Sprint 2

Rules

In this sprint we include Rules for Things, e g it's ok to pick up the bird if you have the cage (and not the Rod).

Bird rule

The bird can be picked up iff (if and only if):

- The player has the Cage in her/his inventory
- The player does NOT have the Rod in her/his inventory

Pirate Chest rule

You can pick up (and unlock and win the game) the Pirate Chest, only if you have all the keys.

The rules are created and added to the RuleBook in the new version of `CaveInitializer`. Download it if you have a version without this feature.

There is also a test, `se.itu.game.test.ThingRuleTest` which tests that a fake rule works for picking up a key, and then a test for picking up the Pirate Chest without and with all the keys.

New responsibilities (and behavior) for `Player`

- `+describeCurrentRoom() : String`
- `+currentRoomThings() : Thing [*]`
- `+canSeeRoomIn(Room.Direction) : boolean`

Two new Exception classes added

- `IllegalMoveException` - for when some one tries to make the `Player` go in an illegal direction
- `RuleViolationException` - for when some one tries to make the `Player` pick up a `Thing` against the `ThingRule` for the `Thing` in the `RuleBook`

New protocol for Player's `go(Room.Direction)` method

The method now throws `IllegalMoveException` if some one tries to go in a `Room.Direction` for which there is no connected `Room`

New interface added - `ThingRule`

A new interface will be added - `se.itu.game.cave.ThingRule` . This interface declare one abstract method:
`+apply() : boolean {exceptions=RuleViolationException}`

New class - `RuleBook`

We will add a class, `RuleBook`, which will hold all the `ThingRule`s and which we can query for the `ThingRule` for a `Thing`. See UML for details. The `ThingRule`s will be added automatically from the `CaveInitializer`, provided by the teachers.

New implementation of `Player`'s `takeThing(Thing)` method

The method now declares that it throws `RuleViolationException` (see above), because it now gets the `ThingRule`, from the `RuleBook`, for the `Thing` to take, and applies the rule. If the The `ThingRule` doesn't allow the thing to be taken, the `ThingRule` throws a `RuleViolationException` - which will be thrown from the `takeThing(Thing)` method. The GUI now must perform `player.takeThing(thing)` in a try-block which catches `RuleViolationException`.

Changes to the GUI

The GUI must be updated to reflect these new design decisions:

- `player.go(Direction)` must now be put inside try-catch(`IllegalMoveException`)
- `player.takeThing(Thing)` must now be put inside try-catch(`RuleViolationException`)
- We shouldn't use the `Player`'s `currentRoom` any more, but rather use
    - `player.describeCurrentRoom()`
    - `player.currentRoomThings()`
    - `player.canSeeDoorIn(Direction)`

TODO: for the teachers

- Implement death-rooms - where it's Game Over if the Player enters
    - Will be implemented in `CaveInitializer` - some rooms will be dead-ends with no connecting rooms and a description of "Game Over"

Excluded objects and other stuff (in this version/sprint):

- Snake
- Dragon
- Room rules (room rules)