Bread Crumb Trail Tag 2.0 Requirements Specification

1. Scope

1.1 Overview

Today's websites have complex navigation rules. As a user navigates through a website, it is very easy to become disoriented. A bread crumb trail provides links following the path of the user. This component provides an easily customizable tag to provide breadcrumb functionality to a website. The look and feel of the tag is set using CSS style sheets.

While simple web pages can assume the ownership of the whole screen, a *portlet* application running under a *portal* environment can only render to an assigned section of the screen. The new version of this component will be enhanced to support such clustered environments.

1.2 Logic Requirements

1.2.1 Existing Requirements

This version of Bread Crumb Trail Tag will assume all previous requirements from version 1.0. The new design will enhance the existing one to meet the additional requirements.

1.2.2 Multiple Parent Nodes

The version 1.0 design assumes that each node has only one parent, and thus each page can only go back to one page. However, it is possible that multiple pages can point to the same page. The new design will provide enhanced support for handling multiple parent nodes and automatically detect the correct parent in the generated trail.

1.2.3 Support for Advanced Servlet Navigation

In version 1.0, the URL attribute of must be written explicitly as "/path/xyz.jsp". For some servlets, the target URL may be of the form "/path/xyz.do", which performs some processing before redirecting to the target JSP. Moreover, it is likely that some parameters need to be passed. The new design will support URLs specified in the advanced servlet navigation style.

e.g. url = "/path/xyz.do" or url = "/path/xyz.do&param1=value1&param2=value2"

1.2.4 Support for Portals and Portlets

In a portal environment (e.g. IBM WebSphere, BEA WebLogic), each portlet owns a part of the screen assigned to its portlet container. The tag rendering process must take this into account and not assume it has control of the full screen. Otherwise, the rest of the portal skeleton will not render correctly. Portlets follow the JSR 168 specification.

In addition, each portal environment may use its own custom tags to display hyperlinks and GUI elements. For example, WebLogic provides the *netui* tag library to render anchors and forms. The design will support the usage of special tags for rendering through configuration. This version will provide support for WebLogic 8.1.

Reference: JSR 168 – Portlet Specification (http://www.jcp.org/en/jsr/detail?id=168)

1.2.5 Dynamic Bread Crumb Trail Generation

Related to multiple parent nodes and portlets, the URL patterns of a web site may be very dynamic. Static XML configuration becomes near impossible to use in some situations. The design will provide a mechanism to dynamically generate the bread crumb trail based on the session state.

1.3 Required Algorithms

The design will provide a reference algorithm for dynamically generating the bread crumb trail.

1.4 Example of the Software Usage

The TopCoder website contains numerous useful pages. However, a user may lose their way while surfing TopCoder.com. In order to orient a user within the website and to provide a way for a user to backtrack a bread crumb trail would be used.

1.5 Future Component Direction

Store the history of URL navigation for various statistical analyses.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

The design will allow configuration of custom GUI tag libraries for the particular portal environment.

2.1.2 External Interfaces

The design will follow the interfaces defined by the JSR 168 Specification.

2.1.3 Environment Requirements

- Development language: Java 1.4
- Compile target: Java 1.4, Java 1.5

2.1.4 Package Structure

com.topcoder.web.ui.tag.breadcrumbtrail

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

XML binding needs to be configurable.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

- JSR 168 (http://www.jcp.org/en/jsr/detail?id=168)

3.2.2 TopCoder Software Component Dependencies:

None.

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

3.2.3 Third Party Component, Library, or Product Dependencies:

None.

3.2.4 QA Environment:

- Solaris 7
- RedHat Linux 7.1
- Windows 2000
- Windows Server 2003
- BEA WebLogic 8.1 (http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/weblogic)

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.