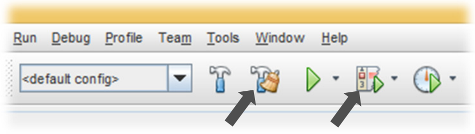
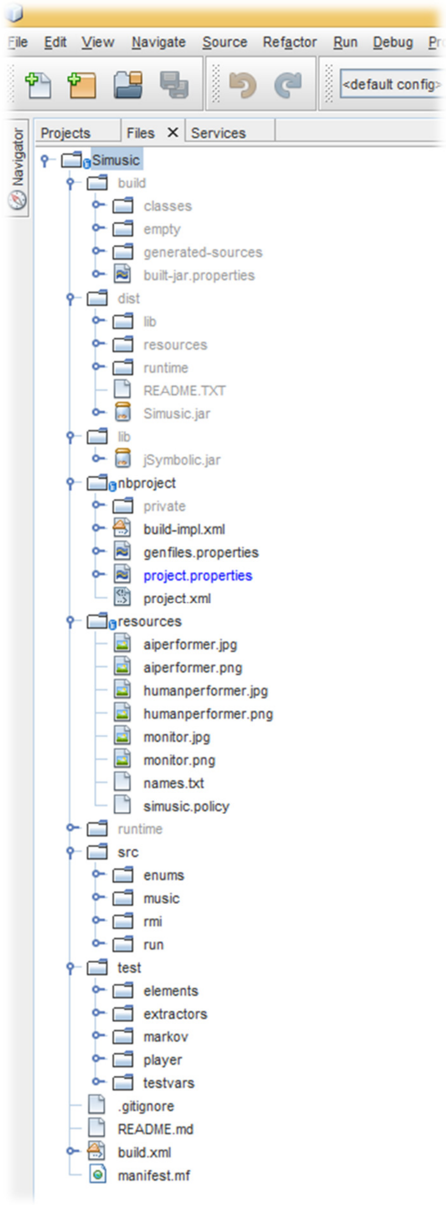
Appendix B: Maintenance manual

Installation and compilation

- The system is a Java Application. In order to install it, simply copy the **./dist** folder to a preferred location. As long as the internal structure of the folder remains intact, the application should run on every computer with Java 8 installed.

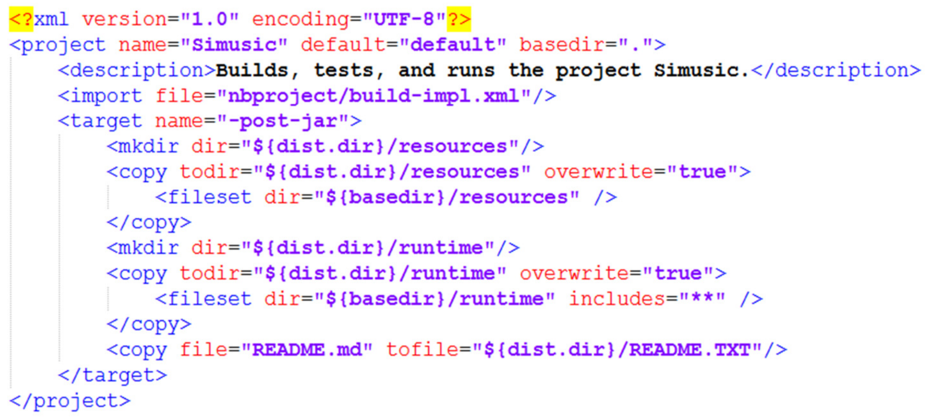


- To compile the source files and re-build the (already built) project files in the dist folder, the host computer needs both NetBeans 8.0.2 and Java 8 installed.
- Open the project folder as regular project in NetBeans. Select Run > Clean and Build Project. The only dependency is the jSymbolic library which is included in the **./lib** folder. It can also be downloaded from the project webpage:
- URL: http://jmir.sourceforge.net/jSymbolic.html



The NetBeans Files view

- If the Files window is not visible, open it from the Window menu.
- Items coloured in gray are ignored by GIT – these are either build/distribution files, external libraries or runtime data (mostly MIDI or temporary test files).
- Items coloured in **blue** are monitored by GIT files that have been modified since the last commit.
- The **build.xml** file in the project root dir contains instructions followed by **Ant** (the NetBeans' native build system). In order to include files and folders in the build from the main project folder, these need to be describes in the build.xml file:

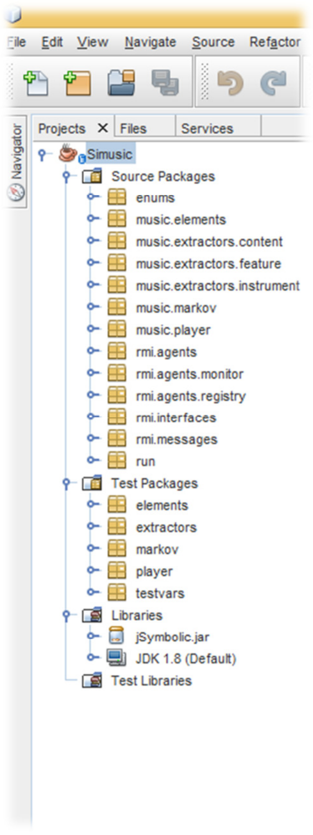


```
<?xml version="1.0" encoding="UTF-8"?>
<project name="Simusic" default="default" basedir=".">
    <description>Builds, tests, and runs the project Simusic.</description>
    <import file="nbproject/build-impl.xml"/>
    <target name="-post-jar">
        <mkdir dir="${dist.dir}/resources"/>
        <copy todir="${dist.dir}/resources" overwrite="true">
            <fileset dir="${basedir}/resources" />
        </copy>
        <mkdir dir="${dist.dir}/runtime"/>
        <copy todir="${dist.dir}/runtime" overwrite="true">
            <fileset dir="${basedir}/runtime" includes="**" />
        </copy>
        <copy file="README.md" tofile="${dist.dir}/README.TXT"/>
    </target>
</project>
```

- The **manifest.mf** file stores information about the files contained in the JAR file.
- The **./nbproject** folder contains files maintained by NetBeans such as the project.properties and project.xml files which describe dependencies.
    - In the project.propertiesfile, the line
      `file.reference.jSymbolic.jar=lib\\jSymbolic.jar`
      specifies a package dependency located in the **./lib** folder
    - The line
      `run.jvmargs=-Xmx1536m -Djava.security.policy==resources/simusic.policy`
      sets the JVM arguments:
        - Specify maximum memory heap of 1,5GB. The host computer needs at least 2GB memory installed.
        - Specify the RMI remote policy file.
- The file nbproject/project.xml specifies where the source and test packages are located.
- The **./resources** folder contains the RMI's simusic.policy file This folder is monitored by GIT.
- It also contains image files used by the GUI layer of the program. They are downloaded from the website http://tech-kid.com/ whose authors claim that "we do not own the copyright to any of the images on this website they are provided as-is".
- The **./runtime** folder contains MIDI files and XML files generated by jSymbilic during runtime. All MIDI files included in the project are freely redistributable and available to download from many online sources.

The Project view:

- If the Projects window is not visible, open it from the Window menu. This window provides a developer-friendly source and test package list as well as any external library dependencies.
- For detailed information about packages and classes, refer to the project class map in section 3.3 Implementation.



```
Simusic
  Source Packages
    enums
    music.elements
    music.extractors.content
    music.extractors.feature
    music.extractors.instrument
    music.markov
    music.player
    rmi.agents
    rmi.agents.monitor
    rmi.agents.registry
    rmi.interfaces
    rmi.messages
    run
  Test Packages
    elements
    extractors
    markov
    player
    testvars
  Libraries
    jSymbolic.jar
    JDK 1.8 (Default)
  Test Libraries
```

Warnings

- Is jSymbolic encounters an error, it terminates the program with System.exit(-1) and there is no mechanism for catching the exception. Check the console if Simusic closes unexpectedly.

Future adaptations and extensions

- The class structure design of the system heavily maximizes system scalability and allows for productive programming flow while implementing and debugging new features due to the high-standard self-documenting code. Some of the planned features are described in section 4 Evaluation.