Using Objects (cont.) / Reading input

- Review of Linux, object references
- Leftover from last time:
  - reference and value semantics
  - immutable types: String class
- Java API and packages
- using the Java API documentation
    - packages
    - method headers
- Ex of creating and using objects: reading input
    - Scanner class

Announcements

- Claire's office hours for today are on piazza.
- Permanent staff office hours will be announced on piazza later today.
- Deadline for experience-level survey is tonight (google form; you received link last week)
- Students in the following categories need to see me after class (or in office hours today):
  - missed the first lecture
  - just recently added the class
  - no programming experience
  - not officially enrolled (e.g., on waiting list)

Review of zoom lecture guidelines

- video ON during lecture encouraged
- screen name is real name, e.g.,
  Ruiwen (Ray) Li
- zoom photo: head shot
- no one will be let in from the waiting room starting next lecture. To get into lecture:
  - go to `usc.zoom.us`, log in using USC SSO before joining meeting

pollEv.com

- To take classroom polls today (and get credit for it) you have to have registered on pollEv.com with USC email.
- And you have to be logged into pollEv.com when you take the poll.
- For asynchronous participation: each poll will be available for one week after the live lecture.

Linux cp review

Suppose you already made a directory called **`lab1`** in your home directory that contains **`Hello.java`**. Now suppose you make another directory in your home directory called **`lab2`**. These are the commands you have done so far:

```
mkdir lab2
cd lab2
```

Write a single Linux command to copy **`Hello.java`** from **`lab1`** dir to a file **`Date.java`** in **`lab2`** dir.

How to take the poll:

During lecture: `pollev.com/cbono`

Asynchronous participation: Link to Linux poll

Review of object references

```
Rectangle rect;
```

Leftover from last time…

- See Thurs slides.

Java API

- API = Application Programmer Interface
- all the classes in the extensive Java library
  - Includes 1000s of classes

- We'll discuss further:
  - library organized into *packages*
  - how to find out how to use the classes provided

Java API packages

- *packages* organize sets of related classes
- Examples:
  - **`java.awt`**
    (has some graphics-related classes such as **`Rectangle`**),
  - **`java.lang`**
    (has basics such as **`String`** and **`System`**)
- to use a class, need to know the package it is in
- import statement:

```
import java.awt.Rectangle;
```

Java API documentation

- API doc.
  - available on the web (linked from our course web page)
  - can get a local copy (install JDK)
  - selected classes/methods in textbook (Appendix)
- API doc describes the *interface* for the classes:
  - i.e., what you need to know to use it

How to take the poll:

During lecture: `pollev.com/cbono`

Asynchronous participation: Link to class interface poll

What is a method header?

- Like function headers, but appear inside a class definition (we'll see class def next lecture)
- Example method call:

```
rect.translate(5, 10);
```

- Corresponding method header:

```
public void translate(int x, int y)
```

Java API Documentation

- Let's check out the API documentation on the web…

Reading console input

- use **`java.util.Scanner`**
- Pass **`System.in`** to scanner constructor:

```
Scanner in = new Scanner(System.in);
```

- Skips over whitespace, and reads in next sequence of non whitespace chars:

```
in.nextInt()
in.nextDouble()
in.next()
```
[String]

- Reads in whole line as a string:

```
in.nextLine()
```

Example: Reading input

- Small example program in **`ReadName.java`**

`nextLine`

How do you write code that combines word-by-word reading (e.g., with **`nextInt()`** or **`next()`**) and line-by-line reading (i.e., with **`nextLine()`**)?

- Specific example: write code that when run does…

```
Enter your age: 32
Enter your whole name: Joseph P. Blow
```

See **`TestReadLine.java`**