CE0825a - Object Oriented Programming II
8: Memory, Java Native Access, Animation

James A Sutherland

Abertay University

Monday, 29th February 2016

Memory Basics

- Generally, it's just a single bunch of bytes, numbered from 0 upwards.
- (DEC Alpha is different!)
- Operating systems group them into *pages*, usually 4k.

Memory Basics

- Generally, it's just a single bunch of bytes, numbered from 0 upwards.
- (DEC Alpha is different!)
- Operating systems group them into *pages*, usually 4k.

Memory Basics

- Generally, it's just a single bunch of bytes, numbered from 0 upwards.
- (DEC Alpha is different!)
- Operating systems group them into *pages*, usually 4k.

Stacks and Heaps

- Usually, there is a *stack* – short term storage, arguments. Often where buffer overflows target: see Return Oriented Programming later on.
- Also a *heap*, for longer term allocations.
- Good news: Java takes care of all this for us!

Stacks and Heaps

- Usually, there is a *stack* – short term storage, arguments. Often where buffer overflows target: see Return Oriented Programming later on.
- Also a *heap*, for longer term allocations.
- Good news: Java takes care of all this for us!

Stacks and Heaps

- Usually, there is a *stack* – short term storage, arguments. Often where buffer overflows target: see Return Oriented Programming later on.
- Also a *heap*, for longer term allocations.
- Good news: Java takes care of all this for us!

Garbage Collection

Just ask for chunks of memory, then forget about them:
Someone Else's Problem!
Basic variants of that:

- Reference counting
- Mark-sweep
- Generational
- No-op: don't bother! (Good for transient utilities: see Busybox)

Garbage Collection

Just ask for chunks of memory, then forget about them: Someone Else's Problem!

Basic variants of that:

- Reference counting
- Mark-sweep
- Generational
- No-op: don't bother! (Good for transient utilities: see Busybox)

Garbage Collection

Just ask for chunks of memory, then forget about them: Someone Else's Problem!
Basic variants of that:

- Reference counting
- Mark-sweep
- Generational
- No-op: don't bother! (Good for transient utilities: see Busybox)

Garbage Collection

Just ask for chunks of memory, then forget about them: Someone Else's Problem!
Basic variants of that:

- Reference counting
- Mark-sweep
- Generational
- No-op: don't bother! (Good for transient utilities: see Busybox)

Memory Leaks

Still possible in Java (and other garbage collected systems)

```
Vector<String> v=new Vector<String>();
for (int i=0;true;i++) {
        v.add("blah"+i);
}
```

Usually subtler than that, of course.

Types

If everything is just a number ... what's $0x107b214a7$?
As it happens it's a function inside 64 bit Java for OS X ...
but could also be a timer, a buffer, someone's password...
Java protects you (a bit) by not letting you use pointers directly. You can't just read off the end of a String.

Going Native: JNA

So how do we access native platform functions? Enter JNA: Java Native Access.

- Previous: JNI, a bit cumbersome
- JNA: Friendly wrappers for e.g. Pointer

Get the JNA library JAR from `https://github.com/java-native-access/jna`. (Inside the most recent Release; open the zip, find `jna.jar` under the `dist` folder.)

Going Native: JNA

So how do we access native platform functions? Enter JNA: Java Native Access.

- Previous: JNI, a bit cumbersome
- JNA: Friendly wrappers for e.g. Pointer

Get the JNA library JAR from `https://github.com/java-native-access/jna`. (Inside the most recent Release; open the zip, find `jna.jar` under the `dist` folder.)

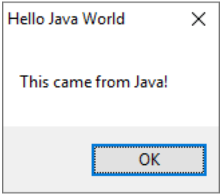
An actual whole program!

```
import com.sun.jna.*;

public class Week8a {
        public interface User32 extends Library {
                User32 i =
                    (User32)Native.loadLibrary("user32.dll",
                        User32.class);
                int MessageBoxA(Pointer hwnd,String
                    msg,String title ,int type);
        }

        public static void main(String[] args) {
                User32.i.MessageBoxA(null, "This came from
                    Java!", "Hello Java World", 0);
        }
}
```

Results



Hello Java World

This came from Java!

How SWT works

Every system API call is a DLL function, like MessageBoxA. So, SWT defines a Java counterpart like `int MessageBoxA(Pointer hwnd,String msg,String title,int type);` for each of them, and can then call them straight from Java.

Animation 1

```
canvas = new Canvas(shell, SWT.NO_BACKGROUND);
canvas.addPaintListener(new PaintListener() {
  public void paintControl(PaintEvent event) {
    Image image = new
      Image(shell.getDisplay(),
      canvas.getBounds());
    GC gcImage = new GC(image);
    gcImage.setBackground(event.gc.getBackground());
    gcImage.fillRectangle(image.getBounds());
    gcImage.setBackground(shell.getDisplay().getSyst
    gcImage.fillOval(x, y, IMAGE_WIDTH,
      IMAGE_WIDTH);
    event.gc.drawImage(image, 0, 0);
    image.dispose();
    gcImage.dispose();  } });
```

Animation 2

```
Runnable runnable = new Runnable() {
  public void run() {
          if (display.isDisposed()) return;
    animate();
    display.timerExec(TIMER_INTERVAL, this);
  }
};
display.timerExec(TIMER_INTERVAL, runnable);
```

Animation 3

```
public static void animate() {
x += directionX;
y += directionY;

// Determine out of bounds
Rectangle rect = canvas.getClientArea();
if (x < 0) {
  x = 0;
  directionX = 1;
} else if (x > rect.width - IMAGE_WIDTH) {
  x = rect.width - IMAGE_WIDTH;
  directionX = -1;
}
// ... slide break here ...
```

Animation 3 cont

```
// ... slide break here ...
if (y < 0) {
  y = 0;
  directionY = 1;
} else if (y > rect.height - IMAGE_WIDTH) {
  y = rect.height - IMAGE_WIDTH;
  directionY = -1;
}

// Force a redraw
canvas.redraw();
```

Lab Task 8

1. Find a useful API call (`http://msdn.microsoft.com/`) and call it via JNA.
2. Animate something other than a circle.[1]

[1] Full example code came from here: `http://www.java2s.com/Code/Java/SWT-JFace-Eclipse/DemonstratesanimationItusesdoublebuffering.htm`

Lab Task 8

1. Find a useful API call (`http://msdn.microsoft.com/`) and call it via JNA.
2. Animate something other than a circle.[1]

[1] Full example code came from here: `http://www.java2s.com/Code/Java/SWT-JFace-Eclipse/DemonstratesanimationItusesdoublebuffering.htm`