User manual of JTDE - Java Test Data Editor

Content

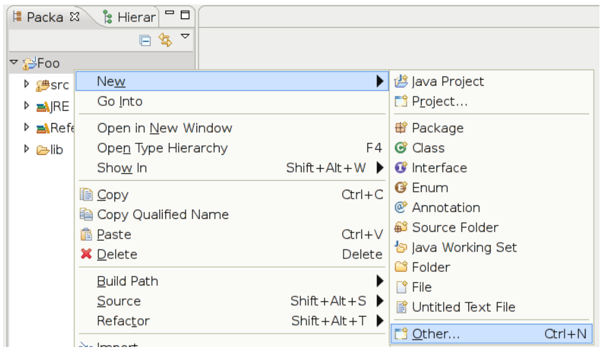
Use case

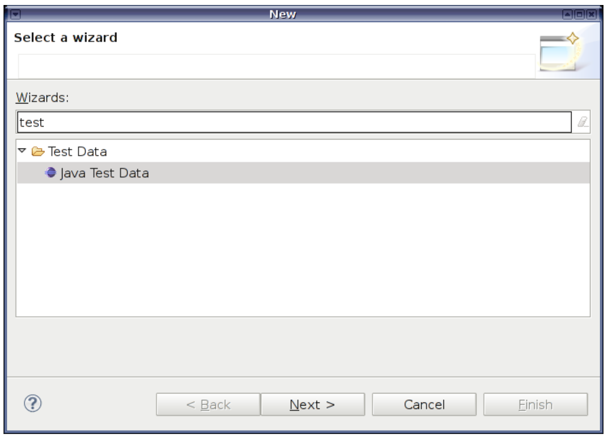
The main purpose of this editor is to create/edit test data persisted in XStream files. Outsourcing test data allows cleaner test code cause it has not to deal with setting up data objects. You may use it also to edit/adapt existing XStream records.

Installing the plugin

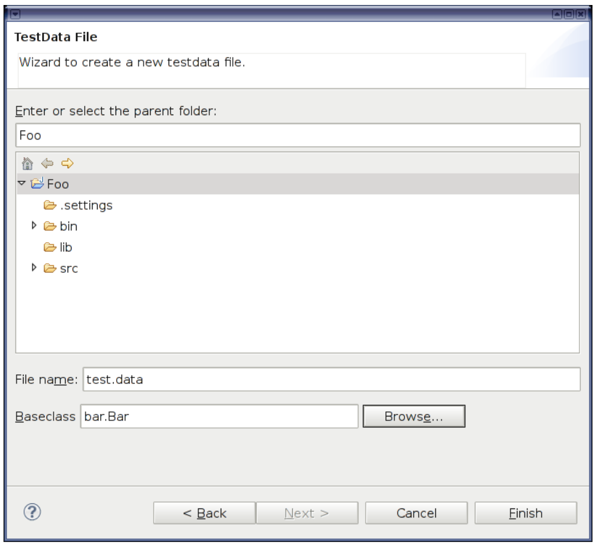
- Close Eclipse
- Copy ch.jtde.xyz.jar to <Eclipse home>/dropins
- Start Eclipse (maybe with -clean)

Create test data file





Wizards: test



Wizard to create a new testdata file.

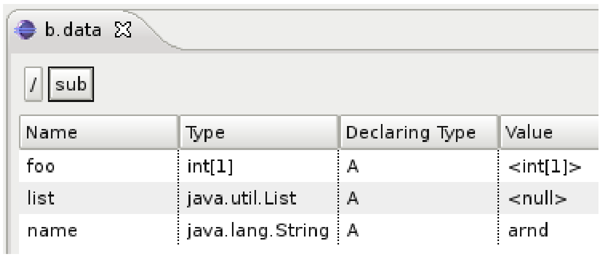
Enter or select the parent folder: Foo

File name: test.data

Baseclass bar.Bar

- **Parent folder:** Target folder for the file
  - The classpath of the parent project will be used to retrieve the required information about classes – all classes to be used in the test data file must be present in this classpath!
- **File name:** Name of the file
- **Baseclass:** Class for which you want to create test data
  - This must be a concrete class (excepting for Collection types) which have to be present on the projects classpath

The editor

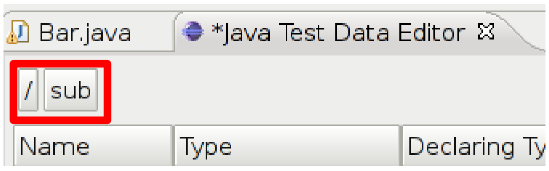


Name	Type	Declaring Type	Value
foo	int[1]	A	<int[1]>
list	java.util.List	A	<null>
name	java.lang.String	A	arnd

Columns

- **Name:** Name of the field or the key of an map entry
- **Type:** Type of the field
- **Declaring Type:** The class owning this field
- **Value:** String representation of the value
  Editable for 'atomic' types (i.e. strings and primitive types)
  - Shows the concrete type name for non 'atomic' types

Button path



- The path starting at the object root toward the currently displayed element (analogous to Thunar/Nautilus)
- Click on a button to jump back to the represented element

General

- Double click a row to create an instance for a nulled attribute
- Double click a row holding an instance to edit it (steping into it)
- Click on the value cell to edit 'atomic' values (like strings and primitive types)

Create an 'instance'

- Double click the row of the field
- Choose the concrete type for the instance(only for non final types)
  - Type-dialog filters for assignable types
  - Abstract classes/interfaces were not supported by default (but you may write an extension for them...)t
    - Currently there is one exception for collection types
- The editor jumps into the newly created instance

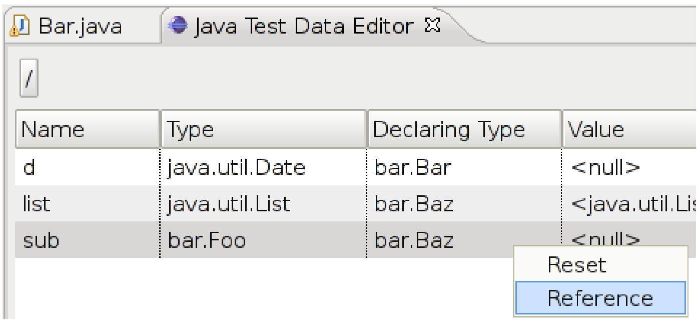
Context menu

Further functionality may be found in the context menu of a field.

Create a reference to an instance

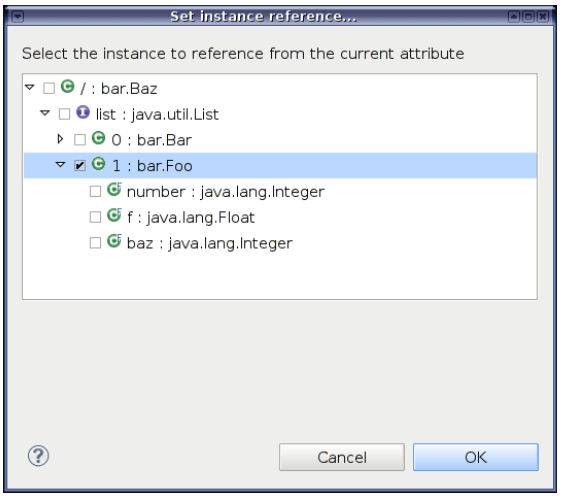
An instance may be referenced by several fields – this is also supported by the editor.

How-to



Name	Type	Declaring Type	Value
d	java.util.Date	bar.Bar	<null>
list	java.util.List	bar.Baz	<java.util.Lis
sub	bar.Foo	bar.Baz	<null>

Choose reference in a fields context menu (only available for object types!)


Set instance reference...
Select the instance to reference from the current attribute
Cancel
OK

```
/ : bar.Baz
  list : java.util.List
    0 : bar.Bar
    1 : bar.Foo
      number : java.lang.Integer
      f : java.lang.Float
      baz : java.lang.Integer
```

The populated fields of the whole data structure were shown – now you can choose the instance to reference.

How to use the test data

XStream

Cause the data is written in the XStream format your project must contain XStream.
Just add XStream and it's XML parser XPP3 to the classpath.
XStream and further information can be found on http://xstream.codehaus.org/.

Parse

As it's the normal XStream format you can simply deserialize the data using:

```
Object data = new XStream().fromXML(new FileInputStream("foo.data"));
```

The rest of the test code can be written as usual.