.ConcurrentSessionFilter.doFilter(ConcurrentSessionFilter.java:133)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.authentication.AbstractAuthenticationProcessingFilter.doFilter(AbstractAuthenticationProcessingFilter.java:205)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.authentication.logout.LogoutFilter.doFilter(LogoutFilter.java:120)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.header.HeaderWriterFilter.doFilterInternal(HeaderWriterFilter.java:64) <1 internal calls>
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.context.SecurityContextPersistenceFilter.doFilter(SecurityContextPersistenceFilter.java:91)
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.context.request.async.WebAsyncManagerIntegrationFilter.doFilterInternal(WebAsyncManagerIntegrationFilter.java:53) <1 internal calls>
    at org.springframework.security.web.FilterChainProxy$VirtualFilterChain.doFilter(FilterChainProxy.java:330)
    at org.springframework.security.web.FilterChainProxy.doFilterInternal(FilterChainProxy.java:213)
    at org.springframework.security.web.FilterChainProxy.doFilter(FilterChainProxy.java:176) <2 internal calls>
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:239)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
    at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
    at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:239)
    at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
    at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:219)
    at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:106)
    at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:502)
    at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:142)
    at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:79)
```

Logging

Logging

- Logging provides important auditing information about the application,
- Good to log:
    - Exceptions (duh),
    - Application state information,
    - Input/output data information,
- Configurable levels,
    - Minimal logging for production,
    - Maximal for debugging/testing.

Java - logging frameworks

- **JUL (java.util.logging)**
  - Part of JDK, no external libraries necessary,
  - multitude of logging levels,

- **Apache Log4j**
  - Leading logging frameworks for years,
  - Easy configuration,

- **Simple Logging Facade for Java (SLF4J)**
  - Logging facade, requires implementation,
  - This separation is favourable – possible to use with JUL, Log4j, logback or others,
  - Current logging leader,

- **logback**
  - Successor of the original Log4j,
  - Today's most common configuration – SLF4J + logback.

Logging levels

From SLF4J:

1. Trace
2. Debug
3. Info
4. Warn
5. Error
6. Fatal
7. All

Testing

Enterprise Application Testing

We are...
Assuming you already have experience with testing.

Enterprise application testing is:

- Harder,
- Much more about the right configuration,
- Otherwise similar to the usual:
    - JUnit,
    - Mockito,
    - Cucumber,
    - Selenium.
- Should be automated as much as possible,
    - Big applications, not enough testers, not enough time.

Code coverage

- How much of the code is covered by tests,
- Coverage of lines, branches, classes, methods etc.,
- **100% coverage does not mean code without bugs**,
- JaCoCo, Cobertura,
- Testing and coverage can be configured as part of Maven build.

Tasks

Time to do some work

- Reporting Tool will not deploy, let's find out why,

Time to do some work

- Reporting Tool will not deploy, let's find out why,
- After deploy, it seems to refuse to return any reports.

Time for **you** to do some work

- Now we have reports, but when we update one, the information about the update is not correct,
- Find out why,
- Locate the issue,
- Write a test which reproduces it,
- Fix the issue,
- Run the test again to verify the fix worked.

Resources

- BI-ATS at FIT
  - https://edux.fit.cvut.cz/courses/BI-ATS/
- JaCoCo Maven Plugin
  - http://www.eclemma.org/jacoco/trunk/doc/maven.html
- SJF4J Docs
  - http://www.slf4j.org/docs.html