Software Documentation : Topcoder Commons Utility Design

This page last changed on Apr 19, 2011 by irabbit.

1. Scope

1.1 Overview

Helper/Utility class is widely used in Component Development Competitions.
Most components use the similar methods.
e.g. checkNull, checkNullOrEmpty, logEntry, logExist, logExecption.....

Developers have to include it in most components and write unit tests for the Helper.
We want to extract the frequently used methods to Topcoder Commons Utility Component.

1.1.1 Version

1.0

1.2 Logic Requirements

The design should cover following functions.

1.2.1 Parameter Checker

Object: null

File: null, exists, isFile, isDirectory

String: null, empty

Numeric: negative, positive, non-negative, non-positive, zero, (a, b), (a, b], [a, b), [a, b].

Collection: null, empty, null element, contains empty String/Collection/Map

Map: null, empty, contains null key, contains empty String/Collection/Map key, contains null value, contains empty String/Collection/Map value

1.2.2 Logging Utility

Supported logging libraries:

Topcoder Logging Wrapper 2.0.0, Log4j 1.2.16.

Scope:

Method Entry, Method Exit, Parameters, Exception.

Note: logException should return the exception. So the user could use it by following ways.

throw Log4jLoggingHelper.logException(logger instance, method name, class name, exception, timestamp). (the class/method declaration is just the sample)

1.2.3 Configuration Utility

Supported configuration files:

Properties file.

Scope:

Read single/multi value.

Read, parse validate the values. Numeric value, String, Date and Time.

1.2.4 Database Utility (JDBC)

Query, validate and parse the value from ResultSet.

1.2.5 Others (Enhancement)

Please confirm the enhancement (e.g. new useful utilities) in forum or by Contact Mangers.

1.3 Required Algorithms

N/A

1.4 Example of the Software Usage

The utility should be easy to use and it shouldn't require any configuration.

e.g.

LogException(logger, exception) // if user just wants to log the exception

LogException(logger, class name, method name, exception) // if user wants to log the exception with class name and method name

LogException(logger, class name, method name, timestamp, exception) // if user wants to log the exception with timestamp

bad design sample:

ExceptionUtils.checkNull(Object item, ResourceBundle bundle, String key, String defaultMessage).

99% usage is ExceptionUtils.checkNull(item, null, null, message).

1.5 Future Component Direction

More utilities can be added.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

N/A

2.1.2 External Interfaces

N/A

2.1.3 Environment Requirements

- Development language: Java1.5
- Compile target: Java1.5, Java1.6

2.1.4 Package Structure

com.topcoder.commons.utils

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

N/A

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

N/A

3.2.2 TopCoder Software Component Dependencies:

None.

3.2.3 Third Party Component, Library, or Product Dependencies:

Apache Log4j 1.2.16 - http://logging.apache.org/log4j/1.2/

3.2.4 QA Environment:

- Solaris 7
- RedHat Linux 7.1
- Windows 2000
- Windows 2003

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines. Modifications to these guidelines for this component should be detailed below.

3.4 Required Documentation

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines.

3.4.1 Design Documentation

- Use-Case Diagram
- Class Diagram
- Sequence Diagram
- Component Specification

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the 'Documentation' tab of TCUML.