Assignment 5

SE1 Proseminar, winter 2007/2008

Stefan Resmerita
stefan.resmerita@cs.uni-salzburg.at

Exercise 1

Use the Decorator pattern to implement the following commands:

`PowerOn` ***`<robot type> ["<display string>"] [beep]`***

This is an extension of the **`PowerOn`** command from the previous assignment:
- The user can optionally specify a string to be displayed along with the identification tag of this robot. This is useful when the user wants to track specific robots in the system.
- If the **`beep`** parameter is present, this indicates that a sound should be issued whenever this robot finishes a job execution.

It is assumed that the user will indicate display strings and/or beeps only for few robots. Upon successful execution of this command, the id tag of the robot is displayed, as in the previous assignment.

`ShowContents <id>`

This command displays the contents of a team, showing the id tag of each member team and the type and id tag of each member robot. Moreover, if the robot has a display string, this is printed near the id of the robot. If the robot was given beeping ability, then "beeping" should be displayed near the robot's id. See the example below for the display format.

Input/ouput example:

```
>CreateTeam TransportRobot=2
2001
>CreateTeam PaintingRobot=1 CleaningRobot=1
2002
>CreateTeam PaintingRobot=2 2001 2002
2003
>PowerOn TransportRobot
107
>PowerOn TransportRobot "This is Hercules."
108
>PowerOn PaintingRobot "This is Picasso." beep
109
>PowerOn CleaningRobot beep
110
>ShowContents 2003
Begin Team 2003
 Begin Team 2001
  TransportRobot 101
  TransportRobot 102
 End Team 2001
 Begin Team 2002
```

```
  PaintingRobot 103
  CleaningRobot 104
 End Team 2002
PaintingRobot 105
PaintingRobot 106
End Team 2003
>AddToTeam 2001 108
>AddToTeam 2002 109
>AddToTeam 2003 110
>ShowContents 2001
Begin Team 2001
 TransportRobot 101
 TransportRobot 102
 TransportRobot 108 This is Hercules.
End Team 2001
>ShowContents 2003
Begin Team 2003
 Begin Team 2001
  TransportRobot 101
  TransportRobot 102
  TransportRobot 108 This is Hercules.
 End Team 2001
 Begin Team 2002
  PaintingRobot 103
  CleaningRobot 104
  PaintingRobot 109 This is Picasso. beeping
 End Team 2002
PaintingRobot 105
PaintingRobot 106
CleaningRobot 110 beeping
End Team 2003
```

Exercise 2

Use the Chain of Responsibility design pattern to implement the following requirement. We want to add the possibility to associate a short description string to a team. Modify the implementation of the *CreateTeam* command the previous assignment to accept an additional optional string parameter (see examples below). Then implement the following command:

GetTeamDescription <*id*>

The parameter *id* represents the identification tag of a robot or a team. If *id* indicates a team with a description string, then this string is printed to standard output. If *id* corresponds to a team with no description or to a robot, then its *closest* parent team with a description string is found and that string is displayed. If no parent team has a description string, then nothing is printed.

Input/output example:

```
>CreateTeam TransportRobot=2
2001
>CreateTeam PaintingRobot=1 CleaningRobot=1 “Team for painting and cleaning”
2002
```

```
>CreateTeam PaintingRobot=2 2001 2002 “Heterogeneous team 3”
2003
>PowerOn TransportRobot
107
>PowerOn CleaningRobot
108
>AddToTeam 2001 107
OK
>AddToTeam 2002 108
OK
>GetTeamDescription 107
Heterogeneous team 3
>GetTeamDescription 108
Team for painting and cleaning
>GetTeamDescription 2002
Team for painting and cleaning
>GetTeamDescription 2001
Heterogeneous team 3
```

The checking interface for this assignment should be the same as for the previous assignment.