**Package communication**

Package communication describes the client-server communication entities and the server-client communication entities for use with the back end. These were previously described in the Specifications document. The same classes are realized with typescript type definitions on the front end so that the front end and back-end share communication paths.

*Client-Server Communication Entities*

Class Ban

Description: Ban is dispatched when a moderator or administrator bans a user.

Data Members: int64 UserId, string Reason, string Domain

Class ChangeEmail

Description: ChangeEmail is dispatched to the server when a client wants to change their email. They must supply the correct password as well.

Data Members: string NewEmail, string Password

Class ChangeFeedback

Description: ChangeFeedback is dispatched to the Server when an admin wants to remove or hide a Feedback record from being shown to them again.

Data Members: bool Delete, int64 FeedbackId, bool SetHiddenTo

Class ChangeProfileBlurb

Description: ChangeProfileBlurb is dispatched to the server when a client updates their profile blurb.

Data Members: string NewBlurb

Class CommentReply

Description: CommentReply is dispatched to the server when a logged-in user submits a reply to an existing comment or posts a new root-level comment on a page.

Data Members: int64 ReplyingTo, string Reply

Class CommentVote

Description: CommentVote is dispatched to the server when a logged-in user votes on a comment.

Data Members: int64 VotingOn, string VoteType, int8 Value

Class Feedback

Description: Feedback is dispatched to the Server when a user submits feedback on Comment

Data Members: string FeedbackType, string Content

Class GetComments

Description: GetComments is dispatched to the server when a user opens the Browser Extension or when they navigate to a new page with the browser extension over. It is a request for all comments associated with the given url.

Data Members: string Url, string SortedBy, bool SortAscending

Class GetUserProfile

Description: GetUserProfile is dispatched to the server when the user needs to see a user’s profile.

Data Members: int64 UserId

Class Login

Description: Login is dispatched to the server when the client clicks “Submit” on the login form.

Data Members: string Username, string Password

Class Logout

Description: Logout is dispatched to the server when the client clicks “Logout”. It does not carry any additional data.

Data Members:

Class Moderate

Description: Moderate is dispatched to the server when a moderator or admin takes a moderation action on a comment.

Data Members: int64 CommentId, int64 AssociatedReport, bool SetHiddenTo, bool SetRemovedTo, string Reason

Class PasswordResetCode

Description: PasswordResetCode is dispatched by a user when they enter a password reset code. After a user clicks “Forgot My Password”, users may enter the code emailed to them. When they subsequently click the “submit” button, this request is dispatched to the server.

Data Members: int32 Code

Class PasswordReset

Description: PasswordReset is dispatched to the server when a password reset is requested. The client supplies the email associated with their account.

Data Members: string Email

Class PostCommentReport

Description: PostCommentReport is dispatched to the server when the user reports a comment.

Data Members: int64 CommentId, string Reason

Class Register

Description: Register is dispatched to the server when the client clicks “Submit” on the register form.

Data Members: string Username, string Password, string RetypePassword, string Email, bool AgreedToTerms

Class RequestVerification

Description: RequestVerification is dispatched to the server when the client wants a new validation code. If a client does not validate their account in a timely fashion, the validation code expires. The client may request a new validation code through their settings tab. When they do so, this entity is created and dispatched to the server.

Class SetNewPass

Description: SetNewPass is dispatched to the Server when the user changes their password. After submitting a valid password reset code, users are prompted to set a new password. When they subsequently click “submit”, this request is dispatched to the server.

Data Members: string Password, string RetypePassword

Class Verify

Description: Verify is dispatched to the server when the client inputs a validation code they received in an email to verify their account.

Data Members: Code int32

Class ViewBans

Description: ViewBans is dispatched to the server when an admin requests records of banned users.

Data Members: []string ForDomains

Class ViewCommentReports

Description: ViewCommentReports is dispatched to the server when a moderator requests comment reports. It does not have any data. The server will always respond to this with all reports which have not already been moderated. If the client is a DomainModerator, the server will filter appropriately and does not require additional information from the client.

Data Members:

Class ViewDomainReport

Description: ViewDomainReport is dispatched to the server when an admin requests a report on a domain.

Data Members: string domain

Class ViewUsersReport

Description: ViewUsersReport is dispatched to the server when an admin requests a report on the overall users of comment anywhere.

Data Members:

Class ViewFeedback

Description: ViewFeedback is dispatched to the Server when an admin wishes to view feedback submitted by users of Comment Anywhere.

Data Members: int64 From, int64 To, string FeedbackType

Class ViewLogs

Description: ViewLogs is dispatched to the server when an admin requests access logs.

Data Members: int64 ForUserId, string ForIp, string ForDomain, int64 StartingAt, int64 EndingAt

Class ViewModRecords

Description: ViewModRecords is dispatched to the server when an admin requests moderation records. It does not have any data. The server will always respond to this with all moderation records, sorted from newest to oldest.

Class ViewMods

Description: ViewMods is dispatched to the server when an admin requests records of who has been assigned as moderators.

Data Members: []string ForDomains

*Server-Client Communication Entities*

Class AdminAccessLog

Description: AdminAccessLog contains data needed by Admins to see an access log.

Data Members: string Ip, int64 LogId, string Url, string UserId, string Username

Class AdminDomainReport

Description: AdminDomainReport contains data needed by Admins to see information about activity on a particular domain.

Data Members: int32 CommentCount, string Domain

Class AdminUsersReport

Description: AdminUsersReport is dispatched when an Admin requests the Users report.

Data Members: int32 LoggedInUserCount, int64 NewestUserId, string NewestUsername, int64 UserCount

Class BanRecord

Description: BanRecord contains data about a banning or unbanning which occurred, which is used by Admins to see information about Moderator actions in certain reports.

Data Members: string BannedFrom, int64 BanRecordId, int64 BannedUserId, string BannedUsername, int64 BannedByUserID, string BannedByUsername, bool SetBannedTo, int64 BannedAt, string Reason

Class Comment

Description: Comment provides the data the Front End needs to render a comment.

Data Members: string UserId, int64 CommentId, string Content, CommentVoteDimension Factual, CommentVoteDimension Funny, CommentVoteDimension Agree, bool Hidden, int64 Parent, bool Removed, int64 TimePosted, string Username

Class CommentReport

Description: CommentReport contains data the Front End needs to render a CommentReport, which are reports submitted by users and which Moderators can review and take action on.Data Members: bool ActionTaken, Comment CommentData, string ReasonReported, int64 ReportId, int64 ReportingUserId, string ReportingUsername, int64 TimeReported

Class CommentVoteRecord

Description: CommentVoteRecord contains data for the number of votes on a comment.

Data Members: bool AlreadyVoted, int64 Downs, int64 Ups

Class DomainModeratorRecord

Description: DomainModeratorRecord contains data needed by Admins to see information about DomainModerator assignments.

Data Members: int64 GrantedAt, int64 GrantedBy, string GrantedByUsername, int64 GrantedTo, string GrantedToUsername, int64 RecordId

Class FeedbackRecord

Description: FeedbackRecord contains data the Front End needs to render a FeedbackRecord, which is a record of a user-submitted feedback, viewed by an Admin, such as a feature request, or bug report.

Data Members: string Content, bool Hide, int64 Id, int64 SubmittedAt, string FeedbackType, int64 Userid, string Username

Class GlobalModerator

Description: GlobalModerator record contains data needed by Admins to see information about GlobalModerator assignments.

Data Members: int64 GrantedAt, int64 GrantedBy, string GrantedByUsername, int64 GrantedTo, string GrantedToUsername, int64 RecordId

Class LoginResponse

Description: LoginResponse is sent to the client when they successfully log in.

Data Members: UserProfile LoggedInAs

Class Message

Description: Message is a general communication entity used to provide feedback to a client that some action has completed (or not completed) on requests where the client has not asked for any particular data

Data Members: bool Success, string Text

Class ModerationRecord

Description: ModerationRecord contains data the Front End needs to render a ModerationRecord, which is a record of a moderator action, such as hiding or removing a comment.

Data Members: CommentReport AssociatedReport, int64 ModerationRecordId, int64 ModeratorUserId, string ModeratorUsername, string Reason, bool SetHiddenTo, bool SetRemovedTo, int64 TimeModerated

Class UserProfile

Description: UserProfile contains data needed by the Front End to display a profile for a user.

Data Members: int64 CreatedOn, []string DomainsModerating, bool IsAdmin, bool IsDomainModerator, bool IsGlobalModerator, string ProfileBlurb, int64 UserId, string Username