

# User Guide - Exercise catalogs in Extensible Markup Language

The Ävaders

July 14, 2016

## 1 Introduction

The following document describes the available tags and the general structure of exercise catalogues, written in XML, that are accepted by the catalogue loader included with this user guide.

Additionally simple troubleshooting tips and contact information are provided in case further help is needed.

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Available tags</b>	<b>3</b>
<b>3</b>	<b>General structure of an exercise catalogue</b>	<b>5</b>
<b>4</b>	<b>Troubleshooting</b>	<b>9</b>
<b>5</b>	<b>Where to find further help</b>	<b>9</b>

## 2 Available tags

In this section the xml-tags the catalogue loader recognizes, as well as their general use are described.

- exercises** Use this mandatory start- and end-tag to mark the beginning and end of the exercises you want to declare.
- exercise** Utilize this as start- and end-tag **as content of an exercises tag** to declare the beginning and end of a single exercise.

You may only use the following tags **as content of an exercise tag**:

- description** Use this start- and end-tag to attach a description to an exercise.
- classes** Utilize this start- and end-tag to mark the beginning and end of the content that should be interpreted as classes.
- tests** Use this start- and end-tag to declare the beginning and end of the tests you want to declare.
- config** Make use of this tag if you want to configure the parameters of the exercise this tag appears in.

You may only use the following tag **as content of a classes tag**:

- class** Surround content with this start- and end-tag to mark it as template for a java class.  
You are able to set a name for each class by attaching a **name**-attribute to the tag.  
**Please Note** that the name-attribute and the class name in the source code have to match.

You may only use the following tag **as content of a tests tag**:

**test** Surround content with this start- and end-tag to mark it as a test-class.

You are able to set a name for each test by attaching a **name**-attribute to the tag.

**Please Note** that the name-attribute and the class name in the source code have to match.

**Please Note** each test-class has to have a corresponding name to a class, i.e. a class named "Example" has a test-class named "ExampleTest".

You may only use the following tags as content of a **config** tag:

**babysteps** Add this optional empty-element tag to manipulate whether or not babysteps is activated and the time until until it's effect takes place.

You are able to do so by setting the attribute **value** to *"true"* or *"false"* and assigning a string to the **time**-attribute in the format: *"m:s"*, where *m* are minutes and *s* are seconds.

Leaving out either attribute will set them to their default, which is *"false"* for value and *"2:00"* for time.

**timetracking** Use this optional empty-element tag and set it's **value**-attribute to *"true"* or *"false"* to enable or disable tracking of statistics in this exercise respectively.

It's default value is *"false"*.

**atdd** Add this optional empty-element tag and a **value**-attribute with *"true"* or *"false"* assigned to activate or deactivate this feature.

Skipping this tag will default atdd to *"false"*.

**comments** May appear anywhere in a document outside other markup. To use the comment tag simply surround the symbols you wish to be a comments with *"<!--"* and *"-->"*.

### 3 General structure of an exercise catalogue

This section describes, with examples, the structure of an exercise catalogue. It could be helpful to refer to this section while writing a catalogue as it goes through the necessary tags step by step.

1. **Initialization** Initialize an **exercise catalogue** by writing

```
<exercises>
</exercises>
```

into a file with a **".xml"-extension**.

2. **Adding an exercise** Add **exercises** with the name "example1" through "exampleN" to the empty exercise catalogue by writing:

```
<exercises>
  <exercise name = "example1">
  </exercise>
  .
  .
  .
  <exercise name = "exampleN">
  </exercise>
</exercises>
```

3. **Adding a description** Attach a **description** to the exercise named "example" by adding

```
<exercises>
  <exercise name = "example">
    <description>
      This is an example for a description
    </description>
  </exercise>
  .
  .
  .
</exercises>
```

to the file.

4. **Adding classes** Add one or more **classes** to the exercise "example" by writing:

```
<exercises>
  <exercise name = "example exercise">
    <description>
      This is an example for a description
    </description>
    <classes>
      <class name = "ExampleClass1">
```

```

        public class ExampleClass1{
        }
    </class>
    .
    .
    .
    <class name = "ExampleClassN">
        public class ExampleClassN{
        }
    </class>
    <classes>
</exercise>
</exercises>

```

**5. Adding tests** Add one or more **tests** to the same exercise by declaring them as follows:

```
<exercises>
  <exercise name = "example exercise">
    <description>
      This is an example for a description
    </description>
    <classes>
      <class name = "ExampleClass1">
        public class ExampleClass1{
        }
      </class>
      .
      .
      .
    </classes>
    <tests>
      <test name = "ExampleClass1Test">
        import static org.junit.Assert.*;
        import org.junit.Test;
        public class ExampleClass1Test{
          @Test
          public void testExample(){
          }
        }
      </test>
      .
      .
      .
    </tests>
  </exercise>
  .
  .
  .
</exercises>
```

**6. Configuring the exercise** Configure the exercise to your liking by adding the config-tag to the file:

```
<exercises>
  <exercise name = "example exercise">
    <description>
      This is an example for a description
    </description>
    <classes>
      <class name = "ExampleClass1">
        public class ExampleClass1{
        }
      </class>
      .
      .
      .
    </classes>
    <tests>
      <test name = "ExampleClass1Test">
        import static org.junit.Assert.*;
        import org.junit.Test;
        public class ExampleClass1Test{
          @Test
          public void testExample(){
          }
        }
      </test>
      .
      .
      .
    </tests>
    <config>
      <babysteps value="true" time="3:15"/>
      <timetracking value="false"/>
      <atdd value="true"/>
    </config>
  </exercise>
  .
  .
  .
</exercises>
```



## 4 Troubleshooting

Why do i get a **"Line \_ expected  
----- but did not find it"** error  
message?

Most likely is that a symbol like = or  
" in a tag or the whole tag (usually  
the end-tag) itself are missing in the  
xml-file.

Check the xml-file to see if some-  
where one or more symbols or tags  
are missing.

Why do i get a **"Line \_ expected  
property: -----, but found -----  
instead"** error message?

This usually indicates that a specific  
attribute or tag was expected, but  
something else found.

Refer to 2 to see which tags and at-  
tributes are supported in this release,  
to help find the tag which is causing  
this error.

What does the **"Please choose a  
file with the ".xml"-extension"**  
error message mean?

If you get this error, the parsed file  
does not have the correct (meaning  
the ".xml"-) extension.

Either choose a file with the ".xml"-  
extension or change the file's exten-  
sion to the correct one to fix this.

What does the **"The specified  
catalogue does not contain an  
exercise"** error message mean?

If you get this error, the parsed file  
contains no exercises.

Fix this by adding at least one exer-  
cise with a class and test to the xml-  
file. Refer to 2 and 3 to help with  
this task.

What do I do when I get a **"Exer-  
cise name, classes, tests or con-  
fig missing in file"** error message?

This error occurs if one or more of  
the mentioned elements are missing  
in the parsed xml-file.

Refer to 2 and 3 and compare them  
carefully to the file to spot the miss-  
ing tag or attribute and add it at the  
appropriate position.

## 5 Where to find further help

If you need further help please contact one of the contributors at the  
The Ävaders github repository.