CSE 11 Accelerated Intro to Programming Lecture 2

Greg Miranda, Spring 2021

Announcements Ayor

- Ayar Shihua M
- Discussion starts today @ 4pm & 5pm
- Quiz 2 due Friday @ 8am
- Survey 1 due Friday @ 11:59pm
- PAO.5 due tomorrow @ 11:59pm > visit tutors
 - PA1 released due 4/7

Example

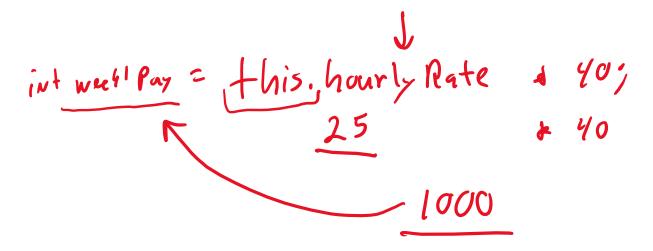
- > Class Name Create a new file class Examples2 int rate = 20;
- class will talk about more later
 - For now: describes a group of fields
- Problem:
 - Calculate the pay you would receive at a certain hourly rate given a number of hours
 - New field: # of hours worked
 - Calculate total pay using Java as a calculator

- Calculate total pay (cont.)
 - Can we use these fields in another calculation?
 - Why is it useful to do this with fields instead of writing this directly?
 - What if:
 - Use same hourly rate, but a number of different weeks to calculate?
 - What if:
 - We want to change the hourly rate?
 - Change once, changes all values
 - Many times, you will have one field whose value can be used in many places
 - Configure how the program works
 - Changing the value in one spot can affect many other places in the program
 - Powerful concept in programming:
- Variables
 Coustants

 Define a value in one place
 Change it by editing the program

 - Watch its changes be reflected in all the other places next time it's run

- Using this. hourlyRate
 - Call that a field look-up or a field access
 - Looking up the current value of a field that has been defined before



Text

- Integers (int) common kind of data programmers work with
- New kind of data also really common text
 - Examples: usernames, passwords, email, names, addresses
 - Data type for text String
- Previous examples had int as the type
 - int numberOfStaff = 14;
- Now using String as the type
 - String name = "Greg Miranda"; //String value, string literal
- String className = 11;
 - What happens? Does it work?
- String className = "11";
 - What happens? Does it work? Is it text or a number?

Types

14

"Greg Miranda

- int integer type integer literal
- String text type string literal (written in double quotes)
- Java will enforce that we always
 - store string values in String typed fields
 - numeric values in numeric typed fields
- Programmer's job to get this right
 - Java will give an error message if we don't

Java Strongly typed language

String

- We learned we can store Strings values in fields
 - What else can we do with them?
 - String fullName = "Greg" + "Miranda"; >> + Concat evation
 Will this work?
 Ne multiply Strings by a result of Can we add Strings together, like integers?
 - Can we multiply Strings by a number?
 - String str = this.firstname * 2;
 - What about Divide? Subtract?
 - What about +? Can we add a String and a number?
 - String str = this.firstname + 2;
 - What's going to happen if we try this?
 - Compiler error?
 - Works? If it works, what does it store in the str field?

- We can + other things besides numbers to Strings and get similar behavior
 - More on this in upcoming weeks
- Adding Strings and numbers
 - Can be convenient
 - Can turn a number into text
 - Can also be confusing
 - String className = "11" + 200;
 - int klassName = 11 + "200";
 - Error
 - String klassName = 11 + "200";
 - Java does do this automatic conversion of Strings and numbers
 - Be careful in your own code

Vocabulary

26/45> Name> 2 class definition class Example { int x = 3 + 2; field definitions int y = this.x * 4;(more hinds of "inside" the defs coming) class definition < Name > = < expression ? body " of the A class def has a sequence of field dets How many field definitions are in this class?

How many field definitions are in this class?

Do you think there's a limit on how many field definitions can be in a class?

- No limit



Program Steps

```
class Example {
  int x = 3 + 2;
  int y = this.x * 4;
}
```

Expressions

- int x = 3 + 2;
 - 3 + 2
 - Arithmetic expression
 - Binary operator expression
- int y = this.x * 4;
 - this.x
 - Field access expression
 - this.x * 4
 - Arithmetic expression where left hand operand is a field access expression