

Common User Interface

Functional Requirements

Approved in May 2005 by the California Digital Library spec team
M. Heath, S. Sugarman, R. Tennant, B. Tingle, S. Toub

The Common User Interface (CUI) will provide common methodologies and infrastructure to CDL web producers and UI programmers for constructing user interfaces to Common Framework components and services. It will enable CDL to construct user interfaces that make common features available to users of CDL services. It will also enable CDL to embed branding functionality for campus staff to use to customize user interfaces.

Principles

- *The UI must be easy to implement*, since a diverse variety of campus staff will be called upon to do so.
- *Simple tasks should be simple*. That is, it should not be difficult to change the appearance of a block of text (e.g., a simple CSS change).
- *Complex tasks can be moderately difficult* (e.g., reordering elements through XSLT).
- *The User Interface must allow for flexibility and extensibility*.
- *All code sent to the web browser will adhere to CDL standards for markup and accessibility, and will not require the browser to enable client-side scripting (e.g., JavaScript) or plug-ins (e.g., Flash) to perform required functions*.
- *Server response times shall fall within acceptable parameters*.
- *The CDL reserves the right to make changes to the output streams to its services (e.g., changing how elements are mapped)*. These changes should not put an undue burden on the UI maintainers.
- *The CDL has a need to monitor the performance impact of UI implementations*. Any design should ensure that UI configurations should not negatively impact performance of CDL services.

Basic Functionality

CUI should provide simple, consistent methods to CDL web producers and UI programmers on individual project teams for including the following basic functionality into user interfaces:

- Processing of user-entered information on forms, including validation of input
- Submitting requests to CDL services (including XTF)
- Retrieving results from CDL services
- Formatting results from CDL services
- Directing UI process flow and display of pages or page components based on conditions such as form entry, clicks on hyperlinks, errors, user's privileges, etc.
- Establishment of UI sessions and storage and retrieval of session data
- Interaction with CDL authentication and affiliation services

Branding Functionality

CUI should provide the ability for CDL UI teams to add a “branding layer” to user interfaces. This layer should allow:

- Establishment of suites of “skins”
- Ability to easily select a different “skin” (including both deep and superficial control, see below); i.e., through a parameter passed as an HTTP GET request and/or based on client IP or hostname requested (e.g., virtual server)
- Ability to alter the display components of the interface without substantial software coding knowledge or experience. For example, knowledge of Java programming should not be required, although XHTML, and CSS knowledge can be assumed.

Separation of Logic, Markup, and Presentation (a.k.a. Controller, Model and View?)

To the extent possible, CUI should separate the underlying business logic—that interacts with CDL services and controls process flow—from the markup of forms, menus, data, etc. The presentation layer should be separate from both the logic and the markup (e.g., controlled via an external CSS file). This should include:

- Division of processing into “logic,” “markup” and “presentation” layers
- Well-defined structure/protocol for passing “raw” data returned from CDL services
- Input on the logic aspects of the output stream:
 - Ability choose from a suite of possible screen flows and interaction design
 - Ability to request changes to existing screen flows and interaction design
 - Etc.
- Significant control over markup aspects of the output stream (e.g., editing a configuration file that provides XSLT type functionality):
 - Ability to make output elements disappear
 - Ability to re-order output elements
 - Ability to format dates and times
 - Etc.
- Complete control over the presentation aspects of the output stream (e.g., CSS type functionality):
 - Ability to change font styles, colors, etc.
 - Ability to control the position of logical blocks or page divisions
 - Etc.
- Ability to integrate other external data sources, such as RSS feeds (see Figure 1.), external files, user preferences and other user profile information, or calls to other programs (for example, a spell checker, “best bets”, authority file, etc.)
- Ability to customize error messages displayed to users

UI Creation and Maintenance

- Ability to easily create, delete, modify, and validate code
- Ability to see changes without restarting a server
- Ability to ping status of servers
- Ability to work on a server (different from production) that includes all functionality and all ingested data
- Ability to easily integrate with version control system (e.g., CVS)
- Ability to independently push any web presentation layer changes from staging to production
- Ability to store user preferences across sessions

Appendix I. Example Skinning and Portal Selection Scenarios:

User Request: <http://findit.cdlib.org/> without skin parameter or a campus IP

System Response: “portal finder” with/CDL skin

User Request: <http://findit.cdlib.org/> with skin parameter or a campus IP

System Response: “portal finder” with skin appropriate to the skin parameter (takes precedence for skinning purposes) or IP

User Request: <http://findit.cdlib.org/earthsciences/> without skin parameter or campus IP

System Response: Earth Sciences FindIt portal w/CDL skin

User Request: <http://findit.cdlib.org/earthsciences/> with skin parameter or campus IP

System Response: Earth Sciences FindIt portal with skin appropriate to the skin parameter (takes precedence for skinning purposes) or IP

User Request: <http://americanwest.cdlib.org/> with or without skin parameter or campus IP

System Response: American West portal with CDL skin