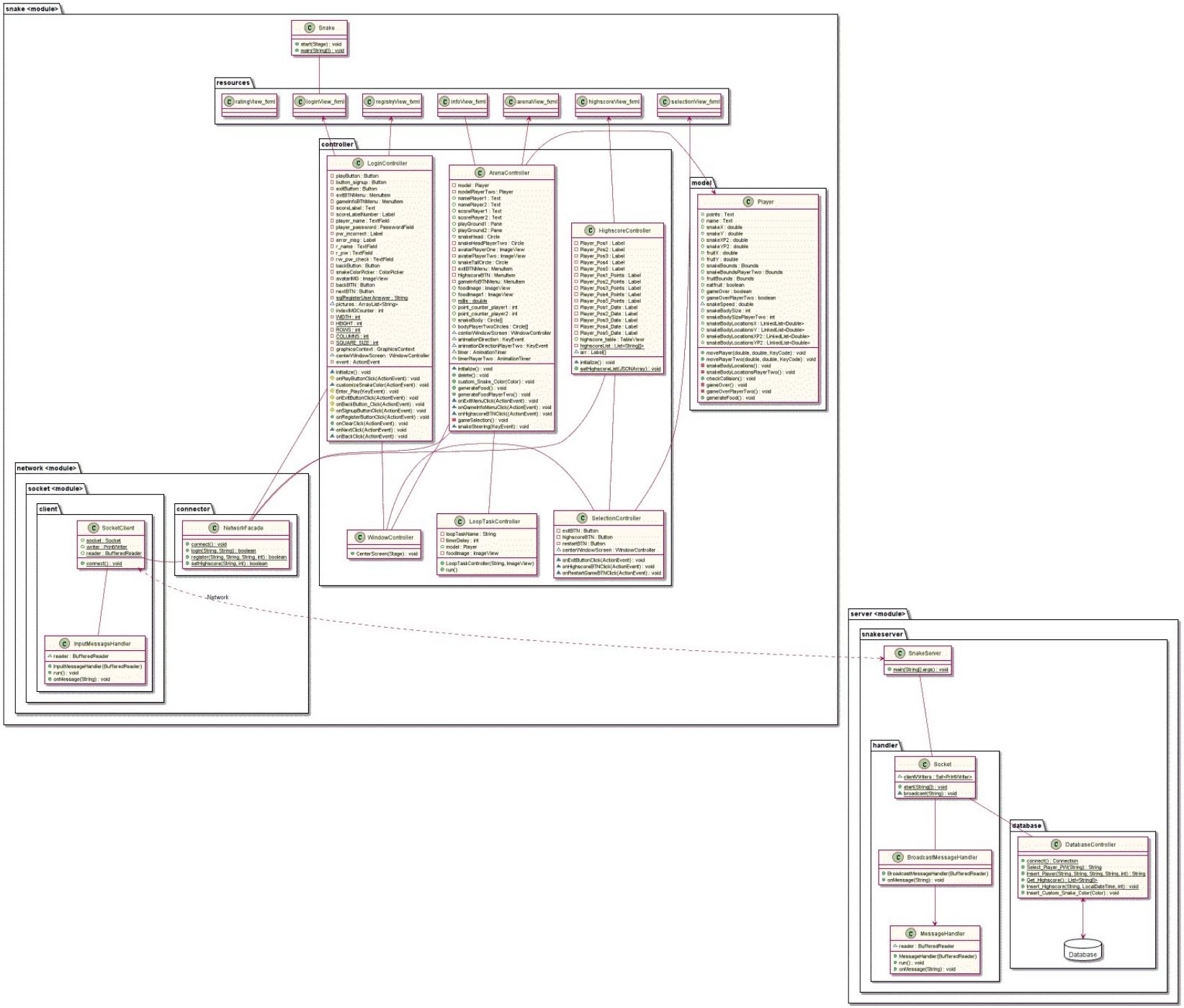
snake <module>
Snake
start(Stage) : void
main(String[]) : void
resources
ratingView_fxml
loginView_fxml
registryView_fxml
infoView_fxml
arenaView_fxml
highscoreView_fxml
selectionView_fxml
controller
LoginController
playButton : Button
button_signup : Button
exitButton : Button
exitBTNMenu : Menuitem
gameInfoBTNMenu : Menuitem
scoreLabel : Text
scoreLabelNumber : Label
player_name : TextField
player_password : PasswordField
pw_incorrect : Label
error_msg : Label
r_name : TextField
r_pw : TextField
rw_pw_check : TextField
backButton : Button
snakeColorPicker : ColorPicker
avatarIMG : ImageView
backBTN : Button
nextBTN : Button
sqlRegisterUserAnswer : String
pictures : ArrayList<String>
indexIMXCounter : int
WIDTH : int
HEIGHT : int
ROWS : int
COLUMNS : int
SQUARE_SIZE : int
graphicsContext : GraphicsContext
centerWindowScreen : WindowController
event : ActionEvent
initialize() : void
onPlayButtonClick(ActionEvent) : void
customizeSnakeColor(ActionEvent) : void
Enter_Play(KeyEvent) : void
onExitButtonClick(ActionEvent) : void
onBackButton_Click(ActionEvent) : void
onSignupButtonClick(ActionEvent) : void
onRegisterButtonClick(ActionEvent) : void
onClearClick(ActionEvent) : void
onNextClick(ActionEvent) : void
onBackClick(ActionEvent) : void
ArenaController
model : Player
modelPlayerTwo : Player
namePlayer1 : Text
namePlayer2 : Text
scorePlayer1 : Text
scorePlayer2 : Text
playGround1 : Pane
playGround2 : Pane
snakeHead : Circle
snakeHeadPlayerTwo : Circle
avatarPlayerOne : ImageView
avatarPlayerTwo : ImageView
snakeTailCircle : Circle
exitBTNMenu : Menuitem
HighscoreBTN : Menuitem
gameInfoBTNMenu : Menuitem
foodImage : ImageView
foodImage1 : ImageView
mills : double
point_counter_player1 : int
point_counter_player2 : int
snakeBody : Circle[]
bodyPlayerTwoCircles : Circle[]
centerWindowScreen : WindowController
animationDirection : KeyEvent
animationDirectionPlayerTwo : KeyEvent
timer : AnimationTimer
timerPlayerTwo : AnimationTimer
initialize() : void
delete() : void
custom_Snake_Color(Color) : void
generateFood() : void
generateFoodPlayerTwo() : void
onExitMenuClick(ActionEvent) : void
onGameInfoMenuClick(ActionEvent) : void
onHighscoreBTNClick(ActionEvent) : void
gameSelection() : void
snakeSteering(KeyEvent) : void
HighscoreController
Player_Pos1 : Label
Player_Pos2 : Label
Player_Pos3 : Label
Player_Pos4 : Label
Player_Pos5 : Label
Player_Pos1_Points : Label
Player_Pos2_Points : Label
Player_Pos3_Points : Label
Player_Pos4_Points : Label
Player_Pos5_Points : Label
Player_Pos1_Date : Label
Player_Pos2_Date : Label
Player_Pos3_Date : Label
Player_Pos4_Date : Label
Player_Pos5_Date : Label
highscore_table : TableView
highscoreList : List<String[]>
arr : Label[]
initialize() : void
setHighscoreList(JSONArray) : void
model
Player
points : Text
name : Text
snakeX : double
snakeY : double
snakeXP2 : double
snakeYP2 : double
fruitX : double
fruitY : double
snakeBounds : Bounds
snakeBoundsPlayerTwo : Bounds
fruitBounds : Bounds
eatfruit : boolean
gameOver : boolean
gameOverPlayerTwo : boolean
snakeSpeed : double
snakeBodySize : int
snakeBodySizePlayerTwo : int
snakeBodyLocationsX : LinkedList<Double>
snakeBodyLocationsY : LinkedList<Double>
snakeBodyLocationsXP2 : LinkedList<Double>
snakeBodyLocationsYP2 : LinkedList<Double>
movePlayer(double, double, KeyCode) : void
movePlayerTwo(double, double, KeyCode) : void
snakeBodyLocations() : void
snakeBodyLocationsPlayerTwo() : void
checkCollision() : void
gameOver() : void
gameOverPlayerTwo() : void
generateFood() : void
network <module>
socket <module>
client
SocketClient
socket : Socket
writer : PrintWriter
reader : BufferedReader
connect() : void
InputMessageHandler
reader : BufferedReader
InputMessageHandler(BufferedReader)
run() : void
onMessage(String) : void
connector
NetworkFacade
connect() : void
login(String, String) : boolean
register(String, String, String, int) : boolean
setHighscore(String, int) : boolean
WindowController
CenterScreen(Stage) : void
LoopTaskController
loopTaskName : String
timerDelay : int
model : Player
foodImage : ImageView
LoopTaskController(String, ImageView)
run()
SelectionController
exitBTN : Button
highscoreBTN : Button
restartBTN : Button
centerWindowScreen : WindowController
onExitButtonClick(ActionEvent) : void
onHighscoreBTNClick(ActionEvent) : void
onRestartGameBTNClick(ActionEvent) : void
-Network
server <module>
snakeserver
SnakeServer
main(String[] args) : void
handler
Socket
clientWriters : Set<PrintWriter>
start(String[]) : void
broadcast(String) : void
BroadcastMessageHandler
BroadcastMessageHandler(BufferedReader)
onMessage(String) : void
MessageHandler
reader : BufferedReader
MessageHandler(BufferedReader)
run() : void
onMessage(String) : void
database
DatabaseController
connect() : Connection
Select_Player_Pw(String) : String
Insert_Player(String, String, String, String, int) : String
Get_Highscore() : List<String[]>
Insert_Highscore(String, LocalDateTime, int) : void
Insert_Custom_Snake_Color(Color) : void
Database