Table of Contents

During the software development process, whether it is for an application or a service, functions are always used that make the developer's work easier. These functions are usually saved in utility or helper classes so they can be reused.

JVx also uses various utility classes, which can be found in the package com.sibvisions.util.

XML Editing

An XML parser is required to read XML files. Various frameworks exist that allow reading and saving of XML files. However, accessing the individual elements is often solved tediously using object trees.

JVx contains the classes XmlWorker and XmlNode, which allow direct access to tags. In addition, attributes can be accessed like sub tags. Performance and memory use are also very low.

In the following example, we read an XML file, change a sub tag, and save the result as a new XML file:

```
XmlWorker xmw = new XmlWorker();

XmlNode xmnRead = xmw.read(new FileInputStream("example.xml"));

xmnRead.setNode("/archive/element(1)/user", "xml");

xmw.write(new FileOutputStream("example_v2.xml"), xmnRead);
```

Find more details about XML Handling.

Working With Arrays

Array operations are always tedious coding work, and every developer could easily do without. Removing a single element from an array is not rocket science, but there is no standard function available for this task. And who would not like to search an array for an element without coding a loop?

With ArrayUtil significant array operations were implemented, which also fully include the list interface. We gain performance compared to standard lists as an added benefit.

The Object Cache

From time to time objects have to be cached and exchanged between various instances. Parameters and public methods are very useful for this purpose, but we do not always want to pass an object on across countless instances to the target. Also, the required objects are not always located in the same scope.

This is what the ObjectCache is for: to handle objects statically, for certain time periods or permanently, within a VM.

This can be compared to a static hash table complemented by a timeout.

Ordered Hashtable

The standard hashtable manages its keys in an undefined order. Sometimes it can be useful to keep the order in which the data was added.

This is handled by the OrderedHashtable.

Hashtable With Automatic List Management

The standard hashtable manages exactly one value per key. However, often more than one value per key has to be stored. In this case, a list has to be used as a value in which a number of individual values are saved. This is not particularly difficult but a tedious and boring task nonetheless.

The KeyValueList handles the administration of lists and leaves the developer time for more important tasks.

Reflective

The reflective class allows you to create class instances or call methods via reflection but without directly using `java.lang.reflect`.

For more details, look here.

More Classes

There are many other interesting utility classes, such as StringUtil, BeanUtil, DateUtil, FileUtil, ImageUtil, …

Additional information can be found in javadoc.