CSE 8B Today

Inheritance!

JPanel → GraphicLetter → YOUR Graphic Letter

Graphics in Java

Casting! (How to turn a Person into a Student!)

ArrayLists (the array that never ends)

Exam 2 makeup as promised...
Inheritance Tree

Object

Person

Circus Performer

Teacher

Student

HistoryMajor

BiologyMajor

CSEMajor

All classes inherit by default

Things to keep in mind!

a HistoryMajor is a Person…
"everything" is an Object…
PSA5 Inheritance Tree

Your Graphic letter is a JPanel…

Things to keep in mind!
Why whichChar and makeCopy?

The LetterFactory (Holds GraphicLetters)

- GraphicLetter_cs8szz1
- GraphicLetter_cs8szy1
- GraphicLetter_cs8sdr1
- GraphicLetter_cs8suv2
- GraphicLetter_cs8sim2

"Give me a Z!"
(The LetterFactory knows the objects are GraphicLetters, but doesn’t know which ones)
Why whichChar and makeCopy?

The LetterFactory

- GraphicLetter_cs8szz1
- GraphicLetter_cs8szy1
- GraphicLetter_cs8sdr1
- GraphicLetter_cs8suv2
- GraphicLetter_cs8sim2

whichChar are you?

'\text{A}'

"Give me a Z!"
Why whichChar and makeCopy?

The LetterFactory

GraphicLetter_cs8szz1
GraphicLetter_cs8szy1
GraphicLetter_cs8sdr1
GraphicLetter_cs8suv2
GraphicLetter_cs8sim2

whichChar are you?

‘D’

“Give me a Z!”
Why whichChar and makeCopy?

The LetterFactory

GraphicLetter_cs8szz1
GraphicLetter_cs8szy1
GraphicLetter_cs8sdr1
GraphicLetter_cs8suv2
GraphicLetter_cs8sim2

whichChar are you?
‘Z’

Great! makeCopy of yourself!

“Give me a Z!”
In what class is the `drawLine` method defined?

A. NewPanel
B. JPanel
C. Graphics
D. `paintComponent`
E. Other
Who calls the paintComponent method in NewPanel?

A. Java automatically calls this method when it paints (or repaints) the component
B. The programmer must call this method directly on the NewPanel object
C. Java will automatically call it, but the programmer may also call it if she or he wants the painting to happen immediately.
But sometimes an exact type is not known until run-time:
- The compiler will assume the object is of the declared type.

The constructor still determines the actual type of the Object.
- At run-time, Java will use the actual type's latest (most-derived) methods.

A: Person p = new Student("Sally", 16);
B: System.out.println("p's name is " + p.name);
C: System.out.println( p.isAsSleep( 24 ) );
D: p.status( 24 );
E: System.out.println("P is taking " + p.units);

One of these lines of code will cause an error. Which one? (And how do you fix it)
Casting!

But sometimes an exact type is not known until **run-time**:  
- The compiler will assume the object is of the **declared** type.

The constructor still determines the **actual** type of the Object.  
- At run-time, Java will use the **actual type's** latest (**most-derived**) methods.

A: `Person p = new Student( "Sally", 16 );`
B: `System.out.println("p's name is " + p.name);`
C: `System.out.println( p.isAsleep( 24 ) );`
D: `p.status( 24 );`
E: `System.out.println("P is taking " + ((Student)p).units);`

Casting changes the type of the reference to the object to make the compiler believe us that the object can really do what we ask it. Casting can be dangerous, so be sure you know what you’re doing!