Software Documentation : Java Custom Cockpit Studio Contest Manager 1.3

This page last changed on Jul 12, 2009 by mashannon168.

1. Scope

1.1 Overview

This component contains a ContestManager interface and a ContestManagerBean that handles studio contests in persistence. In version 1.3 of this component ContestManager and ContestManagerBean will be updated with a new method.

1.1.1 Version

1.3

1.2 Logic Requirements

1.2.1 Update Contest Manager and its current implementation (ContestManagerBean)

In this new version of the component the ContestManager interface and its ejb implementation, ContestManagerBean, will be updated with a new method. This competition will handle the classes presented in "Studio Contest Manager 1.3 Class Diagram" from uml. Note that in Pipeline Conversion Architecture the following modifications were made to the entities:

- Resource entity now has a name field (to identify a user)

- StudioCompetition now has an array of Resource type (this array identifies the users involved in this project (client, manager, copilot...)) The following method will be added: - getUserContests(username) method will get the contests where the user identified by username is a resource for the contests (new tables defined in Pipeline Conversion Architecture can be used to identify user contests)

Note that the methods from ContestManagerBean that work with ContestData will need to be updated to take into consideration the updates made for ContestData class.

1.2.2 Update diagram

The class diagrams of this component are not up-to-date and this will be resolved in this competition. The competing designers will check the current code and update theclass diagrams of this component. If necessary, the use case diagram and sequence diagrams also need to be updated.

1.3 Required Algorithms

None

1.4 Example of the Software Usage

The ContestManager is used to manage studio contests in persistence.

1.5 Future Component Direction

None

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None

2.1.2 External Interfaces

Design will follow "Studio Contest Manager 1.3 Class Diagram" from uml.

2.1.3 Environment Requirements

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

2.1.4 Package Structure

com.topcoder.service.studio.contest

com.topcoder.service.studio.contest.bean

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

None

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

- Java 1.5+
- EJB 3.0
- WSDL 1.1
- JBoss 4.2
- Hibernate 3.2 and higher

3.2.2 TopCoder Software Component Dependencies:

- **Logging Wrapper 2.0**: Used to log invocation information and exceptions.
- **Base Exception 2.0**: Custom exceptions in this design extend the base exceptions from this component.
- **Contest and Submission Entities 1.0**: It provides the data entities used in this design.
- **Project Service 1.0**: The used Project data entity is from this component.
- **Search Builder 1.3.2**: This is used for the Filter definition which we utilize in the component.

3.2.3 Third Party Component, Library, or Product Dependencies:

None

3.2.4 QA Environment:

- RedHat Linux 9
- Informix 10

- JBoss 4.2
- Hibernate 3.2 and higher

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

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 Poseidon.