Custom Components, Testing, Inner Classes

Graphics Context — an object that will paint for you
Buffers — 2D raster of pixels
Goal: Custom component that paints a copy of a given component

Original

has a graphics context

Copy

Constructor

Same-sized
Create a buffer
Copy component into the buffer

Painting the copy:
Painting the buffer at the right size
Testing hierarchies of classes

Applicable to BlackBox Testing — use specification to generate test cases
GlassBox Testing — use the code to drive test cases

* Leverage existing test cases
  by extending them — parallels class hierarchy of classes under test
1. **Factory method to create test instance**
   public abstract AbstractCollection getTestInstance();

2. Several tests for the various methods of Abstract Collection
   public A getTestInstance() & return new A();

---

getsize(), add(x), remove(x), contains(x)
Sortable

Abstract Collection

A

B

foo()

D

C

CollectionTest

A

B

@test

DTest

CTest

Since B implements Sortable, call "canned" tests for the sortable interface.
Inner Classes

1. static nested class (not really inner classes) treated like top-level classes, but the name is `OuterClassName$Foo` => No special privileges use static members of containing class

2. member class — not static declared inside another class each has an implicit reference to a containing instance the compiler adds a hidden parameter to all constructors of a hidden instance variable

3. local class class defined inside a method

4. anonymous class just like local classes, but they have no name! declared inside of a statement
```java
class MyCollection {
    static DUMMY_TAIL = ... Object[] a = ...

    static class MyIterator {
        nested class use DUMMY_TAIL (static) can't use instance variables
        int currentLocation; Type: MyCollection.MyIterator can't use a (instance variable)
    }
}

class MemberIterator {
    can use a — have an implicit reference to an instance of MyCollection
    MyCollection.this is the containing object
```
If want to create an instance of a member class from outside the code of the containing class

In Foo:

```
MyCollection coll = ...;
MyCollection.MemberIterator it = coll.new MemberIterator(i);
```

To RARE!
Local Class:

```
Iterator iterator() {
    final int i = 0;
    final Foo f = new Foo(...);
    class MyIterator {
        ...
    }
    return new MyIterator();
}
```

Anonymous class:

```
Iterator iterator() {
    return new Iterator() {
        public boolean hasNext() {
            ...;
        }
    };
}```