CSSE 220 Day 18

Inheritance recap
`Object`: the superest class of all
Inheritance and text in GUIs

Nothing to check out from SVN

Questions?

Inheritance Review

- Sometimes a new class is a special case of the concept represented by another
- The new class inherits from the existing one:
  - all methods
  - all instance fields
- Change just what we need
  - Don’t redeclare fields!
  - Don’t redeclare methods which are good enough
  - But overload ones that aren’t
  - Make use of super.method and super() as needed.

I, Object

- The superest class in Java

Object

- **Every** class in Java inherits from `Object`
  - Directly and **explicitly**:
    - `public class String extends Object {…}`
  - Directly and **implicitly**:
    - `class BankAccount {…}`
  - **Indirectly**:
    - `class SavingsAccount extends BankAccount {…}`

Q1

Object Provides Several Methods

- `String toString()`
- `boolean equals(Object otherObject)`
- `Class getClass()`
- `Object clone()`
- …

Often overridden

Sometimes useful

Often dangerous!

Q2

Overriding toString()

- Return a concise, human-readable summary of the object state
- Very useful because it's called automatically:
  - During string concatenation
  - For printing
  - In the debugger
- `getClass().getName()` comes in handy here…

Q3

Overriding equals(Object o)

- Should return true when comparing two objects of same type with same "meaning"
- How?
  - Must check types—use getClass()
  - Must compare state—use **cast**

Recall that the cast would throw a new ClassCastException if the object isn't THIS_TYPE

```
@Override
public boolean equals(Object object) {
    if (this.getClass() == object.getClass() {
        THIS_TYPE other = (THIS_TYPE)object;
        // Then compare this and other's fields.
    }
    return false;
}
```

Q4

Polymorphism and Subclasses

- A subclass instance **is a** superclass instance
  - Polymorphism still works!
  - `BankAccount ba = new SavingsAccount();`
    `ba.deposit(100);`
- But not the other way around!
  - `SavingsAccount sa = new BankAccount();`
    `sa.addInterest();`
- Why not?

BOOM!

Q5

Another Example

- Can use:

  -
    ```
    public void transfer(double amt, BankAccount o){
        withdraw(amount);
        o.deposit(amount);
    }
    ```
    in BankAccount
- To transfer between different accounts:
  - `SavingsAccount sa = …;`
  - `CheckingAccount ca = …;`
  - `sa.transfer(100, ca);`

Summary

- If B extends or implements A, we can write

```
A x = new B();
```

Declared type tells which methods x can access. Compile-time error if try to use method not in A.

The actual type tells which class' version of the method to use.

- Can cast to recover methods from B:

```
((B)x).foo()
```

Now we can access all of B's methods too.

If x isn't an instance of B, it gives a run-time error (class cast exception)

Q6-8, pass in when done & start BallWorlds

BallWorlds worktime

- Whatever you don’t finish is homework due next session.

  Near the end of class, you should do item 1 on HW18: complete the partner survey for the term VectorGraphics project, since I need to form teams *before* next class.