RoomRule

Here be dragons

```
/*
 *   Xlibint.h - Header definition and support file for the internal
 *   support routines used by the C subroutine interface
 *   library (Xlib) to the X Window System.
 *
 *   Warning, there be dragons here....
 */
```

Final features for Sprint 3

In this, final, sprint we will implement some kind of rules for the Snake and Dragon which are described in the original specification for the Cave Game.

Two rooms are special, according to the original specification.

One harbors a Snake and as long as the snake is there, there is no exit to the South (which we need to visit).

The other harbors a Dragon which prevents the Glass Key from appearing.

Original specification revisited

Room 19, here's **a Snake** *blocking* the South exit. The Snake *disappears* if we *carry* the Cage **with** the Bird and drop them down - then the Southern exit *will be set to* 29.

Room 120. In the room is **a Dragon**. If we drop down Gold, Jewels, Diamonds and Silver, the Dragon will *disappear* and the West exit *will be set to* 0. Finally, a Glass Key *will appear*."

Original specification revisited - Snake

Room 19, here's **a Snake** *blocking* the South exit. The Snake *disappears* if we *carry* the Cage **with** the Bird and drop them down - then the Southern exit *will be set to* 29.

In our implementation of the CaveInitializer, the Room 19 has no exit to the South. But the description of the room says nothing about the Snake. So we need a way to optionally provide information about creatures such as the Snake.

We'll interpret "cage with bird" as put down both, one at the time.

We need to discover that the cage and bird has been put down.

Original specification revisited - Dragon

Room 120. In the room is **a Dragon**. If we drop down Gold, Jewels, Diamonds and Silver, the Dragon will *disappear* and the West exit *will be set to* 0. Finally, a Glass Key *will appear*."

We'll ignore the change of the West exit, and focus on the Glass Key. Once again, there is no description in the Room about the Dragon.

We also need to discover that the valuables have been put down, so that we can add the Glass Key to the Room.

Requirements of the RoomRule

- Need to be connected to a certain Room
- Need to store a description about the creature (snake, dragon)
- Need to detect when things have been dropped down

Requirements of the RoomRule

- Need to be connected to a certain Room
- Need to store a description about the creature (snake, dragon)
- Need to detect when things have been dropped down

Suggestion - a constructor which accepts and stores the Room and description String.

State and constructor - indicates a Class, not an Interface.

The creatureDescription can be acquired using a method which returns it.

Requirements of the RoomRule

- Need to be connected to a certain Room
- Need to store a description about the creature (snake, dragon)
- Need to detect when things have been dropped down

When the rule is applied, it should check what stuff are in the room.

Could we use a method for applying? What should it return? Does it need arguments?

How does it check if the required things have been put down?

Requirements of the RoomRule

If we need to check the room to see if stuff required to trigger some action has been put down, when is a good time to do this check?

Let's look at the Snake room. The Snake is there preventing the South exit to be created until we have put down both the Bird and the Cage.

When we first enter the Room, there are no things in it (except the Snake but that's not a Thing). So, the Player puts down the Bird, to see what happens.

In order for the magic to happen, we must check the rule.

So, let's apply the rule, everytime a Thing is being put down. Where is that, in the code?

Requirements of the RoomRule

How should the User playing the game be notified about the presence of the Snake?

Where and how is the User playing the game notified about the description of a Room in general?

Requirements of the RoomRule

How should the User playing the game be notified about the presence of the Snake?

Where and how is the User playing the game notified about the description of a Room in general?

The GUI asks the Player object to describeCurrentRoom(), right?

So this should be a good place to apply any RoomRule and ask the rule for the creatureDescription (if any). That description could be appended to the room description.

RoomRule - creatureDescription()

If we add a method creatureDescription() to the RoomRule class, how do we know what description to return?

The default description “Here’s a Snake”, for instance, should change if we have put down both the Bird and the Cage in the snake room.

So, creatureDescription() could actually call apply() before returning the creatureDescription String, and if the rule is triggered, apply will have changed the String to return.

Two places to check the RoomRule

- player.describeCurrentRoom()
  - Get the RoomRule
  - return the current room’s description + the rule’s creatureDescription
- player.dropThing(thing)
  - Get the RoomRule
  - Apply the roomRule

Two places to check the RoomRule

- player.describeCurrentRoom()
  - Get the RoomRule
  - return the current room’s description + the rule’s creatureDescription

Example dry-run: Player enters the Snake room.

- Gui gets the description via player.describeCurrentRoom
- Player gets the RoomRule
- Player returns: “You are in the Hall of the Mountain King, with passages off in all directions.” + “\nA snake blocks the south exit”

Two places to check the RoomRule

- player.dropThing(thing)
  - Get the RoomRule
  - Apply the roomRule

Example dry-run:

- Still in the Snake room player drops Bird
- Player gets the RoomRule and applies it - nothing changes
- Player drops Cage
- Player gets the RoomRule and applies it
  - The rule detects both cage and bird - opens the South door and sets creatureDescription to “There’s no snake here!”

Two places to check the RoomRule

Example dry-run:

- Still in the Snake room player drops Bird
- Player gets the RoomRule and applies it - nothing changes
- Player drops Cage
- Player gets the RoomRule and applies it
    - The rule detects both cage and bird - opens the South door and sets creatureDescription to “There’s no snake here!”

When the GUI updates, it will detect the new door to the South and also that the description changed to:
“You are in the Hall of the Mountain King, with passages off in all directions. There’s no snake here!”

RoomRule should be a class

But we'd like to be able to represent both the snake rule and the dragon rule as a RoomRule - but with different implementations of the apply() method.

What if we make RoomRule an abstract base class?

Then we can inherit from RoomRule for the snake rule and override the apply() method.

The apply() method can be asbstract and still be called from the creatureDescription() method in the base class.

Recap - abstract classes

- A class with at least one abstract method must be declared abstract
  - An abstract class doesn't need to have abstract methods though
- The class can have a mix of abstract and concrete methods
- A conctrete method can call an abstract method (virtual method)
- Abstract classes are meant to be abstractions defining behavior
- You cannot instantiate an abstract class

RoomRule

- Should store a room and a creatureDescription
- Should have a creatureDescription() method
  - Can be concrete - will always do the same
  - Can call abstract methods like apply()
  - Could be final - what does that mean?
- apply() - abstract method which will differ between subclasses
  - Should check the rules and trigger changes

Check github for an example of how to create...

On github , cave-sprint-3 , you can check how the CaveInitializer creates RoomRule:s in the addRules() method.

It stores them in RuleBook, so let’s think about that for a moment.

RuleBook needs to be expanded

- We need new Map<Room, RoomRule> to store the rules
- We need a new method to store rules (for the CaveInitializer class)
- We need a new method to look up RoomRules for a Room
  - If no rule is found, we should return a Do-Nothing-rule
    - No (empty) creatureDescription
    - apply() should do nothing

The GUI must be changed slightly

- When the User clicks a thing to drop down, we must update the whole GUI and not just the models for the thing lists
- That’s all
- But it is cheating a little…
- Read up on Observer/Observable (design pattern)

Summary

- Create a RoomRule class (we have a stub for you on github)
- Change Player to get the RoomRule in
  - describeCurrentRoom() to include the creatureDescription in the general description
  - dropThing(Thing) to aplly the rule after each drop to trigger changes
- Change RuleBook so that it
  - Stores a map of Room-> RoomRule
  - Can have new rules added to the map
  - Can give the RoomRule for a given Room
- Change the GUI so that after each drop thing action, updates the whole GUI to detect new description, new Thing:s and new Exits (if any happen due to the rules)