Objektovo Orientované programovanie

8. Prednáška
LS 2024/2025
Juraj Petrík

IO

- FileWriter
- write()
- close()
- Všetko môže hodiť IOException!

- File - reprezentuje meno a cestu súboru (priečinku)

IO - Buffer

- BufferedWriter
- Prečo?
  - Disk je pomalší ako pamäť
  - Pracujeme s “chunkami”
- flush() – keď chceme zapísať predtým ako zaplníme buffer

io, nio, nio2

- Pri NIO.2 hovoríme zvyčajne o dvoch balíkoch:
  - Java.nio.file
    - Path interface
    - Paths class
    - Files class
  - Java.nio.file.attribute – metadáta

- Try-with-resources:
  - implements Autocloseable
  - multiple IO resources closed in opposite order

Dokumentácia (JavaDoc)

- Predstavte si svoj program ako knižnicu, ktorú niekto chce používať
- Kvalitná dokumentácia API je kritická pre (znovu)používanie kódu
- Javadoc sú “špeciálne komentáre”: /** **/
- Štandardizované, HTML ako Java Standard Library
- Núti rozmýšlať
- Napr. “pred” triedou, rozhraním metódou, atribútom
- https://www.oracle.com/technical-resources/articles/java/javadoc-tool.html

JavaDoc what (not) to do

- <code> pre keywordy a mená
- Overuse:
  - In-line links ({@link})
- Používajte tretiu osobu
- This instead of the keď odkazujeme na objekt z triedy ktorú opisujeme
- Najlepšie názvy sú samoopisné, tie zbytočne „nedokumentujte“

JavaDoc Tagy (na poradí záleží)

- @author (classes and interfaces only, required)
- @version (classes and interfaces only, required. See footnote 1)
- @param (methods and constructors only)
- @return (methods only)
- @exception (@throws is a synonym added in Javadoc 1.2)
- @see
- @since
- @serial (or @serialField or @serialData)
- @deprecated (see How and When To Deprecate APIs)

JavaDoc tags

- {@link package.class#member label}
- @see:
  - @see Java Dcoumentation
  - @see "This method performs some function.“
  - @see String#toLowerCase() convertToLowerCase
- {@inheritDoc}

JavaDoc Generovanie

- CLI – javadoc –d docs Class.java
- IDE
- Maven – mvn javadoc:javadoc
- Gradle – gradle javadoc

- Vlastné tagy

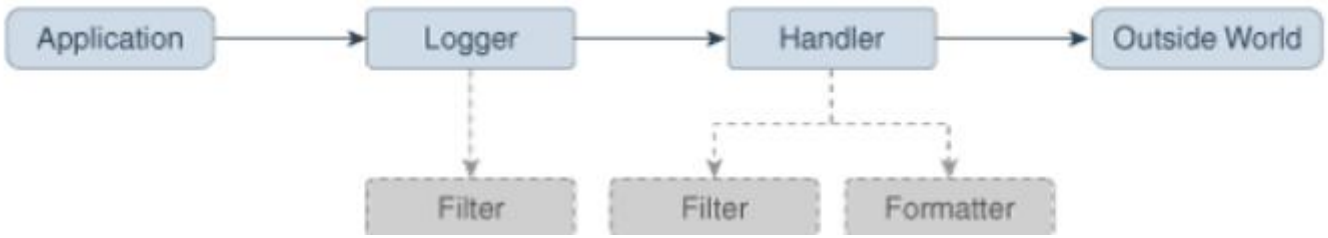
Logovanie

- Prečo?
  - Pomoc pri ladení
  - Auditné logy
  - Monitoring
  - Observabilita
- Viacero úrovní (LogLevel)
  - Napr.: Fatal, Error, Warn, Info, Debug, Trace, (Off)
- Ako?
  - Timestamp
  - Meno triedy
  - Log Level
  - ID Vlakna

- java.util.logging
- Log4j
- logback
- Log4j2
- SLF4J - The Simple Logging **Facade** for Java

- Štandardné kombo:
  - SLF4J + logback
  - SLF4J + log4j2

Java.util.logging



SLF4J

- Simple Logging **Facade** for Java
- Abstrakcia nad rôznymi logovacími frameworkami – plug & play

- import org.slf4j.Logger;
- import org.slf4j.LoggerFactory;

Some things to consider

- if (logger.isEnabled(Level.INFO))

  logger.info(String.format("The result is %d.", superExpensiveMethod()));
- Hot Path
- Asynchrónne logovanie
- Citlivé dáta
- Include Stack Trace pri logovaní výnimiek
- Strojovo spracovateľné logy

log4j2

- Appenders (handlers)
  - log4j2.xml

Asynchrónne logovanie

- Vyššia priepustnosť
- Nižšia latencia

- -- Error handling
- -- pozor na mutable správy

Quiz time