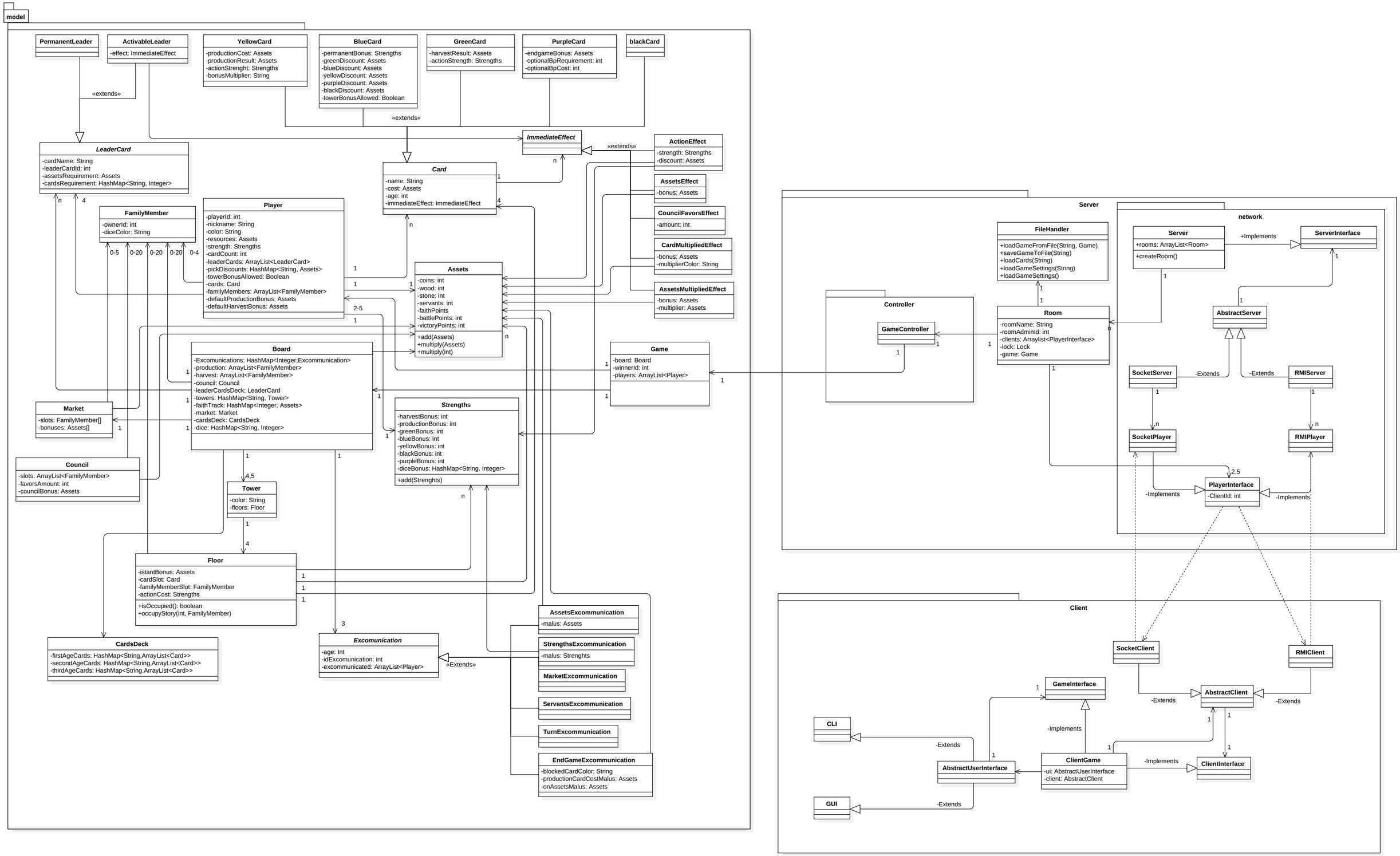
Model::Main


model
PermanentLeader
ActivableLeader
-effect: ImmediateEffect
YellowCard
-productionCost: Assets
-productionResult: Assets
-actionStrenght: Strengths
-bonusMultiplier: String
BlueCard
-permanentBonus: Strengths
-greenDiscount: Assets
-blueDiscount: Assets
-yellowDiscount: Assets
-purpleDiscount: Assets
-blackDiscount: Assets
-towerBonusAllowed: Boolean
GreenCard
-harvestResult: Assets
-actionStrength: Strengths
PurpleCard
-endgameBonus: Assets
-optionalBpRequirement: int
-optionalBpCost: int
blackCard
«extends»
LeaderCard
-cardName: String
-leaderCardId: int
-assetsRequirement: Assets
-cardsRequirement: HashMap<String, Integer>
ImmediateEffect
ActionEffect
-strength: Strengths
-discount: Assets
Card
-name: String
-cost: Assets
-age: int
-immediateEffect: ImmediateEffect
AssetsEffect
-bonus: Assets
CouncilFavorsEffect
-amount: int
CardMultipliedEffect
-bonus: Assets
-multiplierColor: String
AssetsMultipliedEffect
-bonus: Assets
-multiplier: Assets
FamilyMember
-ownerId: int
-diceColor: String
Player
-playerId: int
-nickname: String
-color: String
-resources: Assets
-strength: Strengths
-cardCount: int
-leaderCards: ArrayList<LeaderCard>
-pickDiscounts: HashMap<String, Assets>
-towerBonusAllowed: Boolean
-cards: Card
-familyMembers: ArrayList<FamilyMember>
-defaultProductionBonus: Assets
-defaultHarvestBonus: Assets
Assets
-coins: int
-wood: int
-stone: int
-servants: int
-faithPoints
-battlePoints: int
-victoryPoints: int
+add(Assets)
+multiply(Assets)
+multiply(int)
Board
-Excomunications: HashMap<Integer;Excommunication>
-production: ArrayList<FamilyMember>
-harvest: ArrayList<FamilyMember>
-council: Council
-leaderCardsDeck: LeaderCard
-towers: HashMap<String, Tower>
-faithTrack: HashMap<Integer, Assets>
-market: Market
-cardsDeck: CardsDeck
-dice: HashMap<String, Integer>
Game
-board: Board
-winnerId: int
-players: ArrayList<Player>
Market
-slots: FamilyMember[]
-bonuses: Assets[]
Strengths
-harvestBonus: int
-productionBonus: int
-greenBonus: int
-blueBonus: int
-yellowBonus: int
-blackBonus: int
-purpleBonus: int
-diceBonus: HashMap<String, Integer>
+add(Strenghts)
Council
-slots: ArrayList<FamilyMember>
-favorsAmount: int
-councilBonus: Assets
Tower
-color: String
-floors: Floor
Floor
-istantBonus: Assets
-cardSlot: Card
-familyMemberSlot: FamilyMember
-actionCost: Strengths
+isOccupied(): boolean
+occupyStory(int, FamilyMember)
CardsDeck
-firstAgeCards: HashMap<String,ArrayList<Card>>
-secondAgeCards: HashMap<String,ArrayList<Card>>
-thirdAgeCards: HashMap<String,ArrayList<Card>>
Excomunication
-age: Int
-idExcomunication: int
-excommunicated: ArrayList<Player>
«Extends»
AssetsExcommunication
-malus: Assets
StrengthsExcommunication
-malus: Strenghts
MarketExcommunication
ServantsExcommunication
TurnExcommunication
EndGameExcommunication
-blockedCardColor: String
-productionCardCostMalus: Assets
-onAssetsMalus: Assets
Server
FileHandler
+loadGameFromFile(String, Game)
+saveGameToFile(String)
+loadCards(String)
+loadGameSettings(String)
+loadGameSettings()
network
Server
+rooms: ArrayList<Room>
+createRoom()
+Implements
ServerInterface
Controller
GameController
Room
-roomName: String
-roomAdminId: int
-clients: Arraylist<PlayerInterface>
-lock: Lock
-game: Game
AbstractServer
SocketServer
-Extends
RMIServer
SocketPlayer
RMIPlayer
PlayerInterface
-ClientId: int
-Implements
Client
SocketClient
RMIClient
GameInterface
AbstractClient
CLI
AbstractUserInterface
ClientGame
-ui: AbstractUserInterface
-client: AbstractClient
ClientInterface
GUI