Cafe

**userChange**(UserProfile | undefined)

*Input*: userChange takes a new profile for the user using comment anywhere, such as resulting from a log-in. It can take no value, or undefined, to effectively log the user out.

*Output*: userChange is called on a log in, a log out, and at all other times when user data changes (such as updating their profile blurb). It calls loadProfile on the State object and passes that state to other objects that may need to change what is visible if the users access level has changed.

**pageChange**(string)

*Input*: URL string of new page.

*Output*: pageChange is called when a user navigates to a new page. It tells the fetcher to fetch comments.

**changeState**(string)

*Input*: A string corresponding to the new state key.

*Output*: changeState updates the viewing field in the state object and passes the state to the navbar so that it may render the new state.

**checkForResponses**()

*Output*: checkForResponses is called as a callback after every fetch. The server responses array is retrieved from the fetcher and passed to the dispatcher, along with a reference to cafe so the dispatcher can call the correct methods to realize the information retrieved from the server.

**stateChangeEventReceived**()

*Output*: stateChangeEventReceived listener is called when a user performs an actions that causes a state change. It parses the event and calls changeState to realize the change.

**clientEventReceived**()

*Output*: clientEventReceived is called when an event is picked up that reflects an action performed by the user that requires server communication. It calls fetcher.fetch with the data from that event.

**settingEventReceived**(Settings)

*Input*: A new Settings object.

*Output*: settingsEventReceived is called when a user changes their settings, such as when they change whether comments are sorted by ascending or descending. It updates the settings object with the new data and tells the appropriate objects to update their displayed data based on the new settings.

Fetcher

**fetch**(string, communication entity, Function)

*Input*: A string corresponding the correct API endpoint and a client-server communication entity associated with that endpoint.

*Output*: Fetch dispatches the HTTP Request to the endpoint, stores responses in the responses member of fetcher, and calls the callback function when complete.

**getAndClearResponses**()

*Output*: Clears the responses member of fetcher.

*Returns*: The data in the responses member of fetcher (an array of server responses).

State

**loadProfile**(UserProfile | undefined)

*Input*: A UserProfile to load or nothing, to clear the user.

*Output*: Changes the ownProfile member of State.

**setViewingTo**(string)

*Input*: A string representing a state key.

*Output*: Changes the viewing member of State to the parameter value.

CafeNavbar

**commentsButtonClicked**()

*Output*: commentsButtonClicked is called when commentsButton is clicked. It will cause the state to transition.

**settingsButtonClicked**()

*Output*: settingsButtonClicked is called when settingsButton is clicked. It will cause the state to transition.

**modButtonClicked**()

*Output*: modButtonClicked is called when modButton is clicked. It will cause the state to transition.

**adminButtonClicked**()

*Output*: adminButtonClicked is called when adminButton is clicked. It will cause the state to transition.

**loginButtonClicked**()

*Output*: loginButtonClicked is called when loginButton is clicked. It will cause the state to transition.

**registerButtonClicked**()

*Output*: registerButtonClicked is called when registerButton is clicked. It will cause the state to transition.

**logoutButtonClicked**()

*Output*: logoutButtonClicked is called when logoutButton is clicked. It will cause the state to transition.

**setFromState**(State)

*Input*: A State Object

*Output*: setFromState causes buttons to show and hide depending on whether the user is logged in, what their access level is, and what state they are in. It also changes the currentlyViewing member to the appropriate CafeWindow and hides all other windows.

**displayMessage**(Message)

*Input*: A Message object.

*Output*: displayMessage shows the data in the parameter Message in the globalMessage member of CafeNavBar; this member is used to display a variety of general messages the server wants the client to see.

Dispatcher

**dispatch**(serverResponse[], Cafe)

*Input*: An array of server responses and an instance of Cafe.

*Output*: Dispatch uses the "t" field of the ServerResponse to determine what kind of information is contained in the "d" field. The "t" field determines which Dispatcher method is next called, and the object which will render the data is retrieved using the parameter reference to Cafe.

**dispatchMessage**(Message, CafeNavbar)

*Input*: A Message and a CafeNavBar.

*Output*: dispatchMessage calls displayMessage on the CafeNavbar object.

**dispatchUserUpdate**(UserProfile, Cafe)

*Input*: UserProfile data and the Café root object.

*Output*: dispatchUserUpdate calls userChange on the Cafe root object to change state reflecting any changes that may have happened to the User and to change what is visible on their profile.

**dispatchFeedbackUpdate**(FeedbackRecord, FeedbackReportSection)

*Input*: A FeedbackRecord and a FeedbackReportSection.

*Output*: dispatchFeedbackUpdate calls updateFeedback on a FeedbackReportSection.

**dispatchCommentUpdate**(Comment, CafeCommentsWindow)

*Input*: A Comment with data and a CafeCommentsWindow that will be instantiating and displaying the CafeComment.

*Output*: dispatchCommentUpdate calls updateComment on a CafeCommentsWindow

**dispatchComments**(Comment[], CafeCommentsWindow)

*Input*: A list of comments, and the window to render the comments to.

Output: dispatchComments calls populateNewComments on a CafeCommentsWindow

**dispatchUserProfile**(UserProfile, CafeUserDisplay)

*Input*: The user profile data, and the user display window to render it to.

*Output*: dispatchUserProfile calls changeProfile on a CafeUserDisplay to allow the user to view profile information for some other user.

**dispatchLoginResponse**(LoginResponse, Cafe)

*Input*:

*Output*: dispatchLoginResponse calls userChange on the Cafe root object to change state reflecting any changes that may have happened to the User and to change what is visible on their profile.

**dispatchLogoutResponse**(LogoutResponse, Cafe)

*Input*:

*Output*:

*Returns*:

**dispatchPwResetCodeResponse**(any, Cafe)

*Input*:

*Output*:

*Returns*:

**dispatchPwResetReqResponse**(any, Cafe)

*Input*:

*Output*:

*Returns*:

**dispatchReqVerificationResponse**(any, CafeMessageDisplay)

*Input*:

*Output*:

*Returns*:

**dispatchNewPassResponse**(any, CafeMessageDisplay)

*Input*:

*Output*:

*Returns*:

**dispatchVerifyResponse**(any, CafeMessageDisplay)

*Input*:

*Output*:

*Returns*:

**dispatchBanRecords**(BanRecord[], BanRecordsSection)

*Input*:

*Output*:

*Returns*:

**dispatchCommentReports**(CommentReport[], CommentReportsSection)

*Input*:

*Output*:

*Returns*:

**dispatchCommentReportUpdate**(CommentReport, CommentReportsSection)

*Input*:

*Output*:

*Returns*:

**dispatchFeedbacks**(FeedbackRecord[], FeedbackReportSection)

*Input*:

*Output*:

*Returns*:

**dispatchLogs**(AdminAccessLog[], LogsSection)

*Input*:

*Output*:

*Returns*:

**dispatchModRecords**(ModerationRecord[], ModActionsReportSection)

*Input*:

*Output*:

*Returns*:

**dispatchGlobalMods**(GlobalModeratorRecord[], ModeratorsReportSection)

*Input*:

*Output*:

*Returns*:

**dispatchDomainMods**(DomainModeratorRecord[], ModeratorsReportSection)

*Input*:

*Output*:

*Returns*:

**dispatchDomainReport**(AdminDomainReport, CafeAdminWindow)

*Input*:

*Output*:

*Returns*:

**dispatchUsersReport**(AdminUsersReport, CafeAdminWindow)

*Input*:

*Output*:

*Returns*:

CafeWindow

**show**()

*Output*: Shows a CafeWindow instance

**hide**()

*Output*: Hides a CafeWindow instance

CafeCommentsWindow

**populateNewComments**(Comment[])

*Input*: The list of new comments.

*Output*: Removes and replaces all instances of CafeComment with contents of the comment array.

**updateComment**(Comment)

*Input*: The comment to be updated.

*Output*: Updated data for the target CafeComment. A new CafeComment is created if the target doesn’t exist.

**resortComments**()

*Output*: Removes and resorts all CafeComments on the page according to the user’s settings.

**updateFromSettings**(CafeSettings)

*Input*: CafeSettings instance specifying comment viewing preferences.

*Output*: Saves previous settings and calls resortComments.

CafeSettingsWindow

**updateFromSettings**(CafeSettings)

*Input*: CafeSettings instance containing modified user settings.

*Output*: Calls updateFromSettings for the comment sort settings.

**verifyEmailClicked**()

*Output*: Calls show() to show the validation code input window.

**verifyCodeSubmitClicked**()

*Output*: Dispatch a Verify object to the server to verify the input code.

**passwordResetClicked**()

*Input*:

*Output*:

*Returns*:

CafeAdminWindow

**showDomainReport**(AdminDomainReport)

*Input*:

*Output*:

*Returns*:

**usersReportRequestButtonClicked**()

*Input*:

*Output*:

*Returns*:

**showUsersReport**(AdminUsersReport)

*Input*:

*Output*:

*Returns*:

CafeLoginWindow

**submitLoginButtonClicked**()

*Input*:

*Output*:

*Returns*:

**forgotPasswordButtonClicked**()

*Input*:

*Output*:

*Returns*:

CafeRegisterWindow

**submitRegisterClicked**()

*Input*:

*Output*:

*Returns*:

CafePwResetRequestWindow

**submitButtonClicked**()

*Input*:

*Output*:

*Returns*:

CafePWResetCodeWindow

**submitButtonClicked**()

*Input*:

*Output*:

*Returns*:

CafeNewPasswordWindow

**submitButtonClicked**()

*Input*:

*Output*:

*Returns*:

FeedbackReportSection

**updateFeedback**(FeedbackRecord)

*Input*:

*Output*:

*Returns*:

**requestFeedbackClicked**()

*Input*:

*Output*:

*Returns*:

**populateFeedback**(FeedbackRecord[])

*Input*:

*Output*:

*Returns*:

**clearFeedback**()

*Input*:

*Output*:

*Returns*:

ModeratorsReportSection

**requestDomainModsClicked**()

*Input*:

*Output*:

*Returns*:

**requestGlobalModsClicked**()

*Input*:

*Output*:

*Returns*:

**populateDomainModerators**(DomainModeratorRecord[])

*Input*:

*Output*:

*Returns*:

**populateGlobalModerators**(GlobalModeratorRecord[])

*Input*:

*Output*:

*Returns*:

**clearModsClicked**()

*Input*:

*Output*:

*Returns*:

ModActionsReportSection

**requestModRecordsButtonClicked**()

*Input*:

*Output*:

*Returns*:

**populateModActions**(ModerationRecord[])

*Input*:

*Output*:

*Returns*:

**updateModAction**(ModerationRecord)

*Input*:

*Output*:

*Returns*:

**clearModActionsClicked**()

*Input*:

*Output*:

*Returns*:

BanRecordsSection

**populateBanRecords**(BanRecord[])

*Input*:

*Output*:

*Returns*:

**requestGlobalBansClicked**()

*Input*:

*Output*:

*Returns*:

**requestDomainBansClicked**()

*Input*:

*Output*:

*Returns*:

**clearBanRecords**()

*Input*:

*Output*:

*Returns*:

CommentReportsSection

**populateCommentReports**(CommentReport[])

*Input*:

*Output*:

*Returns*:

**viewReportsClicked**()

*Input*:

*Output*:

*Returns*:

**updateCommentReport**(CommentReport)

*Input*:

*Output*:

*Returns*:

**clearCommentReports**()

*Input*:

*Output*:

*Returns*:

LogsSection

**submitLogsRequestClicked**()

*Input*:

*Output*:

*Returns*:

**populateLogs**(AdminAccessLog[])

*Input*:

*Output*:

*Returns*:

**clearLogs**()

*Input*:

*Output*:

*Returns*:

UIInput<Type>

**clickListen**(HTMLElement, Function[], Boolean)

*Input*:

*Output*:

*Returns*:

**disable**()

*Input*:

*Output*:

*Returns*:

**enable**()

*Input*:

*Output*:

*Returns*:

**destroy**()

*Input*:

*Output*:

*Returns*:

CafeComment

**replyClicked**()

*Input*:

*Output*:

*Returns*:

**submitReplyClicked**()

*Input*:

*Output*:

*Returns*:

**reportClicked**()

*Input*:

*Output*:

*Returns*:

**submitReportClicked**()

*Input*:

*Output*:

*Returns*:

CafeCommentVote

**upVoteClicked**()

*Input*:

*Output*:

*Returns*:

**downVoteClicked**()

*Input*:

*Output*:

*Returns*:

CafeMessageDisplay

**updateMessage**(Message)

*Input*:

*Output*:

*Returns*:

**clearMessage**()

*Input*:

*Output*:

*Returns*:

CafeUserDisplay

**changeProfile**(UserProfile)

*Input*:

*Output*:

*Returns*:

**hide**()

*Input*:

*Output*:

*Returns*:

**show**()

Input:

Output:

Returns:

CafeFeedbackDisplay

**hideClicked**()

*Input*:

*Output*:

*Returns*:

CafeOwnProfileDisplay

**updateProfile**(UserProfile)

*Input*:

*Output*:

*Returns*:

**editProfileBlurbClicked**()

*Input*:

*Output*:

*Returns*:

**editProfileSubmitClicked**()

*Input*:

*Output*:

*Returns*:

CafeBanRecordDisplay

**bannedUserClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

**bannedByClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

CafeCommentReportDisplay

**reportingUserClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

**moderateButtonClicked**()

*Input*:

*Output*:

*Returns*:

**submitModerationButtonClicked**()

*Input*:

*Output*:

*Returns*:

CafeLogDisplay

**usernameClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

CafeModRecordDisplay

**moderatorUsernameClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

CafeGlobalModDisplay

**grantedByUsernameClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

**grantedToUsernameClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

CafeDomainModDisplay

**grantedByUsernameClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

**grantedToUsernameClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

CafeDomainReportDisplay

**xButtonClicked**()

*Input*:

*Output*:

*Returns*:

CafeUsersReportDisplay

**newestUserClicked**(MouseEvent)

*Input*:

*Output*:

*Returns*:

**update**(AdminUsersReport)

*Input*:

*Output*:

*Returns*:

CafeCommentSortDisplay

**settingChange**()

*Input*:

*Output*:

*Returns*:

**updateFromSettings**(CafeSettings)

*Input*:

*Output*:

*Returns*: