MODEL Character Effetto1 -id: int -cost: int -Effect: EffectStrategy +activateEffect() ActiveCharacters EffectStrategy +getId(): int +getCost(): int +getProhibitionCounter(): int Effetto2 -randomizer: Random -characterList: ArrayList<Character> +getStudentList(): ArrayList<Student> +activateEffect() +useEffect() +incrementCost(): void +getStudent(index: int): Student -randomizeIds(): int[3] +addStudent(student: Student): void Effetto3 +getCharacter(index: int): Character +addProhibition(): void -prohibitionCounter: int +subProhibition(): void +activateEffect() Archipelago . . . GameState Effetto12 Player -studentList: ArrayList<Student> -islandsList: island[12] -indexMap: Map<Integer, Integer> -mageName: MAGE +activateEffect() -prohibitedIslands: ArrayList<Integer> -deck: AssistantDeck -players: ArrayList<Player> -whereIsMN: Integer -school: School -archipelago: Archipelago -coinCounter: int -activeCharacters: ActiveCharacters -studentList: ArrayList<Student> -tower: TOWER -clouds: ArrayList<Cloud> 1 -bag: Bag +setTower(newTower: TOWER): void +getMageName(): MAGE +getTower(): Tower +getStudentsNumByRace(race: RACE): int +addStudent(newStudent: Student): void +getDeck(): AssistantDeck +getMNposition(): Integer +getSchool(): School +getIslandsList(): island[12] +getIslandsByIndex(index: Integer): ArrayList<Island> +getCoin(): int +getIndexMap(): Map<Integer, Integer> +moveMN(value: int): void +setActiveCharacters(): void +addCoin(): void +mergelslands(index1: int, index2: int): void +subCoin(amount: int): void +takeover(index: int, ArrayList<TOWER>): ArrayList<TOWER> <<Enum>> AssistantDeck MAGE -assistantHand: ArrayList<Assistant> MERLIN -assistantDiscardPile: Stack<Assistant> JAFAR 2..3 MORGANA WONG Cloud Bag +getHand(): ArrayList<Assistant> +discardAssistant(int id): void -randomizer: Random -students: ArrayList<Student> -studentsCounters: Map<RACE, Integer> **√** 0..10 -radomize(): RACE Assistant +initializeStudents(): ArrayList<Student> +addStudents(ArrayList<Student>): void <<Enum>> +draw(number: int): ArrayList<Student> +removeStudents(): ArrayList<Student> +reinsert(students: ArrayList<Student>): void TOWER -id: int -turnWeight: int WHITE("W") -maxMNmoves: int BLACK("B") GREY("G") -abbreviation: String +getId(): int +getTurnWeight(): int +getMaxMNMmoves(): int School +Colour(abbreviation: String): void +getAbbreviation(): String -hall: ArrayList<Student> -studentsTables: Map<RACE, Stack<Student>> 0..8 -professorsTable: Map<RACE, boolean> -towers: ArrayList<TOWER> <<Enum>> RACE 0..n +getStudentsInHall(): ArrayList<Student> FAYRY("pink") +getStudentsInTable(Race: RACE): Stack<Student> Student DRAGON("red") +getProfessorInTable(Race: RACE): boolean UNICORN("blue") -race: RACE 3..4 FROG("green") +setProfessor(Race: RACE): void ELF("yellow") +addTowers(towers: ArrayList<TOWER>): void +getRace(): RACE +subTowers(quantity: int): ArrayList<TOWER> 0..n +getColour(): String -colour: String +addStudentsToHall(students: ArrayList<Student>): void +moveStudentToTable(student: Student): void 0..n + Race(colour: String): void +moveStudentToIsland(student: Student): Student +getColor(): String