Main
+ main (args: String[]): void

LoginMenuView
+ input: String
+ matcher : Matcher

+ run(): void

Play Game Menu
+ input: String
+ matcher: Matcher
playerTurn: Player

+ run (Player): void
+printMap(player:Player): void
+ printGuide(): void

PlayGameController
+ field: type

+nextTurn(number: int):void

MainMenuView
+ input: String
+ matcher: Matcher

+ run (user: User): void

User
+ username: String
+ password: String
+ nickname: String
+ score: int
+ users: ArrayList<User>

+User (username:String, password: String, nickname: String): <<constructor>>
+setUsername (String): void
+setPassword (String): void
+getPassword(): String
+setNickname (): void
+setScore (int): void
+getScore(): int
+isThisUsernameAlreadyexist (username: String): boolean
+isThisNickNameAlreadyexist (nickname: String): boolean
+getUsernamebyUsername (String): User

ProfileMenuView
+ input: String
+ matcher: Matcher

+ run (user: User): void

UserController
+ login (username: String, password: String): User
+ logOut (user: User): String
+ signup (username: String, password: String, nickname: String): String
+ changeNickname (newNickname: String, user: User): String
+ changePassword (newPassword: String, user: User): String
+ playGame (users: arrayList<User>): String
+addUser(): void
+changePassword(String): void
+changeNickname(String): void
+changeScore(int): void

Producible
+turnsNeedToMake: int
+name: String
+technologyNeeded: Technology
+cost: int

+ method(type): type

Unit
+position_x : int
+position_y : int
+isSleep: boolean
+movementPoint: int
+owner: Player

+ move(destination_x: int , destination_y : int): void
+ isMovePossible (destination_x: int , destination_y : int): Boolean
+ getUnitByPosition(poition_x: int, position_y: int): unit: Unit
+sleep() void
+delete(): void
+wakeup():void
+cancelMission(): void
+getOwenr(): Player
+setOwner(Owner): void
+fortify(unit: Unit): void

Settler
+ field: type

+buildCity (hex: HEX): void
+effectOnFoodAndProuduction (foodProduction: int): v

Unmiliatry Unit
.

.

//TODO --> Technology
+ field: type

+ method(type): type

Building
+increaseFoodByName(int,String): void

Worker
isBusy: Boolean

MilitaryUnit
+strength: int
+XP: int
+ health: int
+attackRange: int
+temporaryStrength: int
owner:Player

+MilitaryUnit(x:int, y: int)<<constructor>>
+boost() :void
+boostToHeal():void
+attack(defender : Unit) :void
+loot (MilitaryUnit):void

Resources
+name :String
+food: int
+production: int
+gold: int

.

StrategicResources
.

.

Hex
+position_x: int
+position_y: int
+owner: player
+type: Land
+feature : Land
+price: int
+movementCost: int
+unmmilitaryUnitInside: Unmiliatry Unit
+militaryUnitInside : Military Unit
+northHex : Hex
+southHex : Hex
+westHex : Hex
+eastHex : Hex
+northEastHex : Hex
+southEastHex : Hex
+hasRiverOnNorth:Boolean
+hasRiverOnEast: Boolean
+hasRiverOnWest:Boolean
+hasRiverOnNorthEast: Boolean
+hasRiverOnSouthEast: Boolean

+Hex(int x, int y, type, feature, price):<<constructor>>

UnitController
+ field: type

+ Moveto (position_x: int, position_y: int, unit: Unit): void
+ isMovePossible (x: int , y: int): Boolean
+ sleep (unit:Unit): void
+ alert (unit: Unit): void
+ fortify (unit: Unit): void
+ fortifyByHeal (unit: Unit): void
+ garrison (unit: Unit,hex: Hex): void
+ setupByRange (unit: Unit): void
+ attack (unit: Unit): void
+ found (unit: Unit, hex: Hex): City
+ cancel (unit: Unit): void
+ wake (unit: Unit): void
+ delete (unit: Unit): void
+ build (unit: Unit, hex: Hex): void
+ remove (unit: Unit): void
+ repair (unit: Unit): void

Land
+name: String
+food: int
+production: int
+gold: int
+battleEffect: int
+movmentCost: int

+ method(type): type

Info
+ currentPlayer: Player

+printUnitList(player : Player):void
+printCitiesList(player :Player): void
+printDiplomacyInfo(player: Player): void
+printTradeHistory(player: Player): void
+printEconomicalInfo(player :Player):void
+printTotalMilitaryUnit(): void
+printNotificationInfo(): void
+printVictoryImprovement(): void
+printCivilization (): void
+researchInfo():void
+

LuxuryResources
.

.

Bonus Resources
.

.

Player
+nickname: String
+coinAmount: int
+coinProduction: int
+HappinessLevel: int
+ownedHex: Arraylist<Hex>
+ownedCity: Arraylist<City>
+ownedunits: Arraylist<Unit>
+mapState: HashMap<Hex,String>
+technoloyAchived: Arraylist<Technology>
+tradesHistory: Arraylist <String> //TODO
+

+ method

City
+isCapital: Boolean
+isFisrtCapital: Boolean
+foodProduction: int
+turns: int
+population: int
+inProduceImprovement: Producible
+ trunsPassFromObj: int
+ maxTurnNeeded: int
+ goldAmount: int
+ garrisonUnit: Militaryunit
+ownedBuildings: Arraylist<Building>

+addBuilding(building Building): void

PlayerController
+ showCoin (matcher: Matcher): void
+ showHappinesLevel (matcher: Matcher): void
+ showMap (matcher: Matcher): void
+ technologyReached (matcher : Matcher): voi
+ technologyWorkCity (matcher: Matcher)
+ showCities (matcher: Matcher): void
+ buildBuilding (matcher: Matcher): void
+ isTurns (matcher: Matcher): void
+ creatCityBySettler (matcher: Matcher): void
+ declarWar (matcher: Matcher): void
+ selectUnit (matcher: Matcher): void
+ selectCity (matcher: Matcher): void

Database
+users: Arraylist<User>
+map: Arraylist<Hex>
+

+ getUserByUsername (String):User
+getMapByPosition(int , int):Hex
+addHex(Hex):void
+addUser(User):void

Combat
+ player1: Player
+ player2: Player
+ unitAttacker: Unit
+ unitDefender: Unit
+ tileFeatures: Land

+ diplomaticallyDeclaringWar (player1: Player, player2: Player): void
+ attackingAnotherUnitDeclaringWar   (unitAttacker: Unit, unitDefender: Unit, player1: Player, player2: Player): void
+ meleeCombat (unitAttacker: Unit, unitDefender: Unit): void
+ rangedCombat (unitAttacker: Unit, unitDefender: Unit): void
+ combatBonuses (unitAttacker: Unit, unitDefender: Unit, tileFeatures: Land): void
+ combatSiegePreAttack (unitAttacker: Unit, unitDefender: Unit)

UnitMode(enum)
SLEEP,
STANDBYE
BOOST,
BOOST_TO_HEAL,
DEPLOYED,
LOOT,
BUILD_CITY,
CANCEL,
WAKE_UP,
DELETE,

Commands(enum)
regex: String
matcher: Matcher

ENTER_MENU,
LOGIN,
MENU_EXIT,
MENU_SHOW_CURRENT,
CREAT_USER,
MENU_ENTER,
CREATE_USER,
USER_LOGGOUT,
PLAY_GAME,
CHANGE_NICKNAME,
CHANGE_PASSWORD,
INCRESE_GOLD,
INCRESE_TURN,
INFO_PANEL,
EXPLORE_INFO_PANEL,
UNIT_LIST_PANEL,
CITY_LIST_PANEL,
DIPLOMACY_INFO_PANEL
VICTORY_PROGRESS_PANEL,
CIVILIZATION_PANEL,
NOTIFICATION_HISTORY,
GENERAL_MILITARY_PANEL,

Main Menu Commands(enum)
+ field: type

+ method(type): type

HEX State(enum)
FOG OF WAR
,CLEAR
,SPECIFIC

Extends

oid