Planet - technologyDefense: int - technologyAttack: int - metal: int - deuterium: int - upgradeDefenseTechnologyDeuteriumCost: int - upgradeAttackTechnologyDeuteriumCost: int - army: ArrayList<MilitaryUnit>[] - battleReports: String[] - isActiveThreat: boolean - currentThreat: Battle - nBattles: int - metalMineLvl: int - deuteriumMineLvl: int - upgradeMetalMineLvlMetalCost: int - upgradeDeuteriumMineLvlDeuteriumCost: int - difficulty: int + upgradeTechnologyDefense(): void + upgradeTechnologyAttack(): void + newLightHunter(int n): void + newHeavyHunter(int n): void + newBattleShip(int n): void + newArmoredShip(int n): void + newMissileLauncher(int n): void + newlonCannon(int n): void + newPlasmaCannon(int n): void + printStats(): void + getDifficulty(): int + setDifficulty(int i): void + getTechnologyDefense(): int + setTechnologyDefense(int i): void + getTechnologyAttack(): int + setTechnologyAttack(int i): void + getMetal(): int + setMetal(int n): int + getDeuterium(): int + setDeuterium(int n): void + getArmy(): ArrayList<MilitaryUnit>[] + setArmy(ArrayList<MilitaryUnit>[] army): void + generateDefaultArmy(): void + resetArmyArmor(): void

+ getFixArmyCost(): int[]

Battle - planetArmy: ArrayList<MilitaryUnit>[] - enemyArmy: ArrayList<MilitaryUnit>[] - armies: ArrayList[][] - battleDevelopment: String - initialCostFleet: int[][] - initialNumberUnitsPlanet: int - initialNumberUnitsEnemy: int - wasteMetalDeuterium: int[] - enemyDrops: int[] - planetDrops: int[] resourceLosses: int[][] - initialArmies: int[][] - hasCombatStarted: boolean - planetArmyPercRemaining: int - enemyArmyPercRemaining: int - attackingArmy: int - skipBattle: boolean + initInitialArmies(): void

Tiene

1

+ updateResourceLosses(): void

+ fleetResourceCost(ArrayList<MilitaryUnit> army): int

+ initialFleetNumber(ArrayList<MilitaryUnit> army): int

+ remainderPercentageFleetPlanet(): int

+ remainderPercentageFleetEnemy(): int

+ getGroupDefender(ArrayList<MilitaryUnit>[] army): int

+ getPlanetGroupAttacker(): int

+ getEnemyGroupAttacker(): int

+ resetArmyArmor(): void

+ announceCombat(): void

+ combat(): void

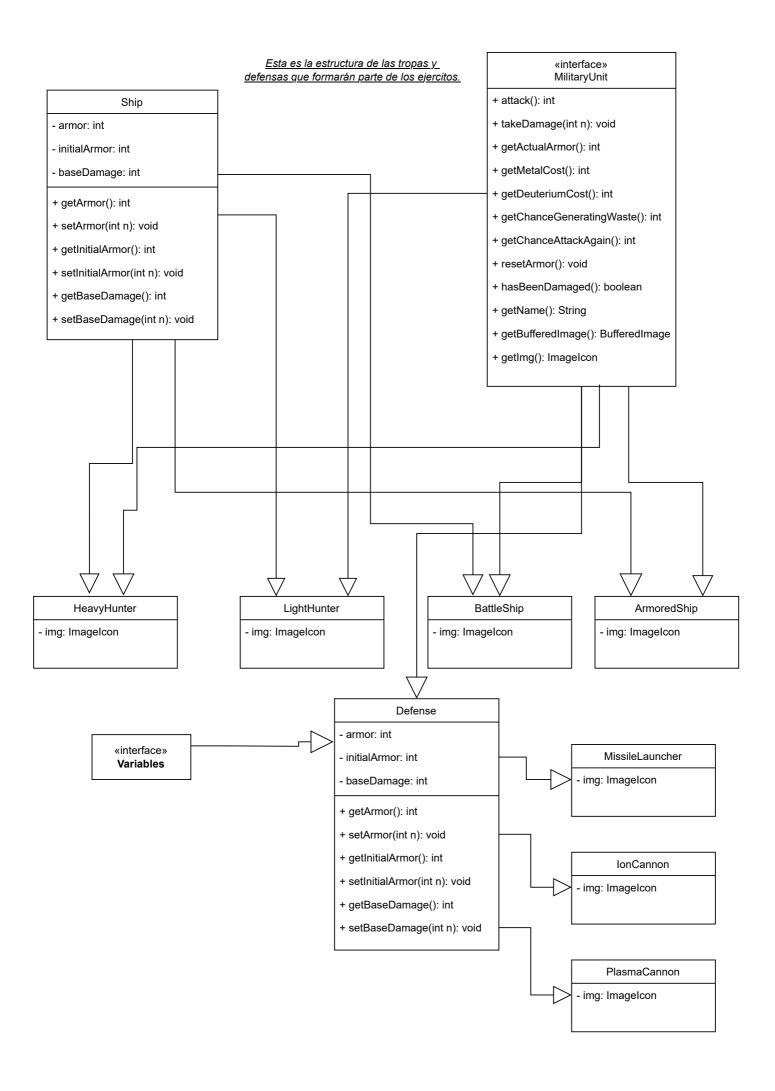
+ printEnemyStats(): void

+ getCostOfGroup(ArrayList<MilitaryUnit> group): int

+ isHasCombatStarted(): boolean

+ isSkipBattle(): boolean

+ getEnemyArmy(): ArrayList<MilitaryUnit>



GUI MainScreen - mainPanel: MainPanel LeftPanel + getMainPanel(): MainPanel - mainPanel: JPanel - infoPanel: JPanel + actionPerformed(ActionEvent e): void MainPanel - leftPanel: LeftPanel - rightPanel: RightPanel - bottomPanel: BottomPanel RightPanel - middlePanel: MiddlePanel - mainPanel: JPanel - buttonsPanel: JPanel + getLeftPanel(): LeftPanel + getRightPanel(): RightPanel + getBottomPanel(): BottomPanel + actionPerformed(ActionEvent e): void + getMiddlePanel(): MiddlePanel + updateAll(Planet planet): void MiddlePanel - activelmage: BufferedImage BottomPanel - planet: Planet - mainPanel: JPanel - shopPanel: JPanel + paintComponent(Graphics g): void + changeScreenToBattleScene(): void + actionPerformed(ActionEvent e): void + changeScreenToDefaultScreen(): void Estos frames se instancian desde los paneles superiores a través de botones ReportsFrame ResultFrame - mainPanel: JPanel - mainPanel: JPanel SettingsFrame ThreatFrame - mainPanel: JPanel - mainPanel: JPanel