Australian National University

Inheritance 2

05

java.lang.Object
Final classes, methods and fields
Abstract classes and methods

Object as superclass

In Java, all classes ultimately inherit from one root class:
`java.lang.Object`

Implemented methods you might need:

- `toString()` – we have seen this a lot already
- `hashCode()` – to support certain data structures (C4/A4/A6)
- `equals(Object obj)` – to support notions of equality other than ==
  - NOTE: if you override `equals`, you also need to override `hashCode` and guarantee that two equal values also produce equal hashCodes

Object as superclass

In Java, all classes ultimately inherit from one root class:
`java.lang.Object`

Implemented methods you will see later (maybe not in this class):

- `notify()`
- `notifyAll()`
- `wait()` – (multiple overloadings)

These methods have to do with synchronization/concurrency, which we will touch on in J16/C7

Object as superclass

In Java, all classes ultimately inherit from one root class:
`java.lang.Object`

Other implemented methods:

- `finalize()` – **deprecated** / do not worry about it
- `clone()` – a bad way of copying objects
- `getClass()` – returns run time class of object; sometimes useful, generally avoid

Final Classes and Methods

The `final` keyword

- In a class declaration states that the class ***cannot be*** subclassed

- In a method declaration states that the method ***cannot be*** overridden

Abstract Classes and Methods

The `abstract` keyword

- In a class declaration states that the class is abstract, and therefore cannot be directly instantiated (its subclasses may be, if they are not abstract)

- In a method declaration states that the method declaration is abstract; the method must be provided by a subclass. Only valid within an abstract class.