programming in breve

CS263: Artificial Intelligence in 3D Virtual Worlds
Object-orientation

- We program with objects
- Objects do computations and hold data
- A class is a specific type of object
- We create instances of these classes
- Classes inherit behaviors from their parent classes
- An agent in the simulated world is an instance of the class “Real”
- Instances of the “Abstract” class do not appear in the world
breve Class Hierarchy

- Object
  - Real
    - Mobile
      - MultiBody
      - Braitenberg Vehicle
    - Stationary
  - Abstract
    - Control
    - Data
  - WUBAgent
The “Controller”

- Must be a subclass of “Control”
- Created automatically when the simulation starts
- Provides services to agents
- Coordinates communication between agents and the simulation software
Implementing objects

- Methods (code)
- Data (variables)

```plaintext
ParentClass : ClassName {
    + variables:
        ...
        ...

    + to init:
        ...
        ...

    + to ...:
        ...
        ...
}
```
an "agent" in breve

virtual world

"init"

"iterate" loop

sensors (input)

computation

actuators (output)
init & iterate

• init: code run when object is created
• iterate: code run at every iteration

+ to init:
  ⋯

+ to iterate:
  ⋯
  super iterate.
print-statement

• prints expressions to the output log

• print expression.

• print expression, expression, ...

Examples:

print 4 + 5.

print “the answer is”, x.

print “the object is $self”.
Creating an instance

• Uses an object definition to create an instance of that object
• Also called “instantiating” an object
• new ObjectName.

Examples:

new BraitenbergVehicle.

10 new BraitenbergLights.
Summary so far...

• We create agent objects and specify their behaviors (using the init and iterate methods)

• We create a controller object to setup the simulation and make instances of other objects (from its init method)

• We can print expressions using print and created new instances using new
WUBWorld...

- “Connect to Server” (command-K) from Finder
- Find server “Urza”
- Login as cs263, password cs263
- Select “Course Storage”
- Open cs263/Handouts
- Drag “WUBWorld” to desktop
Exercise

- Start from the myWUBWorldTemplate.tz file
- Create an Agent subclass
- Add `init` & `iterate` methods
- make the Agent’s `init` method print out “Hello, world!”
- Instantiate your Agent using the controller

What you’ll need...

+ to `methodname`: declare a method

`new`: instantiate an object

`print`: print a message
Calling instance methods

• “tell” an instance to perform an action

• can pass in and return data

• instance method [ keywords and values ].

Examples:

leftSensor link to rightWheel.

self set-color to (1, 0, 0).

self do-stuff with-x 100 with-y 200.

time = (controller get-time).
Built-in instance variables

- `self`, the object itself
- `controller`, the controller object
- `super`, the parent object (of the class “superclass”)
- `super.iterate`: says to also use the iterate behavior of the parent class
“WUBWorldControl”

• subclass of Control

• the parent class for our simulation controller

• useful methods:
  
  • get-time
  
  • watch item cameraTarget
  
  • aim-camera at cameraLocation
“Agent”

- the parent class for our Agents
- subclass of Mobile
- Useful methods:
  - `turn-left`
  - `turn-right`
  - `set-speed to floatValue`
  - `get-angle to vectorValue`
  - `get-closest-food`
  - `detect-edge`
if-statement

• Tests whether an expression is true

• if expression: { ... } else { ... }

Examples:

```python
if (self get-angle to (0, 0, 0)) < 0: {
    self turn-left.
} else {
    self turn-right.
}
```

```python
if x == 1: print "yippee!!!".
```
Summary

• We call methods to tell agents to do things
• We make use of behaviors inherited from parent classes (Mobile, WUBAgent, Control, etc.)
• We can use “if” statements to make decisions on how to behave
Things to try...

• Make the Agent follow a specific pattern (circle, figure-8, etc.)
• Make the Agent find the food
• Make the Agent follow the edges
• Create a second agent and play “tag”