References

Chris Piech
CS106A, Stanford University
Learning Goals

1. Be able to write a large program
2. Be able to trace memory with references
Today, we build!
Advanced memory model
Core memory model
```java
public void run() {
    println(toInches(5));
}

private int toInches(int feet) {
    int result = feet * 12;
    return result;
}
```
public void run() {
    println(toInches(5));
}

private int toInches(int feet) {
    int result = feet * 12;
    return result;
}
public void run() {
    println(toInches(5));
}

private int toInches(int feet) {
    int result = feet * 12;
    return result;
}
public void run() {
    println(toInches(5));
}

private int toInches(int feet) {
    int result = feet * 12;
    return result;
}
public void run() {
    println(toInches(5));
}

private int toInches(int feet) {
    int result = feet * 12;
    return result;
}
public void run() {
    println(toInches(5));
}

private int toInches(int feet) {
    int result = feet * 12;
    return result;
}
Aside: Actual Memory
What is a bucket

feet

5
What is a bucket

* Each bucket or “word” holds 64 bits

Piech, CS106A, Stanford University
#0: don’t think on the binary level (yet)
End aside
# Primitives vs Classes

<table>
<thead>
<tr>
<th>Primitive Variable Types</th>
<th>Class Variable Types</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int</code></td>
<td><code>GRect</code></td>
</tr>
<tr>
<td><code>double</code></td>
<td><code>GOval</code></td>
</tr>
<tr>
<td><code>char</code></td>
<td><code>Gline</code></td>
</tr>
<tr>
<td><code>boolean</code></td>
<td><code>Color</code></td>
</tr>
</tbody>
</table>

Class variables (aka objects)
1. Have upper camel case types
2. You can call methods on them
3. Are constructed using `new`
4. Are stored in a special way
How do you share wikipedia articles?

Antelope Canyon Article

Antelope Canyon is a slot canyon in the American Southwest. It is located on Navajo land east of Page, Arizona. Antelope Canyon includes two separate, photogenic slot canyon sections, referred to individually as Upper Antelope Canyon or The Crack, and Antelope Canyon or The Corkscrew.[2]

The Navajo name for Upper Antelope Canyon is Tsé bighánílíí, which means "the place where water runs through rocks." Lower Antelope Canyon is Hazdistazí (advertised as "Hasdeestwazi" by the Navajo Parks and Recreation Department), or "spiral rock arches." Both are located within the LeChee Chapter of the Navajo Nation.[4]

Contents [hide]
1 Geology
2 Tourism and photography
2.1 Upper Antelope Canyon

https://en.wikipedia.org/wiki/Antelope_Canyon
Objects store addresses
(which are like URLs)
What does an object store?
Objects store addresses
(which are like URLs)
A Variable love story

By Chris Piech
A Variable Story

origin

love

Nick Troccoli

By Chris Piech
Once upon a time...
...a variable x was born!

```c
int x;
```
...a variable x was born!

```
int x;
```
x was a primitive variable...

```c
int x;
```

Aww...!

It’s so cuuuute!
...and its parents loved it very much.

```c
int x;
```

We should give it.... value 27!
...and its parents loved it very much.

\[ x = 27; \]

We should give it... value 27!
A few years later, the parents decided to have another variable.
...and a variable rect was born!

GRect rect;
rect was an object variable...

GRect rect;

Who’s a cute GRect???

It’s so square!
...and its parents loved it very much.

GRect rect;

We should make it... a big, strong GRect!
...and its parents loved it very much.

GRect rect = new GRect(0, 0, 50, 50);

We should make it.... a big, strong GRect!
GRect rect = new GRect(0, 0, 50, 50);

...but rect’s box was not big enough for an object!

That box isn’t big enough to store everything about a GRect!
...so they stored the information in a bigger box somewhere else.

```java
GRect rect = new GRect(0, 0, 50, 50);
```

- `x = 0, y = 0`
- `width = 50`
- `height = 50`

See location 5

Location 5
Chapter 2: Coming soon
public void run() {
    GRect r = null;
}

Method memory

Object memory
public void run() {
    GRect r = null;
}

Wahoo!
public void run() {
    GRect r = new GRect(50, 50);
}

Method memory

Object memory

run

r
```java
class Memory {
    public void run() {
        GRect r = new GRect(50, 50);
    }
}
```

**Method memory**

run

r

**Object memory**

memory.com/18
public void run() {
    GRect r = new GRect(50, 50);
}

Method memory

run

r

memory.com/18

Object memory

memory.com/18
public void run() {
    GRect r = new GRect(50, 50);
}

Method memory

Object memory

run

r

18
```java
public void run() {
    GRect r = new GRect(50, 50);
}
```
```java
public void run() {
    GRect r = new GRect(50, 50);
    r.setColor(Color.BLUE);
    r.setFilled(true);
}
```
public void run() {
    GRect r = new GRect(50, 50);
    r.setColor(Color.BLUE);
    r.setFilled(true);
}

Method memory

Object memory
public void run() {
    GRect r = new GRect(50, 50);
    r.setColor(Color.BLUE);
    r.setFilled(true);
}

Method memory

run

r

Object memory
#1: **new** allocates memory for objects

* The data for an object can’t always fit inside a fixed size bucket
#2: object variables store addresses

#ultimatekey
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListener();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }

}
What does an object store?
Objects store addresses (which are like URLs)
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListeners();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }

}
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListeners();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }

}
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListeners();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementType(1, 1);
        remove(obj);
    }
}
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListener();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }
}
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListeners();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }
}

Memory

Instance Variables

canvas

memory.com/12

run

r

www.memory.com/12

Object Memory

www.memory.com/12

www.memory.com/12
Memory

Instance Variables

canvas

run

r

www.memory.com/12

Object Memory

www.memory.com/12
Memory

Instance Variables

canvas

run

r

12

Object Memory

12
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListeners();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }
}

Piech, CS106A, Stanford University
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListener();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }

}
public class SimpleRect extends GraphicsProgram {

    public void run() {
        GRect r = null;
        r = new GRect(300, 300);
        r.setColor(Color.MAGENTA);
        add(r, 0, 0);
        addMouseListeners();
    }

    public void mousePressed(MouseEvent e) {
        GObject obj = getElementAt(1, 1);
        remove(obj);
    }

}
What does an object store?
Objects store addresses (which are like URLs)
Finish Up
Learning Goals

1. Be able to write a large program
2. Be able to trace memory with references