InterBit
Consulting & Training ltd.
מקבוצת א.מ.ת מיחשוב

Workshop Exercises for Ant 1.6 Course

General instructions:

1. Read instructions thoroughly.
2. Read instructions thoroughly!
3. Follow each chapter assignments step-by-step.
4. Before starting, create a new directory for all workshop's exercises. This directory will be referenced to as the *build root directory* or *master build directory*.

Chapter 3 - Installation

Exercise 1 - Create a 'Hello World' build file

1. Create a build file named build.xml under master build directory
2. Define an empty target called 'hello'
3. Make that target default project target
4. Add an <echo> task to the 'hello' target that prints out a welcome message
5. Launch ant, make sure the message is printed out

Exercise 2 - Debug Ant

1. Repeat step 5 of the previous exercise, this time with the *-debug* option. Get to know and try to understand the output, as you may find it useful in future, more complicated, "real life" builds.

Exercise 3 - Using <javac> and <jar> tasks

1. Create a subdirectory called 'src' under your build-root directory
2. Under that directory, write a simple class named HelloWorld in a package called ch3, that prints out a welcome message
3. Add a <javac> task to the default target in your build file, that compiles the source file into a new directory named 'build' (which you would also need to create in advance)
4. Add a <jar> task to the default target, that jars your .class file according to the package/directory conventions

Chapter 4 - Basic Types

Exercise 1 - Externalizing Properties

1. Define another target in your build file called 'initialize'
2. Create a property file named 'build.properties', located in the build root directory
3. Inside the 'initialize' target, load the property file using a <property> element
4. Replace hard-coded file and directory references throughout the build file with properties located in the file you've created in step 2.
5. Run the build, make sure it succeeds with the new properties

Exercise 2: Fine tuning the build process

1. Break the 'hello' target into 2 targets, 'compile' and 'jar'
2. Change the default project target to 'compile'
3. Create target dependency representing the following structure: initialize -> compile -> jar

Exercise 3: More Properties

1. Rename 'source' directory to 'src'
2. Launch the build using the -D command line option to override your source location property to point to the right directory
3. Make the 'compile' target execute only if the 'javac.enable' property is set and test
4. Disable jar creation if the 'jar.disable' property is set and test

Chapter 5 - Tasks

Exercise 1: Separating build into modules

1. Move all previous chapters' files to a new subdirectory named 'common', located under the build root directory
2. Under the build root directory, create another build file called build.xml

3. Add a default target to the new build file, that launches ch3_4 module's build using the <ant> task.

Exercise 2: Creating a new web module

1. Under the build root directory, create another module directory named 'webmodule'
2. Under that directory, create the following directory structure:

```
webmodule
¦   build.xml
¦   build.properties
¦
+---src
¦   +---main
¦   ¦   +---ch5
¦   ¦           WelcomeContextListener.java
¦   ¦
¦   +---web
¦   ¦       index.jsp
¦   ¦
¦   +---resources
¦       +---WEB-INF
¦               web.xml
¦
+---output
    +---main
    +---war
    +---lib
```

3. Add html content to index.jsp that prints a welcome message when accessed via browser
4(*). Add a simple ServletContextListener class under 'main' directory, that prints a welcome message when ServletContext is initialized. Register that listener in web.xml using <listener> elements

Exercise 3: Building the web module

In the build file located under 'webmodule', write the following targets, each depending on previous one:

1. 'clean' - deletes output directory
2. 'initialize' - reads property file 'build.properties' containing all location properties
3.(*) 'compile' - compiles the listener class into 'output/main' directory (only if task 4 in exercise 2 was accomplished). Notice: in order to complete compilation you will need a file called servlet-api.jar located under jboss 3.2.3 directory.
4. 'jar' - packs the listener class in a jar archive(*)
5. 'war' - creates a .war archive in directory 'output/dist', containing all web resources in the correct structure:
6. 'deploy' - copies the created war file to jboss's auto-deployment directory(jboss3.2.3\server\default\deploy)

WAR content:

```
web.war
¦   index.jsp
¦
+---WEB-INF
    ¦   web.xml
    ¦
    +---lib
            listener.jar
```

7. Run jboss (jboss3.2.3\bin\run.bat) and enjoy wonders of creation.

Exercise 4: More Tasks

1. Add a task to the war target that sends a mail upon successful completion of the build
2. Add a target named 'javadoc' that generates api documentation for the listener class into directory 'output/docs/api'
3. Add another <ant> task to the master build file(located in the build root directory) that launches the creation of your Javadocs.
4. Add a task to the 'initialize' target, that fails the build if the operating system is windows

Chapter 6 - File System Types

Exercise 1:

1. Under the master build directory, using the <mkdir> task, create a runtime enviroment directory for all our applications
2. Copy all build artifacts to that directory
3. Enhance your master build file to use the <subant> task(you will need to rename some of the targets in the per-module build files)

Exercise 2: Source distribution

1. Add to your master build file a target named 'dist-source', which collect your sources from all modules and packs them into archives, one per module.

Chapter 7 - Advanced File System Types

Exercise 1: File Comparisons

In the master build file create the following targets:
1. 'compare-src' target, which generates a zip file containing all source files changed from current source distribution to another one, specified with a -D command line option
2. 'source-dist-' target, which creates a zip file containing all source files changed from from a specific date, based on command line option argument

Chapter 8 - Condition and references

Exercise 1: Enterprise tasks

Add the following targets to the master build file, that are dependent upon successful build of all modules:
1. 'run-server' target, which starts jboss server(use parallel and wait for)
2. 'test' target, which verifies that the web application can be accessed
3. 'run-nightly' target, which executes the previous 2 targets

Chapter 9 - Custom Components

Exercise 1: Custom mappers

1. Create a custom mapper which maps filenames with .abc extensions to .xyz extensions
2. Create a build file which builds the mapper and packs it into a distributable jar
3. Write another build file which uses the custom mapper

Exercise 2: Custom Tasks

1. Create a custom task which prints out all project details
2. Create a build file which builds the task and packs it into a distributable jar
3. Write another build file which uses the custom task

Exercise 3: Custom Conditions

1. Create a custom condition which evaluates to true if the directory supplied as an argument exists and contains any graphic files(.gif, .jpg, .jpeg, .png)
2. Create a build file which builds the condition and packs it into a distributable jar
3. Write another build file which uses the custom condition