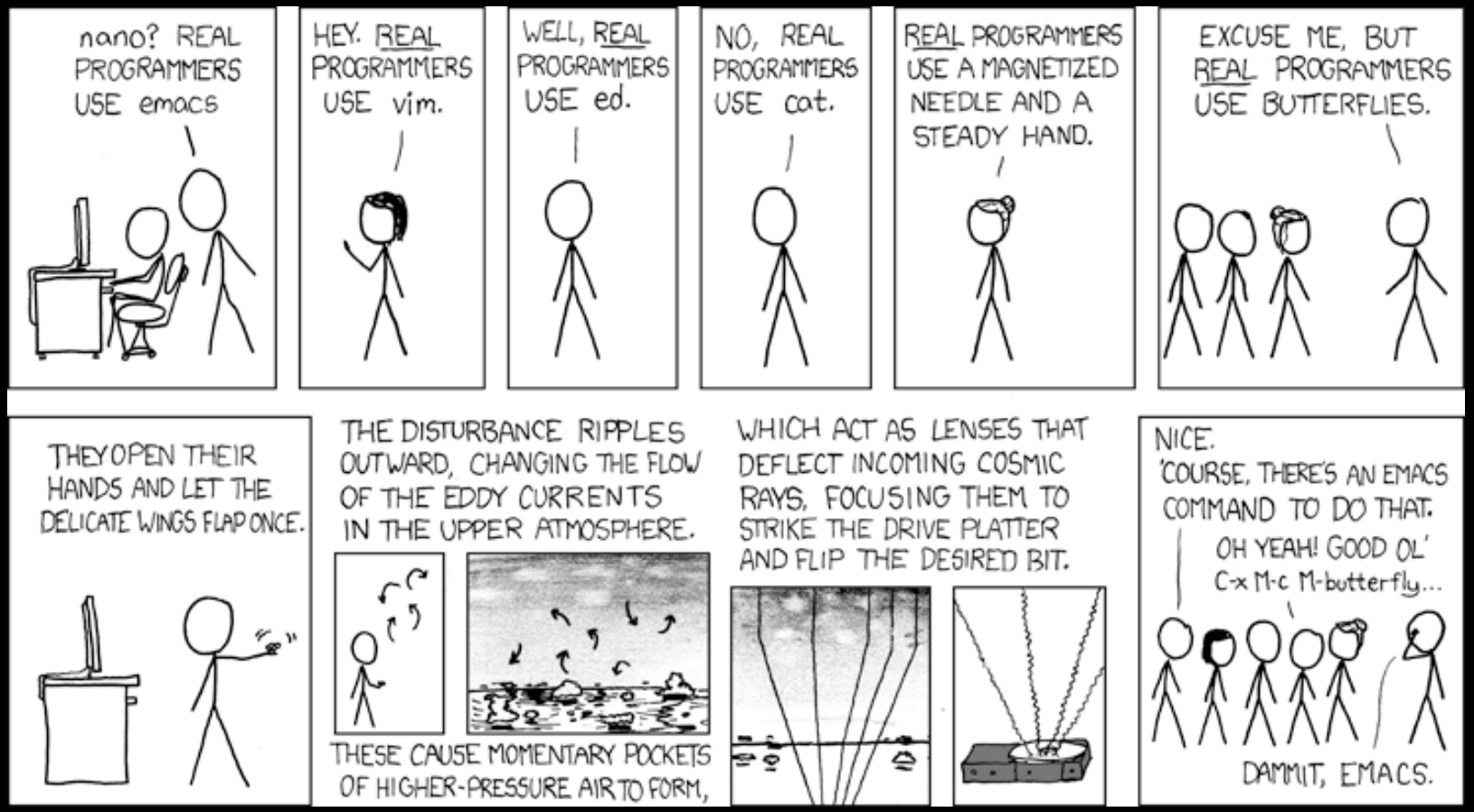


http://xkcd.com/378/

CS 152: *Programming Language Paradigms*

Returning to Java

Prof. Tom Austin
San José State University

Returning home to Java

It's the last day of class, so let's do something simple in Java…

…sort a list of numbers.

Sorting a list of numbers in Java 1

```
public static void sortNums (List lst) {
  for (int i=0; i<lst.size()-1; i++) {
    for (int j=0; j<lst.size()-1; j++) {
      if (((Integer) lst.get(j)).intValue() >
          ((Integer) lst.get(j+1)).intValue()){
        Integer tmp = (Integer) lst.get(j);
        lst.set(j, lst.get(j+1));
        lst.set(j+1, tmp);
      }
    }
  }
}
```

Now we can call our sorting algorithm:

```
List lint = new ArrayList(
    Arrays.asList(1,2,93,-1,3));
sortNums(lint);
```

Except that we could also call:

```
List lstr = new ArrayList(
    Arrays.asList("hi","there"));
sortNums(lstr);
```

Generalizing our sort algorithm

```
public static void sort (List lst,
                         Comparator cmp) {
  for (int i=0; i<lst.size()-1; i++) {
    for (int j=0; j<lst.size()-1; j++) {
      if (cmp.compare(lst.get(j),
                      lst.get(j+1)) > 0) {
        Object tmp = lst.get(j);
        lst.set(j, lst.get(j+1));
        lst.set(j+1, tmp);
      }
    }
  }
}
```

But calling this function is a little ugly:

```
sort(lint, new Comparator() {
   public int compare(Object o1,
                      Object o2) {
      Integer x = (Integer) o1;
      Integer y = (Integer) o2;
      return x.intValue()
           - y.intValue();
}});
```

Using generics (Java 5)

```
public static <T> void sort (List<T> lst,
                             Comparator<T> cmp) {
  for (int i=0; i<lst.size()-1; i++) {
    for (int j=0; j<lst.size()-1; j++) {
      if (cmp.compare(lst.get(j),
                      lst.get(j+1)) > 0) {
        T tmp = lst.get(j);
        lst.set(j, lst.get(j+1));
        lst.set(j+1, tmp);
      }
    }
  }
}
```

And calling this gets a little better:

```
sort(lint, new Comparator<Integer>() {
  public int compare(Integer x,
                     Integer y) {
    return x - y;
 }});
```

Still, compare that to the equivalent in JavaScript:

```
sort(lint, function(x,y){
  return x-y;
});
```

Java 8 Closures

Java 8 introduces lambdas (closures).

We can now write this function more concisely:

```
sort(lint,
   (Integer x,Integer y) -> x-y);
```

Extended Closure Example

(in class)

A (Partial) List of Function Interfaces

Interface	Parameter types	Return type
`Supplier<T>`	`None`	`T`
`Consumer<T>`	`T`	`void`
`BiConsumer<T,U>`	`T, U`	`void`
`Predicate<T>`	`T`	`boolean`
`ToIntFunction<T>`	`T`	`int`
`Function<T,R>`	`T`	`R`
`BiFunction<T,U,R>`	`T, U`	`R`

Limitations of Java Lambdas

- Java lambdas are *not* objects

```
// COMPILE ERROR!
Object o = (x) -> x+1;
```

- Java lambdas only close over *values*, not variables

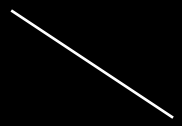
Counter class

```
Supplier<Integer> ctr =
        Counter.makeCounter();
out.println(ctr.get()); // 0
out.println(ctr.get()); // 1
out.println(ctr.get()); // 2
```

Broken `makeCounter` method

```
import java.util.function.Supplier;

public class Counter {
    public static Supplier<Integer> makeCounter() {
        int n = 0;
        return () -> n++; // error
    }
}
```



"Local variable n defined in an enclosing scope must be final or effectively final"

Working `makeCounter` method

```
import java.util.function.Supplier;

class IntHolder {
    int n = 0;
}

public class Counter {
    public static Supplier<Integer> makeCounter() {
        IntHolder ih = new IntHolder();
        return () -> ih.n++;
    }
}
```

Heap allocated memory, so modification is OK. The *reference* is not modified (effectively final).

Lab: Lambdas

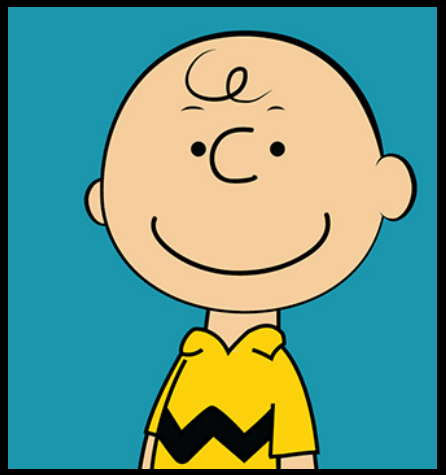
Today, you will write a class to list files using Java 8 lambdas.

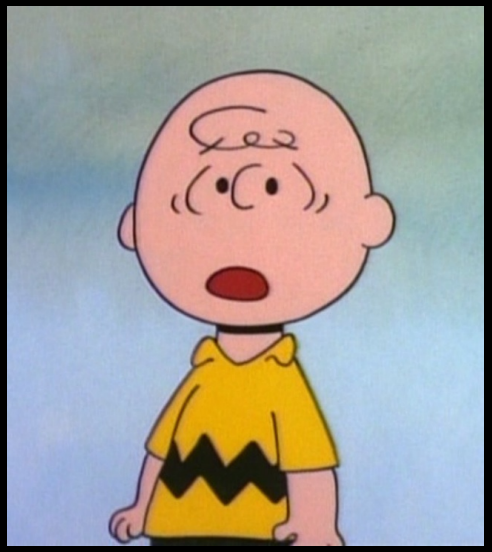
Details are in Canvas.

Final thoughts:

how this class has warped you beyond repair

When you stared this class, you knew how to program.





But now, maybe you are not so sure anymore.

Before, learning a new language might have seemed like a huge task

Now perhaps it does not seem so daunting.

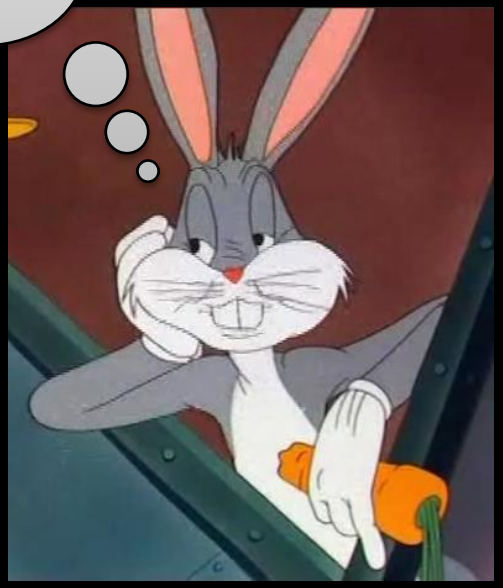
Hopefully, you will see more elegant solutions

I'll need 63 different classes to handle each case.

One lambda should solve this…

New options may open up to you

My IDE does not support Java's newest syntax. Oh well.



Hmm… It looks like I just new to change a few lines of the grammar…

Your view on languages may change

Warning: This way lies madness.


"That's all Folks!"