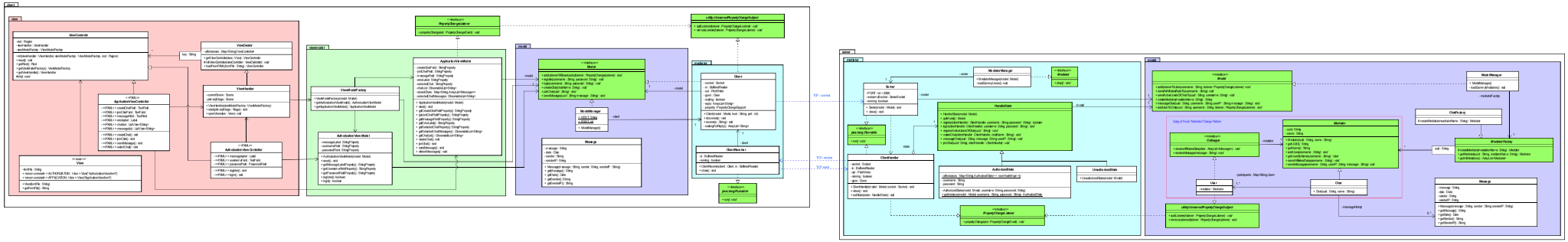
client
View
ViewController
- out : Frame
- eventsHandler : ViewHandler
- modelsFactory : ViewModelFactory
+ initViewHandler : ViewHandler, modelsFactory : ViewModelFactory, out : Frame)
+ setOut() : void
+ getHeap() : Heap
+ getViewModelFactory() : ViewModelFactory
+ getViewHandler() : ViewHandler
#cleg() : void
key : String
ViewCreator
- allvisitors : Map<String,ViewController>
+ getOrCreateVisitor(key : String) : ViewController
#+FabricCreateNewViewController : ViewController) : void
+ loadRootFXMLLikeFile : String) : ViewController
ViewHandler
- currentScene : Scene
- primaryStage : Scene
+ ViewHandler(modelsFactory : ViewModelFactory)
+ initStyle(vsStage : Stage) : void
+ openScene(sc : Scene) : void
ViewModelsFactory
+ createViewModel(modelClass : Class) : ViewModel
+ getApplicationViewModel() : ApplicationViewModel
+ getApplicationAllViewModels() : ApplicationViewModel
<< FXML >>
ApplicationViewController
-+FXML+ + createChatPad() : TextPad
-+FXML+ + joinChat() : ArrayList
-+FXML+ + messageWin() : TextPad
-+FXML+ + onnoSpot() : Label
-+FXML+ + messageLabel() : ArrayList<String>
-+FXML+ + messageList() : ArrayList<String>
-+FXML+ + roomChat() : void
-+FXML+ + joinChat() : void
-+FXML+ + sendMessage() : void
-+FXML+ + exitChat() : void
<< enum >>
View
- fxmlFile : String
+ name() constant = AUTHORIZATION : View = View(FxMLApplicationAuthorizeFF)
+ name() constant = APPRECIATION : View = View(FxMLApplicationAuthorizeFF)
+ fxmlFileFile() : String
+ getFxmlFile() : String
<< FXML >>
AuthorizationViewController
-+FXML+ + messageLabel : Label
-+FXML+ + usernameField : TextPad
-+FXML+ + passwordField : PasswordPad
-+FXML+ + signIn() : void
-+FXML+ + logIn() : void
<<interface>>
PropertyChangeListener
+ propertyChanged : PropertyChangeEvent) : void
utility.UniversalPropertyChangeSupport
+ addListener(listener : PropertyChangeListener) : void
+ removeListener(listener : PropertyChangeListener) : void
viewmodels
ApplicationAllViewModel
-+FXML+ +createChatPad() : TextPad
-+FXML+ +joinChat() : void
-+FXML+ +messageWin() : TextPad
-+FXML+ +onnoSpot() : Label
-+FXML+ +messageLabel() : String
-+FXML+ +messageList() : ArrayList<String>
-+FXML+ +roomChat() : void
-+FXML+ +joinChat() : void
-+FXML+ +sendMessage() : void
-+FXML+ +exitChat() : void
ViewModelChatPadAdmin
+ newChatPadAdmin(model : Model)
+ getApplicationAllViewModel() : ApplicationAllViewModel
AuthorizationViewModel
- messageLabel : StringProperty
- usernameField : StringProperty
- passwordField : StringProperty
+ AuthorizationViewModel(model : Model)
+ reset() : void
+ getMessageLabelProperty() : StringProperty
+ getUsernameFieldProperty() : StringProperty
+ getPasswordFieldProperty() : StringProperty
+ signIn() : void
+ logIn() : void
+ logIn() : boolean
ApplicationAllViewModel
- connectChatPad : StringProperty
- joinChat : StringProperty
- messagePad : StringProperty
- onnoLabel : StringProperty
- removeChat : StringProperty
- chatList : ObservableList<String>
- messageList : ObservableList<Message>
- selectedChatMessages : ObservableList<String>
+ ApplicationAllViewModel(model : Model)
+ reset() : void
+ getConnectChatPadProperty() : StringProperty
+ getJoinChatProperty() : StringProperty
+ getMessagePadProperty() : StringProperty
+ getOnnoLabel() : StringProperty
+ getRemoveChatPropertyy() : StringProperty
+ getSelectedChatMessages() : ObservableList<String>
+ getChatList() : ObservableList<String>
+ exitChat() : void
+ joinChat() : void
+ sendMessage() : void
+ sendMessage() : void
model
<<interface>>
Model
+ addListener(listener : PropertyChangeListener) : void
+ registerUsername : String, password : String) : void
+ registerUsernamer : String, password : String) : void
+ requestSignIn(name : String) : void
+ requestChatJoin(name : String) : void
+ sendMessage(text : String, message : String) : void
Model Manager
- model : Model
+ Model Manager()
+ initModel(model : Model) : void
Message
- message : String
- date : Date
- sender : String
- sendeR : String
+ Message(message : String, sender : String, sendeR : String)
+ getMessage() : String
+ getDate() : Date
+ getSender() : String
+ getSenderR() : String
Client
- socket : Socket
- in : BufferedReader
- out : PrintWriter
- port : Date
- mailing : boolean
- messageMessaging : Message
- property : PropertyChangeSupport
+ Client(model : Model, host : String, port : int)
+ disconnect() : void
+ receive() : String : void
- readingPrivate() : ArrayList<String>
Client Worker
- in : BufferedReader
- writing : boolean
+ ClientWorker(socket : Client, in : BufferedReader)
+ run() : void
<<interface>>
java.lang.Runnable
+ run() : void
server
models.fxy
Ms data Mana ger
- model : Model
+ MsdataManager(model : Model)
- readServerLoop() : void
<<interface>>
Model
+ run() : void
Ms server
- PORT : int = 2999
- socket : ServerSocket
- running : boolean
+ Server(model : Model) : void
+ close() : void
- server
- model
<<interface>>
java.lang.Runnable
+ run() : void
- state
ClientHandler
- socket : Socket
- in : BufferedReader
- out : PrintWriter
- writing : boolean
- port : Date
+ ClientHandler(socket : Socket) : void
+ close() : void
+ setSelectable : Handler(boolean) : void
HandleState
+ HandleState(model : Model)
+ getUsers() : Model
+ registerUsernamer : ClientHandler, username : String, password : String) : boolean
+ signIn(clientHandler : ClientHandler, username : String, password : String) : void
+ requestUsernameOfChatSpace : String : void
+ requestChatJoin : ClientHandler, chatSpaceName : String) : void
+ messageTextSend : String, message : String, sendR : String) : void
+ joinChatSpace : String, clientHandler : ClientHandler) : void
- model
Auth-deAuthate
- allusersmap : Map<String,AuthorizedUser> = new HashMap<>()
- username : String
- password : String
+ AuthorizedUser(model : Model, username : String, password : String)
+ getUsernameFromUser : Model, username : String, password : String) : AuthorizedUser
Unauthe deAuthate
+ UnauthorizedUser(model : Model)
<<interface>>
PropertyChange sterer
+ propertyChanged : PropertyChangeEvent) : void
models
<<interface>>
Model
+ addListener(listener : PropertyChangeListener) : void
+ removeAllMsdataFromSpace(name : String) : void
+ removeUserJoinChatTextSend : String, username : String) : void
+ requestMsdataFromMsdataHere : String) : String
+ addMsdataToSpace(msdata : Msdata, name : String, sender : PropertyChangeSender) : void
+ addUserToSpace : String, username : String, sender : PropertyChangeSender) : void
Model Manager
- model : Model
+ Model Manager()
+ initServer(ms : Ms server) : void
Chat Fa ds ng
+ initMsdataFromMsdataHere : String) : Msdata
Msdata Fa ctory
+ initMsdataFromMsdataHere : String) : Msdata
+ getMsdataSpace(): String, msdataHere : String) : Msdata
+ getMsdata(): Map<Long,Msdata>
Copy of Flock "Mediator" Design Pattern
<<interface>>
Colleague
+ receiveWhenDisconnect : ArrayList<Message>) : void
+ receiveMessage(message : Long) : void
modes
Msdata
- uuid : String
- name : String
+ Msdata(uuid : String, name : String)
+ getUUID() : String
+ getName() : String
+ addUsernameto : String : void
+ addUserToSpace(name : String) : void
+ removeMsdataSpace(name : String) : void
+ sendMessageSpace(name : String, uuid* : String, message : String) : void
uuid : String
MsdataFactory
initMsdataFromMsdataHere : String) : Msdata
+ getMsdataSpace(): String, msdataHere : String) : Msdata
+ getMsdata(uuid) : Map,Long,Msdata>
- participants : Map<String,User>
User
- modes : Mediator
Chat
+ Chat(uuid : String, name : String)
- messageMsg
Message
- message : String
- date : Date
- sender : String
- sendeR : String
+ Message(message : String, sender : String sendeR* String)
+ getMessage() : String
+ getDate() : Date
+ getSender() : String
+ getSenderR() : String
utility.UniversalPropertyChangeSupport
+ addListener(listener : PropertyChangeListener) : void
+ removeListener(listener : PropertyChangeListener) : void
TCP - connect
TCP - receive
TCP - send