

Java 4

Recall

From last time's handout...

"Student s;" semantics

Substitution rule

Up/down rule

Class relationship x 3

Applied OOP #1

Encapsulation

A manages its own state. B sends A a message to request a state change on A. This is just a nice way to divide up a large program.

Applied OOP #2

Co-op with library objects

There are many "built-in" classes -- String, Vector, ... -- you create and send messages to these objects to get them to do work for you. You are the client of the library code.

This is programming with true code re-use.

The skill to research and understand the library interface grows in importance. Contrast to "raw code writing" skill.

Applied OOP #3 -- Zebra

1. "Library" Class Hierarchy
2. Custom Subclass
3. Override
4. Run-time Pop-down

Code Integration

Library Code

Insert Your Code

True "library" code integration

Array

`int[] a;` -- an array of ints

`int a[];` -- alternate syntax for C
refugees

Heap allocated

`int[] a;` -- allocs the pointer, not the pointee (the array itself)
Implementations currently zero the array

`a = new int[100];`

Run time allocate the array, just like an object

`100 == a.length`

Arrays know their length -- cool!
NOT `a.length()`

`a[0] = new Student(12); // NO -- CT`

Err

At CT, arrays know their type (int in this case) and trap errors such as above
Other Java collection classes (Vector, List) do not know their type.

Everything is an Object pointer and you cast in and out.

`Student b[] = new Student[100];`

Allocate 100 Student pointers

Does not allocate any Student objects -- that's a separate pass

String

`"Hello World!\n"`

String consts like C

Unicode

The char type in Java can be two bytes -- supports the Unicode character set, not just ASCII. This applies to char and to String.

String object is "immutable"

It never changes once created

Handy way to finesse memory sharing and multi-threading issues

String + String

```
String a = "foo";
```

```
a + " bar"
```

+ concats strings together -- creates a new String based on the other two

See the docs

"API" Documentation for the many built-in classes --

<http://java.sun.com/products/jdk/1.2/docs/api/index.html>

Look in the String class docs for the many messages it responds to

StringBuffer

Similar to String, but can change the String over time. More efficient to change one StringBuffer over time, than to create 20 slightly different String objects over time.

String Methods

int length()

boolean equals(string)

boolean equalsIgnoringCase(string)

char charAt(index)

int indexOf(char)

int indexOf(String)

String substring(index)

String substring(index, onePastEnd)

String toLowerCase()