Nested Classes

Jonathan Windle

University of East Anglia

J.Windle@uea.ac.uk

May 19, 2017

Overview I

Intro

- `Enum` types are a means of modelling a variable with a discrete, finite number of values.
- They provide type safety, modularity, clarity and flexibility.
- They are instances of Anonymous inner classes.
- In C++ they are just integers, and can be treated as such, in Java, they are in fact each an instantiation of an anonymous inner class.

Enum Extra Features

- Because it's a class, it has built in methods such as:
  - `values()`: Returns the possible values as an array.
  - `ordinal()`: Returns the rank of the enum as an integer.
- `values()` can be used to iterate over enums using `for each`.

Advanced usage of Enums

- Can essentially give an enum a constructor and treat them as a class, e.g:

```
enum Grade{
// Create instances of Grade:
FIRST(70), TWO_ONE(60), TWO_TWO(50), THIRD(40), FAIL(0);
// Define class data:
        final int boundary;
        Grade(int x){
                boundary=x;
        }
        public double getBoundary(){return boundary;}
}

Grade g=Grade.FIRST;
System.out.print("g = "+g.boundary);
```

- They can be mutable (Can change data) generally don't use enums in this case, but can be useful as they can model the singleton pattern.

Summary

- Ensure Type safety, errors detected at compile time.
- Clarity, makes code easier to understand.
- Modularity, add values to existing enum without changing any code that uses the enum type.
- Flexibility, built in function are handy.

The End