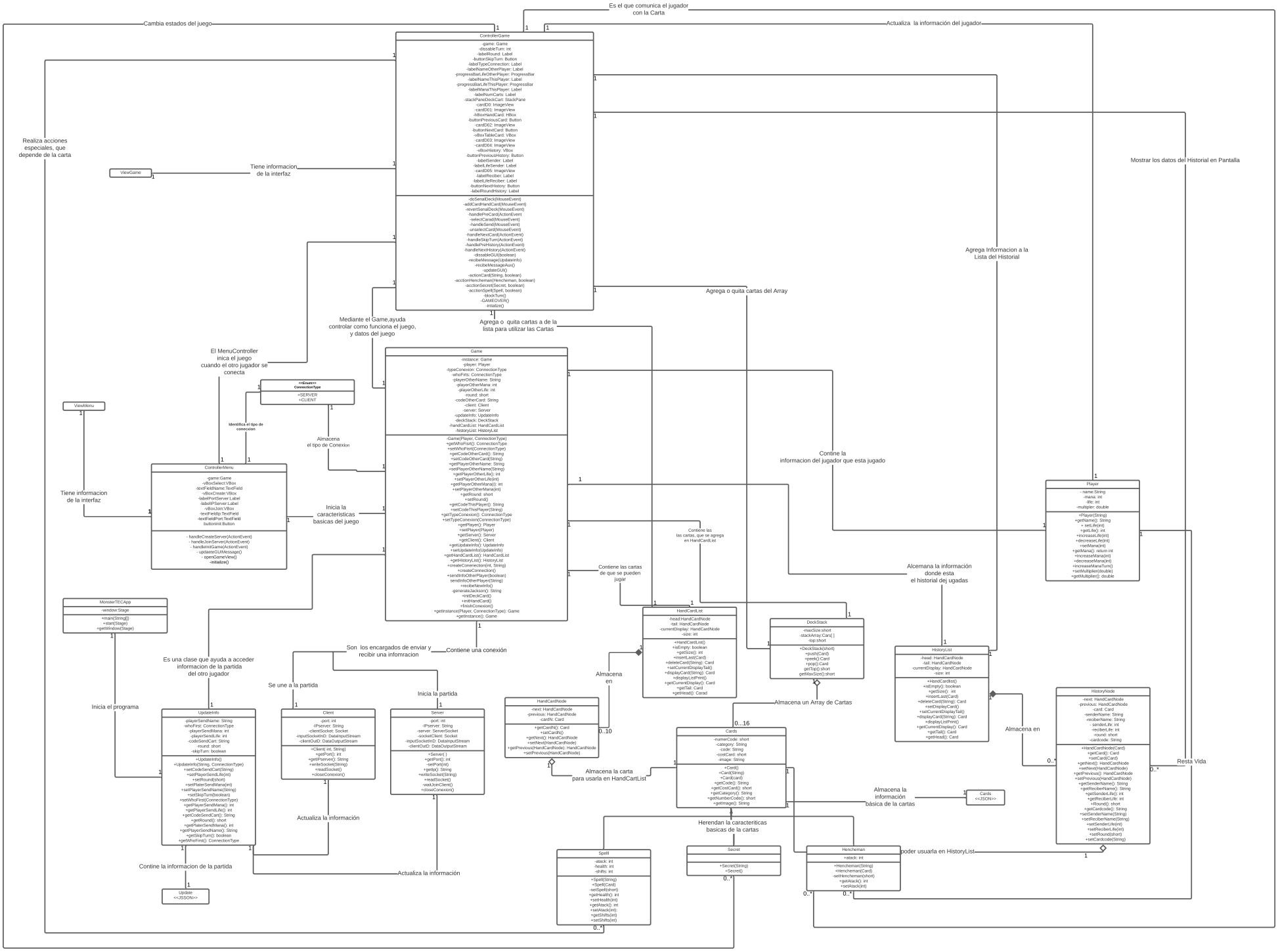
Es el que comunica el jugador con la Carta
Cambia estados del juego
Actualiza la información del jugador
ControlerGame
-game: Game
-dissableTurn: int
-labelRound: Label
-buttonSkipTurn: Button
-labelTypeConnection: Label
-labelNameOtherPlayer: Label
-progressBarLifeOtherPlayer: ProgressBar
-labelNameThisPlayer: Label
-progressBarLifeThisPlayer: ProgressBar
-labelManaThisPlayer: Label
-labelNumCarts: Label
-stackPaneDeckCart: StackPane
-cardD0: ImageView
-cardD01: ImageView
-hBoxHandCard: HBox
-buttonPreviousCard: Button
-cardD02: ImageView
-buttonNextCard: Button
-vBoxTableCard: VBox
-cardD03: ImageView
-cardD04: ImageView
-vBoxHistory: VBox
-buttonPreviousHistory: Button
-labelSender: Label
-labelLifeSender: Label
-cardD05: ImageView
-labelReciber: Label
-labelLifeReciber: Label
-buttonNextHistory: Button
-labelRoundHistory: Label
-doSenaDeck(MouseEvent)
-addCardHandCard(MouseEvent)
-revertSenaDeck(MouseEvent)
-handlePreCard(ActionEvent
-selecCarad(MouseEvent)
-handleSend(MouseEvent)
-unselectCard(MouseEvent)
-handleNextCard(ActionEvent)
-handleSkipTurn(ActionEvent)
-handlePreHistory(ActionEvent)
-handleNextHistory(ActionEvent)
-dissableGUI(boolean)
-recibeMessage(UpdateInfo)
-recibeMessageAun()
-updateGUIQ
-actionCard(String, boolean)
-acctionHencheman(Hencheman, boolean)
-acctionSecret(Secret, boolean)
-acctionSpell(Spell, boolean)
-blockTurn()
-GAMEOVER()
-initialize()
Realiza acciones especiales, que depende de la carta
ViewGame
Tiene informacion de la interfaz
Mostrar los datos del Historial en Pantalla
Agrega Informacion a la Lista del Historial
Agrega o quita cartas del Array
Mediante el Game,ayuda controlar como funciona el juego, y datos del juego
Agrega o quita cartas a de la lista para utilizar las Cartas
El MenuController inica el juego cuando el otro jugador se conecta
<<Enum>> ConnectionType
+SERVER
+CLIENT
Identifica el tipo de conexion
Almacena el tipo de Conexion
Game
-instance: Game
-player: Player
-typeConexion: ConnectionType
-whoFirts: ConnectionType
-playerOtherName: String
-playerOtherMana: int
-playerOtherLife: int
-round: short
-codeOtherCard: String
-client: Client
-server: Server
-updateInfo: UpdateInfo
-deckStack: DeckStack
-handCardList: HandCardList
-historyList: HistoryList
-Game(Player, ConnectionType)
+getWhoFirst(): ConnectionType
+setWhoFirst(ConnectionType)
+getCodeOtherCard(): String
+setCodeOtherCard(String)
+getPlayerOtherName: String
+setPlayerOtherName(String)
+getPlayerOtherLife(): int
+setPlayerOtherLife(int)
+getPlayerOtherMana(): int
+setPlayerOtherMana(int)
+getRound: short
+setRound()
+getCodeThisPlayer(): String
+setCodeThisPlayer(String)
+getTypeConexion(): ConnectionType
+setTypeConexion(ConnectionType)
+getPlayer(): Player
+setPlayer(Player)
+getServer(): Server
+getClient(): Client
+getUpdateInfo(): UpdateInfo
+setUpdateInfo(UpdateInfo)
+getHandCardList(): HandCardList
+getHistoryList(): HistoryList
+createConnection(int, String)
+createConnection()
+sendInfoOtherPlayer(boolean)
sendInfoOtherPlayer(String)
+recibeNewInfo()
-generateJacksonC: String
+initDeckCard()
+initHandCard()
+finishConexion()
+getInstance(Player, ConnectionType): Game
+getInstance(): Game
Contiene la informacion del jugador que esta jugado
Contiene las cartas de que se pueden jugar
Contiene las cartas, que se agrega en HandCardList
Alcemana la información donde esta el historial dej ugadas
Tiene informacion de la interfaz
ViewMenu
ControlerMenu
-game:Game
-vBoxSelect:VBox
-textFieldName:TextField
-vBoxCreate:VBox
-labelPortServer:Label
-labelIPServer:Label
-vBoxJoin:VBox
-textFieldip:TextField
-textFieldPort:TextField
buttonInit:Button
- handleCreateServer(ActionEvent)
- handleJoinServer(ActionEvent)
- handleInitGame(ActionEvent)
- updateGUIMessage()
- openGameView()
-initialize()
Inicia la caracteristicas basicas del juego
MonsterTECApp
-window:Stage
+main(String[])
+start(Stage)
+getWindow(Stage)
Son los encargados de enviar y recibir una infomracion
Contiene una conexión
Es una clase que ayuda a acceder informacion de la partida del otro jugador
Se une a la partida
Inicia la partida
Inicia el programa
UpdateInfo
-playerSendName: String
-whoFirst: ConnectionType
-playerSendMana: int
-playerSendLife: int
-codeSendCart: String
-round: short
-skipTurn: boolean
+UpdateInfo()
+UpdateInfo(String, ConnectionType)
+setCodeSendCart(String)
+setPlayerSendLife(int)
+setRound(short)
+setPlaterSendMana(int)
+setPlayerSendName(String)
+setSkipTurn(boolean)
+getWhoFirst(ConnectionType)
+getPlayerSendMana(): int
+getPlayerSendLife(): int
+getCodeSendCart(): String
+getRound(): short
+getPlaterSendMana(): int
+getPlayerSendName(): String
+getSkipTurn(): boolean
+getWhoFirst(): ConnectionType
Client
-port: int
-IPserver: String
-clientSocket: Socket
-inputSocketIn0: DataInputStream
-clientOutD: DataOutputStream
+Client(int, String)
+getPort(): int
+getIPserver(): String
+writeSocket(String)
+readSocket()
+closeConexion()
Server
-port: int
-IPserver: String
-server: ServerSocket
-socketClient: Socket
-inputSocketInD: DataInputStream
-clientOutD: DataOutputStream
+Server()
+getPort(): int
+setPort(int)
+getIp(): String
+writeSocket(String)
+readSocket()
-waitJoinClient()
+closeConexion()
Actualiza la información
Actualiza la información
Contine la informacion de la partida
Update
<<JSON>>
HandCardNode
-next: HandCardNode
-previous: HandCardNode
-cardN: Card
+getCardN(): Card
+setCardN()
+getNext(): HandCardNode
+setNext(HandCardNode)
+getPrevious(HandCardNode): HandCardNode
+setPrevious(HandCardNode)
0..10
Almacena en
HandCardList
-head:HandCardNode
-tail: HandCardNode
-currentDisplay: HandCardNode
-size: int
+HandCardList()
+isEmpty: boolean
+getSize(): int
+insertLast(Card)
+deleteCard(String): Card
+setCurrentDisplayTail()
+dsplayCard(String): Card
+dsplayListPrint()
+getCurrentDisplay(): Card
+getTail: Card
+getHead(): Carad
DeckStack
-maxSize:short
-stackArray:Cars[ ]
-top:short
+DeckStack(short)
+push(Card)
+peek():Card
+pop():Card
getTop():short
getMaxSize():short
Almacena un Array de Cartas
HistoryList
-head: HandCardNode
-tail: HandCardNode
-currentDisplay: HandCardNode
-size: int
+HandCardList()
+isEmpty(): boolean
+getSize(): int
+insertLast(Card)
+deleteCard(String): Card
+setDisplayCard()
+setCurrentDisplayTail()
+dsplayCard(String): Card
+dsplayListPrint()
+getCurrentDisplay(): Card
+getTail(): Card
+getHead(): Card
Almacena en
HistoryNode
-next: HandCardNode
-previous: HandCardNode
-card: Card
-senderName: String
-reciberName: String
-senderLife: int
-reciberLife: int
-round: short
-cardcode: String
+HandCardNode(Card)
+getCard(): Card
+setCard(Card)
+getNext(): HandCardNode
+setNext(HandCardNode)
+getPrevious(): HandCardNode
+setPrevious(HandCardNode)
+getSenderName(): String
+getReciberName(): String
+getSenderLife(): int
+getReciberLife: int
+Round(): short
+getCardcode(): String
+setSenderName(String)
+setReciberName(String)
+setSenderLife(int)
+setReciberLife(int)
+setRound(short)
+setCardcode(String)
Resta Vida
Player
- name:String
-mana: int
-life: int
-multiplier: double
+Player(String)
+getName(): String
+ setLife(int)
+getLife() : int
+increaseLife(int)
+decreaseLife(int)
+setMana(int)
+getMana(): return int
+increaseMana(int)
+decreaseMana(int)
+increaseManaTurn()
+setMultiplier(double)
+getMultiplier(): double
0...16
Cards
-numerCode: short
-category: String
-code: String
-postCard: short
-image: String
+Card()
+Card(String)
+Card(card)
+getCode(): String
+getCostCard(): short
+ getCategory(): String
+getNumberCode(): short
+getImage(): String
Almacena la carta para usarla en HandCardList
Almacena la información básica de la cartas
Cards
<<JSON>>
Herendan la caracteriticas basicas de la cartas
Spell
-atack: int
-health: int
-shifts: int
+Spell(String)
+Spell(Card)
-setSpell(short)
+getHealth(): int
+setHealth(int)
+getAtack(): int
+setAtack(int)
+getShifts(int)
+setShifts(int)
Secret
+Secret(String)
+Secret()
Hencheman
+atack: int
+Hencheman(String)
+Hencheman(Card)
-setHencheman(short)
+getAtack(): int
+setAtack(int)
poder usuarla en HistoryList
0..*