Net01

Quick Documentation - **DRAFT**

# Introduction

First familiarize yourself with the folder structure:

bin

contains various scripts/wrappers for the toolkit – most of them require a compiled version of the toolkit

build

the generated code during compilation

dist

the generated jar files – the compiled toolkit itself

docs

various data examples and documentation

exercises

working place for storing the question-sets you create

lib

possible 3rdparty libraries

src

obvious

# Building the source

Before you start using the toolset, writing exercises or so, you first need to build the tools.

## Prerequisites

You will need the Oracle Java SDK (any SDK version above v1.5 should do)

and Apache Ant (any recent version should do also)

## Step1: Setting up the environment

Open a console terminal on Linux (or command prompt on Microsoft Windows) and place yourself into the folder where build.xml file exists.

## Step 2: Clean building space

ant clean

## Step 2: Build the toolset

ant compile

Now you have your tools …resting in dist folder and you are ready to proceed with your exercises.

Each time you want to redistribute it with new question-sets, you will need to recompile it with the question-sets prepared and placed using the following instructions.

# Creating the exercises

For each question-set we basically need two (2) files: The original question-set (the one you initially create – but later it will be replaced) and the distribution file (the one the students will need).

The “student” version of questions is a stripped version of the initial question-set containing only the id and the text of each question but not the right answers.

## Writing your question-set

You will find two (2) sample question-sets in docs folder, example-exercise-complete.xml and example-exercise-basic.xml.The DTD schema of each question-set is located at src/data/exercise.dtd, though it’s a superset of the current implementation – better stick to the examples.

Pick an example and start modifying – for now, the critical nodes you have to pay attention are:

**exercise/@id**

This is a sequence id of the question-set and will be matched to student’s answers

**exercise/@mchoice**

The number of questions should be presented in each examination session. For instance, you may want t10 random questions out of a total set of 80 - @mchoice should be 10.

**exercise//question//answer/@correct**

Only one answer per question should have a @correct attribute with a value of «true». Any other value or absence of the @correct attribute, marks the answer as incorrect.

## Compiling the question-set

Note: You should first have compiled the toolset.

Each question-set must be prepared for distribution. Thus, the original exercise files should be checked and prepared with unique question ids. The tool compiling the question-sets is the class named QuestionBuilder in org.exadmin.tools package.

As an example of compiling a question-set could be the following command:

java -cp net07.jar org.exadmin.tools.QuestionBuilder exam.xml

where net07.jar is our toolset and exam.xml the original question-set you have created.

The above will produce three (3) files named exam-ids.xml, exam-distribution.xml, exam-distribution.enc in exam1.xml’s folder.

Note: There are two (2) batch files in bin folder named net07-QuestionBuilder.[bat|sh] for that purpose.

The exam–ids.xml file is the complete question-set file with the questions, answers and the generated –if necessary – ids for each question. This file should be placed in src/org/exadmin/data

The exam-distribution.xml and exam-distribution.enc files are the question-set files for the student and they contain only the questions and their ids. These files should be placed in src/data. Depending on your configuration preferences, you will need only one of these (read later about src/data/configuration.xml)

**Attention**

If you need to modify a previously compiled version of a question-set (one with generated IDs), you should work on a copy of the x-ids.xml file.

In general, the sequence of the process should be:

[initial file] 🡪 QuestionBuilder 🡪 [xx-ids.xml] 🡪 copy [xx-ids.xml] over [initial file] - - 🡪

## Configuring the distribution

When you have completed building your question-sets and placing the result files to the appropriate folders, you need to configure the toolset. You can configure which question-sets you want to activate, their due-times and the global submission method.

The configuration file is locate at src/data/configuration.xml (Sorry, no schema right now)

Additionally you will need to define which students are allowed to proceed with the exams, modifying the list src/data/users.list

### Exercises

Each question-set is configured with the **file** element under the **exercise-configuration** root:

@name

The relative to src path name of the x–distribution.[xml|enc] file

@subname

The relative to src path name of the x–ids.xm file

@start

Starting date of question-set’s active period. Formatted as “YYYYMMDD hhmmss”

@end

Ending date of question-set’s active period. Formatted as “YYYYMMDD hhmmss”

@encoded

Set to «yes» if the @name is the encoded version of the x-distribution file.

### Submission Methods

There are currently two submission methods supported, the file-writer and the url-writer

The file-writer submission method, writes the answers into the local path defined by it’s @path attribute – users should have write access to this folder.

The url-writer submission method, sends the answers through a url defined in it’s @url attribute.

In bin folder you can find an example perl processor named HTTPSubmit.pl that can be executed as a cgi on the remote web server.

### Student List

The student list file src/data/users.list is a comma separated list consisting of

Student’s ID, Student’s UserID, Student’s Full Name

Note: Student’s UserID will be the value read from USER environment variable (see later about execution by the student)

# Distribution

After following the above instructions about writing and placing the question-sets in the source tree, you need to rebuild the toolset. An ant compile in build.xml folder should do.

Student’s need only the exercises.jar file

Professors need only the net07.jar and the list of answers in xml format students submit.

# Execution

## (by the student)

Students need only the exercises.jar file. Examine net07 or net07.bat files in bin folder to see how it is called.

Make sure that the environment variable *USER* is set correctly prior to exercises.jar execution.

The engine will try to find the value of the *USER* environment variable in src/data/users.list (Student’s UsedID column) and will match the first entry found. Exercises.jar will fail if no entry found.

You might need to fix permissions on submission folder, if *file-writer* is configured as the submission method

## (by the professor)

Professors need then net07.jar file. Examine the net07-AnswerChecker.[bat|sh] in bin folder and the java class it calls to see how to use it to check submitted answers.

A sample report can be generated – Examine the net07-Report in bin folder and the java class it calls.

Table 1

|  |  |
| --- | --- |
| *net07.jar* | *student.jar* |
| ├── data  │   ├── configuration.xml  │   ├── sample-distribution.enc  │   └── users.list  ├── META-INF  │   └── MANIFEST.MF  └── org | ├── data  │   ├── configuration.xml  │   ├── sample-distribution.enc  │   └── users.list  ├── META-INF  │   └── MANIFEST.MF  └── org |
| ├── exadmin  │   ├── data  │   │   ├── report-submits.xsl  │   │   └── sample-ids.xml  │   ├── Main.class  │   └── tools  │   ├── AnswerChecker$1.class  │   ├── AnswerChecker.class  │   ├── AnswerCheckerOld$1.class  │   ├── AnswerCheckerOld.class  │   ├── HttpsPost.class  │   ├── QuestionBuilder.class  │   ├── ReportBuilder$1.class  │   ├── ReportBuilder.class  │   └── Report.class | │  │  │  │  │  │  │  │  │  │  │  │  │  │  │ |
| └── exercises  ├── Answer.class  ├── configuration  │   ├── Configuration.class  │   ├── ConfigurationParameters.class  │   ├── ExamFileParams.class  │   └── SubmissionEnums.class  ├── console  │   └── Console.class  ├── Exercise.class  ├── ExerciseResult.class  ├── ExerciseResultComparator.class  ├── log  │   ├── LogEntry.class  │   └── Logger.class  ├── Main.class  ├── Question.class  ├── students  │   ├── Student.class  │   └── StudentList.class  ├── submission  │   ├── FileSubmission.class  │   ├── HttpSubmission.class  │   └── Submission.class  └── util  ├── Base64.class  ├── EncryptEngine.class  ├── StringUtil.class  └── XMLUtil.class | └── exercises  ├── Answer.class  ├── configuration  │   ├── Configuration.class  │   ├── ConfigurationParameters.class  │   ├── ExamFileParams.class  │   └── SubmissionEnums.class  ├── console  │   └── Console.class  ├── Exercise.class  ├── ExerciseResult.class  ├── ExerciseResultComparator.class  ├── log  │   ├── LogEntry.class  │   └── Logger.class  ├── Main.class  ├── Question.class  ├── students  │   ├── Student.class  │   └── StudentList.class  ├── submission  │   ├── FileSubmission.class  │   ├── HttpSubmission.class  │   └── Submission.class  └── util  ├── Base64.class  ├── EncryptEngine.class  ├── StringUtil.class  └── XMLUtil.class |