AppWorks Export/Import tool

Technical Architecture

# **M**odel **V**iew **C**ontroller

Also called MVC and used in frameworks like JavaFX and Spring boot

We choose JavaFX as it is lightweight and easy to use with these principles:

* *Model:* The application specific domain objects
* *View:* The FXML files with a separate designer tool
* *Controller:* The glue between the model/view and GUI behavior for user interaction

## References

* <https://openjfx.io/>
* <https://www.tutorialspoint.com/javafx/index.htm>

# **I**nput/**O**utput to storage

The new way for handling files is the New Input/Output (NIO) of the default Java API. It works with channels and buffers.

## References

* <http://tutorials.jenkov.com/java-nio/index.html>
* <https://www.tutorialspoint.com/java_nio/index.htm>
* <https://docs.oracle.com/javase/8/docs/api/java/nio/package-summary.html>

# Parsing JSON/XML

The tool will work with data in JSON/XML format and we want to marshal/un-marshal or serialize/de-serialize this data to Java objects. For XML we can use the well-known JAXB. For JSON we can use Jackson.

As we also work with SOAP (XML) messages is most logic to also work with a XML parser!

## References

* XML (marshal/un-marshal)
  + <https://docs.oracle.com/javase/tutorial/jaxb/intro/index.html>
  + <https://mkyong.com/java/jaxb-hello-world-example/>
  + <https://www.baeldung.com/java-xml>
  + <https://www.baeldung.com/jaxb>
* JSON (serialize/de-serialize)
  + <https://www.tutorialspoint.com/json/json_java_example.htm>
  + <https://ourcodeworld.com/articles/read/836/how-to-work-with-json-easily-in-java>
  + <https://www.baeldung.com/java-org-json>
  + <https://www.baeldung.com/jackson-object-mapper-tutorial>
  + <https://www.tutorialspoint.com/jackson/index.htm>

# Read property files

Standard feature of Java in the java.util package. The property file I/O can be handled via a NIO Channel with this sample:

Properties properties=new Properties();

try(final FileChannel channel = FileChannel.open(filePath, StandardOpenOption.READ);

final FileLock lock = channel.lock(0L, Long.MAX\_VALUE, true)) {

properties.load(Channels.newInputStream(channel));

}

## Reference

* <https://www.tutorialspoint.com/java/java_properties_class.htm>
* <https://stackoverflow.com/questions/42167828/in-java-using-the-java-nio-library-and-a-filechannel-how-can-i-load-a-properti>
* <https://www.tutorialspoint.com/java/java_properties_class.htm>

# SOAP service calls

JAX-WS is the way to go!

Works together with JAXB too and you can even map SOAP to a java class (see references!). Don’t know yet if we need this last feature!?

## References

* <https://www.baeldung.com/jax-ws>
* <https://www.javatpoint.com/jax-ws-tutorial>
* <https://stackoverflow.com/questions/15940234/how-to-do-a-soap-web-service-call-from-java-class>
* <https://examples.javacodegeeks.com/enterprise-java/jws/jax-ws-jaxb-example/>
* <https://stackoverflow.com/questions/48552497/jaxb-mapping-soap-to-java-classes>

# Mail template

Apache FreeMarker uses the FTL language to define templates (also for mail)

## References

* <https://freemarker.apache.org/>
* <https://www.codejava.net/java-ee/javamail/send-e-mail-in-html-format-using-javamail-api>
* <https://mkyong.com/java/java-how-to-send-email/>
* <https://memorynotfound.com/spring-mail-sending-email-freemarker-html-template-example/>