

Exception Logging

Ab Nuclos 4.43

Exceptions an einen Client enthalten nicht länger einen Stacktrace und Cause (Herkunft), diese werden gesondert gesichert.

Zusätzlich werden alle `RuntimeExceptions` zu einer neutralen `InternalErrorException` mit einem `HashCode` zum Original um den Stacktrace möglichst schnell zu ermitteln. Dazu benötigt man Zugriff auf das Log Verzeichnis des Servers und ist damit vor Unbefugten gesichert.

`BusinessExceptions` werden an den Client geleitet, sind aber ebenfalls ohne Stacktrace und Cause, aber mit allen nötigen Extras die ein Client zur Verarbeitung benötigt (`InputRequired`, `FieldValidation`, etc.)

stacktrace-business.log

Beinhaltet die an die Clients ausgelieferten **reduzierten** Stacktraces der `BusinessExceptions`.

Beispiel:

```
2020-06-24 08:29:40 - null

org.nuclos.api.context.InputRequiredException: null

at extension.nuclet.test.rules.TestInputRequiredExceptionExtension.doInputMemo
(TestInputRequiredExceptionExtension.java:198) ~[nuclos-integration-tests-rules-4.43.0-SNAPSHOT.jar:?]

at extension.nuclet.test.rules.TestInputRequiredExceptionExtension.doInputRequired
(TestInputRequiredExceptionExtension.java:53) ~[nuclos-integration-tests-rules-4.43.0-SNAPSHOT.jar:?]

at extension.nuclet.test.rules.TestInputRequiredExceptionExtension.custom(TestInputRequiredExceptionExtension.
java:34) ~[nuclos-integration-tests-rules-4.43.0-SNAPSHOT.jar:?]
```

Stacktrace Zeilen der Infrastruktur (Nuclos, Spring, Java, Tomcat, Glassfish) werden nicht ausgegeben, da sie für `BusinessExceptions` keine relevanten Informationen enthalten.

stacktrace-internal.log

Beinhaltet den **vollen** Stacktraces aller anderen Exceptions (Runtime) die an die Clients ausgeliefert wurden.

Ein Log Eintrag beginnt direkt mit den `HashCode` # der so auch im Client angezeigt wird.

Beispiel:

```
#-22567581 2020-06-24 09:17:21

java.lang.NullPointerException: null

at org.nuclos.server.eventsupport.ejb3.EventSupportFacadeBean.createBusinessObject(EventSupportFacadeBean.java:
1855) ~[nuclos-server-4.43.0-SNAPSHOT.jar:?]

at org.nuclos.server.eventsupport.ejb3.EventSupportFacadeBean.executeUpdateSupportEvent(EventSupportFacadeBean.
java:1555) ~[nuclos-server-4.43.0-SNAPSHOT.jar:?]

at org.nuclos.server.eventsupport.ejb3.EventSupportFacadeBean.fireSaveEventSupport_aroundBody62
(EventSupportFacadeBean.java:2183) ~[nuclos-server-4.43.0-SNAPSHOT.jar:?]

at org.nuclos.server.eventsupport.ejb3.EventSupportFacadeBean$AjcClosure63.run(EventSupportFacadeBean.java:1) ~
[nuclos-server-4.43.0-SNAPSHOT.jar:?]

at org.springframework.transaction.aspectj.AbstractTransactionAspect.
ajc$around$org_springframework_transaction_aspectj_AbstractTransactionAspect$1$2a73e96cproceed
(AbstractTransactionAspect.aj:66) ~[spring-aspects-4.3.17.RELEASE.jar:4.3.17.RELEASE]

...
```

Ein Eintrag im `Server.log` sieht wie folgt aus:

```
2020-06-24 09:17:21,916 WARN [org.nuclos.server.common.NuclosRemoteInvocationExecutor] - InternalError '#-
22567581 2020-06-24 09:17:21': java.lang.NullPointerException
```

Log Rotation

Um das Auffinden für den Support möglichst einfach zu gestalten, rotieren die neuen Logdateien jede Stunde. Aufbewahrt werden maximal **8** stacktrace-business.log bzw. **512** stacktrace-internal.log Dateien.

Das kann in der log4j2.xml ([NUCLOS]/conf) nach belieben angepasst werden, wird allerdings von einem Installer wieder auf den Auslieferungszustand gesetzt.

Wenn man dies verhindern möchte, kann man die log4j2.xml nach [NUCLOS]/extensions/conf kopieren, dann nimmt der Installer immer diese dort hinterlegte Datei und die Anpassungen bleiben erhalten

```
<RollingFile name="StacktraceBusiness" fileName="${server.home}/logs/stacktrace-business.log"
  filePattern="${server.home}/logs/stacktrace-business-%d{yyyy-MM-dd-hh}.log.gz"
  append="true">
  <PatternLayout pattern="%d{yyyy-MM-dd HH:mm:ss} - %m%n"/>
  <Policies>
    <TimeBasedTriggeringPolicy filePattern="${server.home}/logs/stacktrace-business-%d{yyyy-MM-dd-hh}.log.
gz" />
  </Policies>
  <DefaultRolloverStrategy max="8"/>
</RollingFile>
<RollingFile name="StacktraceInternal" fileName="${server.home}/logs/stacktrace-internal.log"
  filePattern="${server.home}/logs/stacktrace-internal-%d{yyyy-MM-dd-hh}.log.gz"
  append="true">
  <PatternLayout pattern="%m%n"/>
  <Policies>
    <TimeBasedTriggeringPolicy filePattern="${server.home}/logs/stacktrace-internal-%d{yyyy-MM-dd-hh}.log.
gz" />
  </Policies>
  <DefaultRolloverStrategy max="512"/>
</RollingFile>
```

Altes Logging wiederherstellen

Möchte man wieder alles in der einen server.log Datei, so kann auch dies in der log4j2.xml ([NUCLOS]/conf) angepasst werden:

```
<Logger name="StacktraceBusiness" additivity="false" level="error">
  <AppenderRef ref="StacktraceBusiness"/>
</Logger>
<Logger name="StacktraceInternal" additivity="false" level="error">
  <AppenderRef ref="StacktraceInternal"/>
</Logger>
```

Entweder man löscht diese Einträge, oder setzt **additivity="true"**. Letzteres schreibt dann die Stacktraces in das server.log wie auch in die neuen Log Dateien.