

CSSE 220 Day 10

Arrays, ArrayLists,
Wrapper Classes, Auto-boxing,
Enhanced *for* loop

Check out *ArraysAndLists* from SVN

Questions?

Exam Coming!

See the [Schedule page](#), Session 12, for a link to a document that lists the topics covered by this exam

- ▶ Test next Monday
 - Evening exam! Schedule page says where and when.
 - Exam is 7–9 p.m. but you may start the exam up to 1 hour early and stay up to 1 hour late (or both)
- ▶ Topics from Chapters 1–7
- ▶ Will include:
 - A closed-book paper part: short answer, fill-in-the-blank, trace-code-by-hand, draw box-and-pointer diagrams, find-errors-in-code, write short chunks of code
 - We will list in advance ALL the possible topics for this portion of the exam
 - A programming part: a few small programs, unit tests provided for some of them, you write unit tests for others
- ▶ Review in class Thursday
 - Bring questions
 - I won't prepare anything but am happy to cover whatever you want, including working examples

Array Types

- ▶ What it is for:
 - ▶ Bundling a collection of objects under a single name,
 - ▶ With elements in the collection referred to by their index in the collection (0, 1, 2, ...)
- ▶ Syntax for declaring: *ElementType[] name*
- ▶ Examples:
 - A local variable: `double[] averages;`
 - Parameters: `public int max(int[] values) {...}`
 - A field: `private Investment[] mutualFunds;`

Allocating Arrays

- ▶ Syntax for allocating:

new *ElementType*[length]

- ▶ Creates space to hold values

- ▶ Sets values to defaults

- **0** for number types
- **false** for boolean type
- **null** for object types

- ▶ Examples:

- **double[] polls = new double[50];**
- **int[] elecVotes = new int[50];**
- **Dog[] dogs = new Dog[50];**

Don't forget
this step!

This does NOT
construct any
Dog's. It just
allocates space for
referring to Dog's
(all the Dog's start
out as *null*)

Reading and Writing Array Elements

- ▶ Reading:

- `double exp = polls[42] * elecVotes[42];`

Sets the value
in slot 37.

Reads the element
with index 42.

- ▶ Writing:

- `elecVotes[37] = 11;`

- ▶ Index numbers run from 0 to array length – 1

- ▶ Getting array length: `elecVotes.length`

No parentheses, array
length is (like) a field

Arrays: Comparison Shopping

Arrays...	Java	C	Python
have fixed length	yes	yes	no
are initialized to default values	yes	no	n/a
track their own length	yes	no	yes
trying to access “out of bounds” stops program before worse things happen	yes	no	yes

Live Coding

Begin the ElectionSimulator program (in ArraysAndLists), per the instructions in [Homework 10](#)

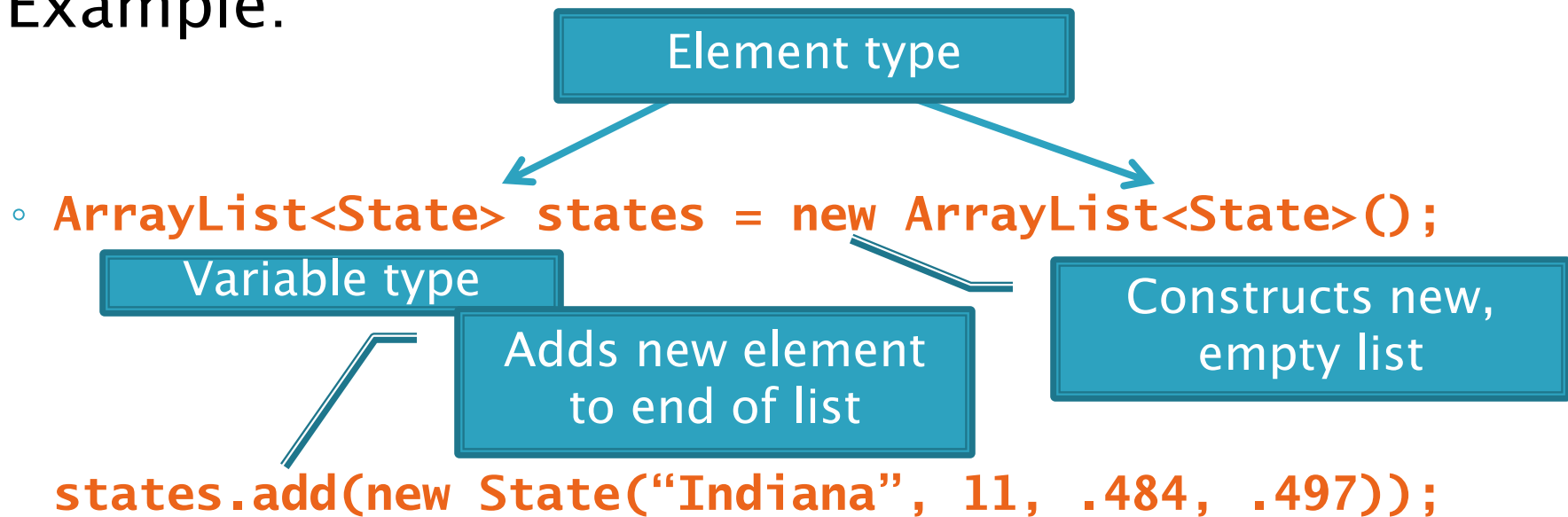


Your instructor will demo a [run of the program](#). Here is a [UML class diagram](#) for it.

You might find the [Summary on Arrays and ArrayList's helpful](#).

What if we don't know how many elements there will be?

- ▶ ArrayLists to the rescue
- ▶ Example:



- ▶ **ArrayList** is a *generic class*
 - Type in <brackets> is called a *type parameter*

ArrayList Gotchas

- ▶ Type parameter can't be a primitive type
 - Not: `ArrayList<int> runs;`
 - But: `ArrayList<Integer> runs;`
- ▶ Use *get* method to read elements
 - Not: `runs[12]`
 - But: `runs.get(12)`
- ▶ Use `size()` not `length`
 - Not: `runs.length`
 - But: `runs.size()`

Lots of Ways to Add to List

- ▶ Add to end:

- `victories.add(new WorldSeries(2008));`

- ▶ Overwrite existing element:

- `victories.set(0,new WorldSeries(1907));`

- ▶ Insert in the middle:

- `victories.add(1, new WorldSeries(1908));`

- Pushes elements at indexes 2 and higher up one

- ▶ Can also remove:

- `victories.remove(victories.size() - 1)`

Live Coding

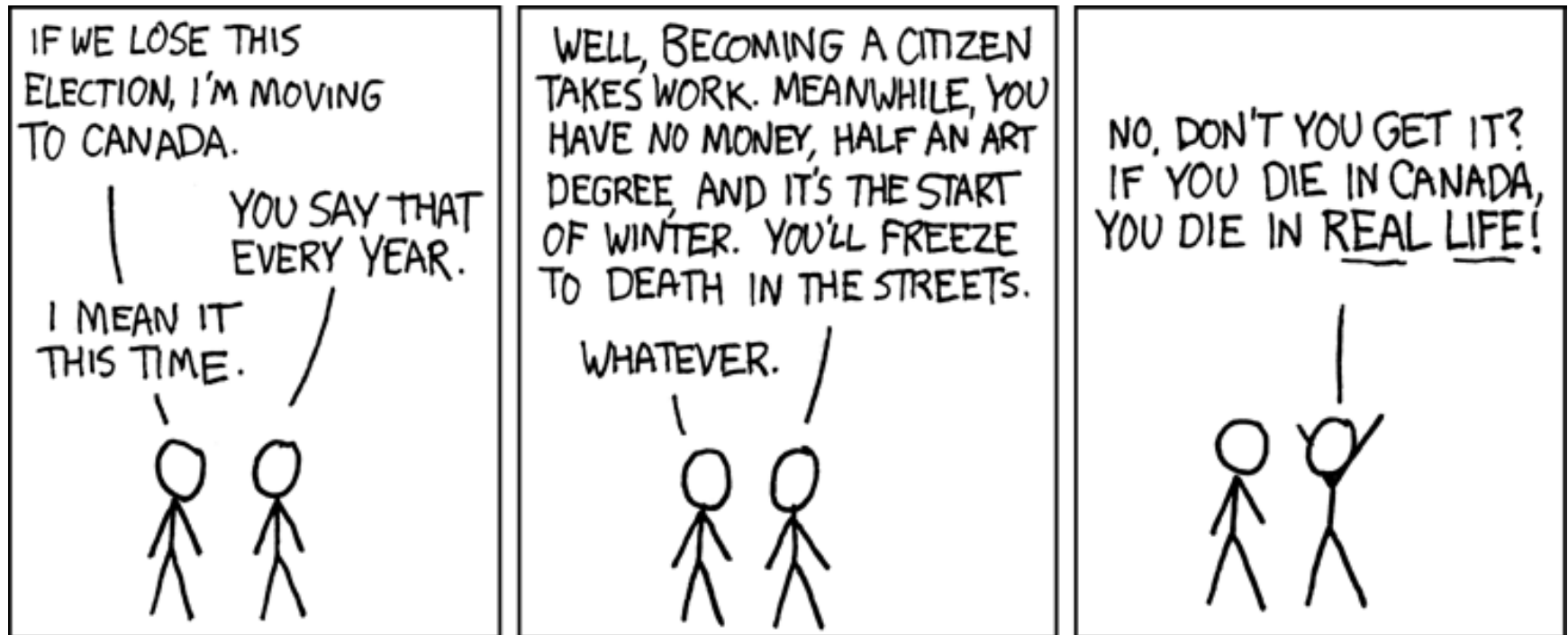
Continue the ElectionSimulator program (in ArraysAndLists), per the instructions in [Homework 10](#)



Your instructor will demo a [run of the program](#). Here is a [UML class diagram](#) for it.

You might find the [Summary on Arrays and ArrayList's helpful](#).

Cartoon of the Day



IT'S ALL REAL!

So, what's the deal with primitive types?

▶ Problem:

- ArrayList's only hold objects
- Primitive types aren't objects

▶ Solution:

- *Wrapper classes*—instances are used to “turn” primitive types into objects
- Primitive value is stored in a field inside the object

Primitive	Wrapper
byte	Byte
boolean	Boolean
char	Character
double	Double
float	Float
int	Integer
long	Long
short	Short

Auto-boxing Makes Wrappers Easy

- ▶ Auto-boxing: automatically enclosing a primitive type in a wrapper object when needed
- ▶ Example:
 - You write: `Integer m = 6;`
 - Java does: `Integer m = new Integer(6);`
 - You write: `Integer answer = m * 7;`
 - Java does: `int temp = m.intValue() * 7;`
`Integer answer = new Integer(temp);`

Auto-boxing Lets Us Use ArrayList's with Primitive Types

- ▶ Just have to remember to use wrapper class for list element type
- ▶ Example:
 - `ArrayList<Integer> runs = new ArrayList<Integer>();`
`runs.add(9);` *// 9 is auto-boxed*
 - `int r = runs.get(0);` *// result is unboxed*

Enhanced For Loop and Arrays

- ▶ Old school

```
double scores[] = ...  
double sum = 0.0;  
for (int i=0; i < scores.length; i++) {  
    sum += scores[i];  
}
```

- ▶ New, whiz-bang, enhanced for loop

```
double scores[] = ...  
double sum = 0.0;  
for (double score : scores) {  
    sum += score;  
}
```

Say “in”

- No index variable (easy, but limited in 2 respects)
- Gives a name (**score** here) to each element

Enhanced For and ArrayList's

```
▶ ArrayList<State> states = ...  
  int total = 0;  
  for (State state : states) {  
      total += state.getElectoralVotes();  
  }
```

Live Coding

TONIGHT, do the short
[Survey for assigning
partners for the Game of
Life exercise](#) on Angel,
under
Lessons ~ Assessments
(at the top, first item listed)



Continue (and strive to finish)
the ElectionSimulator program
(in ArraysAndLists), per the
instructions in [Homework 10](#)

Your instructor will demo a
[run of the program](#). Here is a
[UML class diagram](#) for it.

You might find the [Summary
on Arrays and ArrayList's
helpful](#).