HTML Cockpit Specification Review Struts Actions Requirements Specification

Scope

Overview

As we progress in upgrading the Direct application, the next thing we are focusing on is the complete flow of launching a contest. This includes everything from entering the basic data for a contest to payment, spec review, provisioning, etc. We currently have a basic flow for launching a contest created in the service layer and being assembled for the html version of Direct. This contest will dig deeper into the process of launching a contest to address the requirements introduced and clarified by the requirements that are referenced in this contest.
This component provides the struts actions to start specification review, retrieve specification review scorecard for display and resubmit specification review scorecard.

Version

1.0

Logic Requirements

Each of the following actions will be designed as a Struts action with AJAX handling capability. It is up to the designer to make best use of Struts capabilities, including interceptors and base action classes. The only client of this application will be the JSPs, so there are no APIs or structural requirements. The designer is free to refactor any interceptor or action as long as requirements are met.
This component is responsible for several actions that are used to perform requests from the Frontend.

Struts Actions

Here we have some general information about the actions:

- All actions will extend the AbstractAction class from the struts framework component.
- Javabean properties (i.e. getXXX/setXXX) should be used for all action data. The input data will be provided as request parameters and will need to be mapped to these setters using JavaBean conventions.
- The design may rely on some data conversion to take place before the action, such as converting dates, but the designer must specify what those conversions must be.
- The execute method will place all result data in the AggregateDataModel then it will set it to the action to make it available in the ValueStack. Essentially, we expect return parameters to be sent back. Use simple "return" key for the model.

Validation

Validation errors would be packaged in the ValidationErrors entity with each property that fails validation placed in the ValidationErrorRecord entity. ValidationErrorRecord contains a messages array field that allows multiple validation errors to be provided per field (for example, a field could be too long and contain incorrect format).

Copyright TopCoder, Inc. 2008

TCSubject

The TCSubject needs to be retrieved from the session, please make the session key configurable.

Start Specification Review Action

This action will start the specification review for the project (including software and studio projects).
Input:
contestId - the software or studio contest id.
studio - flag indicating the contest is a studio or software contest
startMode - the value can be "now" or "later"
Process
If the startMode is "now", the specification review will be started immediately, otherwise, the specification review will be started 48 hours (make it configurable) prior to the scheduled contest start time.
The SpecificationReviewService.scheduleSpecificationReview method will be called with the contestId and scheduled date. If the startMode is "now", use current date as the scheduled date; otherwise, retrieve the contest's start date from the ContestServiceFacade (via getContest or getSoftwareContestByProjectId method for studio or software contest).
Result:
None

View Specification Review Action

This action will retrieve the specification review scorecard of the project for display.
Input:
contestId - the software or studio contest id.
studio - flag indicating the contest is a studio or software contest
Process
It will call the SpecificationReviewService.getSpecificationReview method to get the specification review for display. It will call the SpecificationReviewService.getSpecificationReviewStatus method to get the specification review status. It will also need to call the SpecReviewCommentService.getSpecReviewComments method to get additional specification review comments for each question in the scorecard.
Result:
The retrieved SpecificationReview, SpecificationReviewStatus and the list of SpecReviewComment

Save Specification Review Comment Action

This action will add or update the specification review comment.
Input:
contestId - the software or studio contest id.
studio - flag indicating the contest is a studio or software contest
questionId - the question to which to add/update the comment
commentId - the id of the comment. Only available when updating a comment
comment - the added or updated comment.
action - can be "add" or "update" value
Validation
The comment must be non-empty string.
Process
If action is "add", it will call the SpecReviewCommentService.addSpecReviewComment method to add the specification review comment. Otherwise, it will call the SpecReviewCommentService.updateSpecReviewComment method to update the specification review comment.
Result:
The added / updated UserComment.

Resubmit Specification Review Action

This action will resubmit the specification review scorecard and move it to the next specification review phase.
Input:

Copyright TopCoder, Inc. 2008

contestId - the software or studio contest id.
studio - flag indicating the contest is a studio or software contest
content - a string injected through Spring
Process
It will call the SpecificationReviewService.updateSpecificationAsString method to resubmit the specification review.
Result
None

Struts action mapping

The designer does not need to provide a mapping file for struts actions. All struts and spring files will be provided by the assembly.

Services

The actions will directly use façade classes. These will be injected.

Logging and Error Handling

The actions will not perform logging and any error handing. If any errors occur, they actions will simply put the Exceptions into the model as a "result". There will an interceptor that will process the logging of this.

Transaction handling

Any transactions will be handled externally by the container at the level of the façade. Nothing needs to be done in this component.

Thread-safety

Since we are operating in a servlet container, threading is not an issue.

Configuration

All configurations will be done via method injection. Each item being injected via a setter will have a corresponding getter.

Required Algorithms

None

Example of the Software Usage

The actions will handle specification review requests.

Future Component Direction

None

Interface Requirements

Graphical User Interface Requirements

None

Copyright TopCoder, Inc. 2008

External Interfaces

None

Environment Requirements

- Development language: Java1.5, J2EE 1.5
- Compile target: Java1.5, J2EE 1.5
- Application Server: JBoss 4.0.2
- Informix 11

Package Structure

com.topcoder.direct.actions

Software Requirements

Administration Requirements

What elements of the application need to be configurable?

- User session key
- Login page name
- Service Facades
- The period prior to the contest start date used when starting the spec review later
- The content string used to resubmit the spec review

Technical Constraints

Are there particular frameworks or standards that are required?

- Struts 2.1.1
- Spring 3.0
- EJB 3.0

TopCoder Software Component Dependencies:

- Struts Framework 1.0
- Contest Service Façade 1.0
- Specification Review Service 1.0 (Provided by the OR updates dependency architecture)
- Security Manager 1.0
- Spec Review Comment Service 1.0 (Provided by the OR updates dependency architecture assembly)

**Please review the TopCoder Software component catalog for existing components that can be used in the design.

Third Party Component, Library, or Product Dependencies:

None

QA Environment:

- Java 1.5/J2EE 1.5
- JBoss 4.0.2

Copyright TopCoder, Inc. 2008

- Informix 11
- MySQL 5.1
- Struts 2.1.8.1
- Spring 3.0
- Javascript 1.8
- Mozilla Firefox 2.0/3.0
- IE 6.0/7.0
- Google Chrome
- Safari 3/4

Design Constraints

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

Required Documentation

Design Documentation

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

Help / User Documentation

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

Copyright TopCoder, Inc. 2008