CS150 - Multiple Inheritance & Deeper Understanding of Objects

Note Title 3/26/2012

Announcements

- HW due Friday

Quiz recap

class Student(Person):

#2 functions

- called parent versions (with appropriate parameters)

Last time

Inheritance with cs1 graphics.

Examples:

- star (from Polygon)

- Car (from Layer)
- car (from Drawable) - see p. 316

Multiple Inheritance

Only supported by some languages.

Allows you to use code from multiple classes.

Ex: LabeledRectangle in cs1graphics

This object is both Text + Rectangle

Inherits: from Text: setMessage, setFontColor...
from Rectangle: setWidth, setFillColor...

(from both: move, scale, etc.

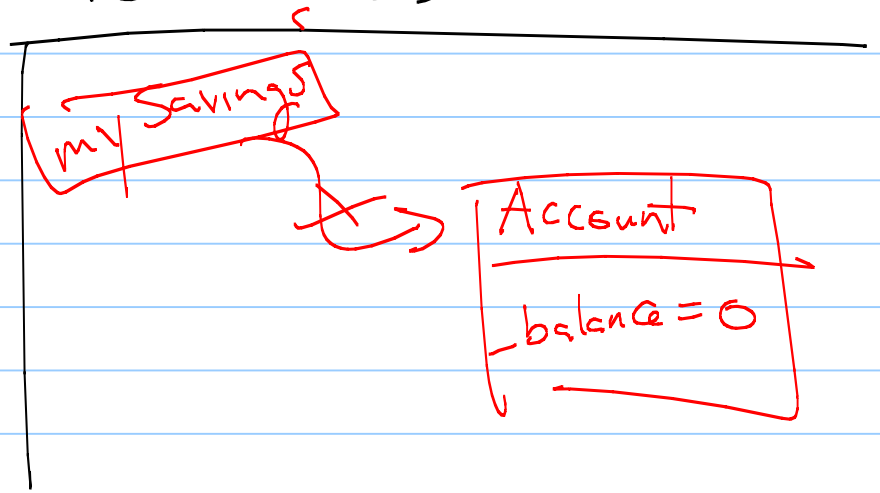
Ch 10: Deeper understanding of objects

Example of a simple, mutable object: Account class

Suppose we say:

```
mySavings = Account()
```

Picture of memory:



- mySavings is really a reference to the actual Account
  - We view it as a pointer to the object.
  - Each call to the constructor creates a new Account object.

Like: myChecking = Account()

(Try the id command - the 2 labels will be distinct)

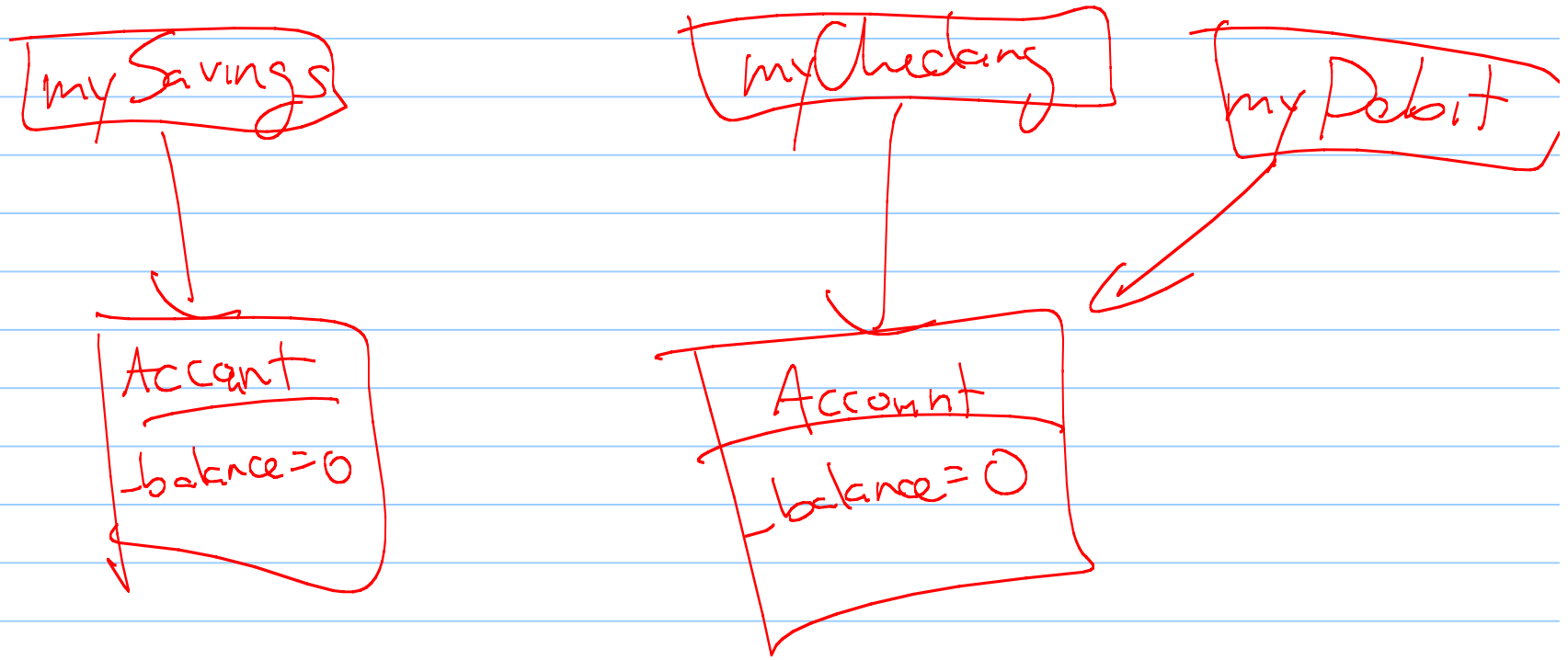
References to the same object

Now try:

```
myDebit = myChecking
id(myDebit)
```

So myDebit is an alias, & changing it will change both.

```
myDebit.deposit(100.0)
myChecking.getBalance()
```


mySavings
myChecking
myDebit
Account
balance=0
Account
balance=0

Equivalence Testing

- Can use is:

  mySavings is myChecking

  myChecking is myDebit

  myChecking is myChecking

How does it work?

checks id

Operator ==:

In contrast, operator== is a broader notion of equivalence.

```
mySavings.deposit(100)
mySavings == myChecking
```

→ true

(Remember, we coded this one.)