Basics of Object-Orientation

4-23-2012
Opening Discussion

- Do you have any questions about the quiz?
- Last assignment due 5/7.

Minute essay comments
- Could our maze algorithm be used for a physical robot?
- Could we make a maze like PacMan? 3-D?
- When will solar panels hit grid parity?
- How terrified should you be of the final?
- Assignment option running out?
- Would $20T in rare-Earths crash the market?
Robots impersonating humans?
I wasn't planning on having autograding for quizzes.
Mazes

- Slow in the worst case because this does all possible paths.
We have been dealing with objects all semester, but we haven't really faced object-orientation head on.

The OO paradigm is characterized by encapsulation, the grouping of data and functions together into objects.

The data is called members and the functions are called methods.

The idea is that an object knows some things and how to do some things.
Scala is a class-based OO language. In the code we write classes which act as the blueprints of objects.

- These start just like the case classes we saw before, but the word case isn't required.
- Put the body of the class in curly braces after the declaration and arguments.
Differences from Case Classes

- Arguments aren't visible by default. Put `val/var` in front to see them in outside code.
- Have to be made with `new`.
- Code in the body of the class is executed immediately at creation.
- Functions defined in the body are methods of the objects.
- Data defined in the class are members of the objects.
- You can make things private.
Making Objects

- The class is only a blueprint. To get an object we have to instantiate an instance form the class.
  - new ClassName(arguments)
  - This expression can be assigned to values or passed into functions. The type is the name of the class.
- Once you have an object you can access members and methods using the dot notation.
object Declarations

- You can declare singleton objects with the keyword “object”.
- An object doesn't take arguments.
- You can declare methods and members in the object.
Applications

- We have been playing with scripts. To make an application you put a main method in an object.
  - def main(args:Array[String]):Unit = { … }
- Compile with scalac and run with scala. (Just give the object name, no .scala.)
Applications typically aren't written with command-line tools. Instead we use an Integrated Development Environment, IDE.

Eclipse is such a program.

- Free download from eclipse.org.
- Scala plug-in from scala-ide.org.

It is installed on these machines. Let's have you run it.
Questions?