First Class Functions

Computer Science Social Climbing

What is "first class"?

A programming language is said to have **first-class functions** if it treats functions as "first-class citizens" (like data) This means:

- the language supports *passing functions as arguments* to other functions,
- *returning functions as the values* from other functions,
- and *assigning functions to variables* or storing them in data structures

- From Wikipedia First Class Function

Early Java : Second Class Functions

- Define an interface that requires a single abstract function
  - e.g. "Comparator" requires a "compare" function
    - int compare(T o1, T o2) where T is any "type"
- Make a concrete class which implements the interface
- Pass a reference to an object in the concrete class as an argument or return value or assignment
- User can invoke the required function by invoking the function using that reference

Demonstrating Function Passing

- We have an array of Car objects that is not Comparable
  - Fields for Make/Model/Year/Owner
- There is an Arrays.sort static method
  - Officially: public static <T> void sort(T[] a, Comparator<? super T> c)
  - In our case: public static void sort(Car[] a, Comparator<Car> c)
- Second parameter is a reference to an object that is in a class that impelements the Comparator<Car> interface… i.e. supports int compare(Car c1,Car c2)

Strategy 1: Verbose, but clear

- Make a new class "CompareByYear" that implements Comparator<Car> in a java file: "CompareByYear.java"
- Pass in either an explicit reference to a CompareByYear object, or a new CompareByYear() to make a reference to a new comparator object.

- Every different compare function needs a new class and a new .java file

Strategy 2: Multi Class .java file

- Exactly like strategy 1, except, instead of putting the CompareByMake class in it's own .java file, just put it in the file where it will be invoked
- Compiler will not allow the comparator class to be public!
- Compiler generates "CompareByMake.class" file

Strategy 3: Named Inner Class

- Put a "CompareByModel" class inside the TestCar class
- Requires a reference to a TestCar object in order to resolve the inner class.
- But the inner class can now be "public"… it becomes a "member" of the TestCar class

- Compiler generates "TestCar$CompareByModel.class" file

Strategy 4: Anonymous Inner Class

- Java allows an un-named single object class that implements Comparator<Car>.
- We can create a reference to that object, or we can bypass keeping a reference (single use)
- Compiler generates "TestCar$1.class" and "TestCar$2.class"
- For a long time, this was the "best" way to pass a function as an argument

"Lambda Expression" Intro

- Introduced in Java 8
- Finally "full class functions"!
  - at least from the programmer's point of view
  - Under the covers, this is still anonymous inner classes

- (parm1,parm2) - > function of parm1 and parm2

- Defines an anonymous method
- If the lambda appears in the context of a single abstract method interface, the lambda is assumed to implement that interface's method!