AP Computer Science

Syntax, simple statements

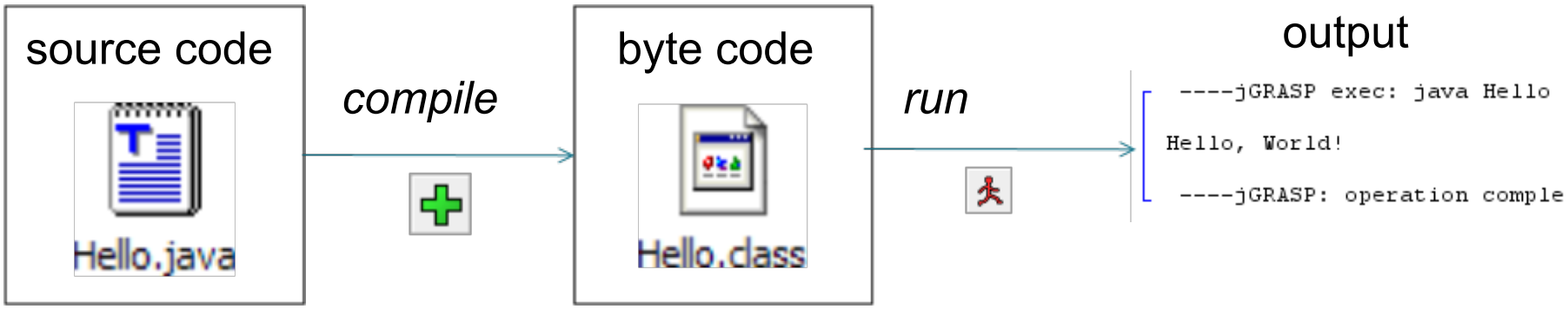
Adapted from Ms. Martin's "prntln strings" slides by Mr. Bergquist, September 2011

GOCOMICS.

GET A LAUGH!

Compiling/running programs

1. *Write* it.
   - **code** or **source code**: The set of instructions in a program.
2. *Compile* it.
   - **compile**: Translate a program from one language to another.
   - **byte code**: The Java compiler converts your code into a format named *byte code* that runs on many computer types.
3. *Run* (execute) it.
   - **output**: The messages printed to the user by a program.



Syntax

- Set of legal structures and commands that can be used in a language
  - semicolons
  - curly braces
  - identifiers
- Compiler checks syntax, gives errors

System.out.println

- A statement that prints a line of output on the console.
  - pronounced "print-linn"
  - sometimes called a "println statement" for short
- Two ways to use System.out.println :
  - System.out.println("**text**");
    Prints the given message as output.
  - System.out.println();
    Prints a blank line of output.

Strings

- Sequence of characters
- Enclosed in double quotes “This is enclosed in double quotes”
- Some characters must be *escaped* using the backslash: ‘\’
  - For example: \" \t \n \\
    - Tab: \t
    - NewLine \n

Reality check

- What is the output of the following println statements?

```
System.out.println("\ta\tb\tc");
System.out.println("\\\\");
System.out.println("'");
System.out.println("\"\"\"");
System.out.println("C:\nin\the downward spiral");
```

- Write a println statement to produce this output:

  / \ // \\ /// \\\

“Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live.”

Martin Golding

Comments

- Lets others know what's on your mind
- Explains tricky bits
- Must be used sparingly
- /* */ and //

Good:

```
/* Prints a greeting */
```

Bad:

```
/* This is my super awesome program
that uses the println statement to go
ahead and display a friendly message
to the user because it's a convention
that was started long ago.*/
```

Algorithms

- algorithm: A list of steps for solving a problem.

- Example algorithm: "Bake sugar cookies"
  - Mix the dry ingredients.
  - Cream the butter and sugar.
  - Beat in the eggs.
  - Stir in the dry ingredients.
  - Set the oven temperature.
  - Set the timer.
  - Place the cookies into the oven.
  - Allow the cookies to bake.
  - Spread frosting and sprinkles onto the cookies.
  - ...

- Potential problems with writing it this way?

Structured algorithms

- structured algorithm: Split into coherent tasks.

  1 Make the cookie batter.
  - Mix the dry ingredients.
  - Cream the butter and sugar.
  - Beat in the eggs.
  - Stir in the dry ingredients.

  2 Bake the cookies.
  - Set the oven temperature.
  - Set the timer.
  - Place the cookies into the oven.
  - Allow the cookies to bake.

  3 Add frosting and sprinkles.
  - Mix the ingredients for the frosting.
  - Spread frosting and sprinkles onto the cookies.

  ...

Static methods

A block of Java statements that is given a name.

- static method: A named group of statements
  - denotes the structure of a program
  - eliminates redundancy by code reuse

- procedural decomposition
  - dividing a problem into methods

- Writing a static method is like adding a new command to Java.

Decomposition

A separation into discernable parts, each of which is simpler than the whole.

- Decide what your related steps are
- Group the steps in a method
- Name the method descriptively
- Call your new method

```
public static void main(String[] args) {
    chorus();
}
public static void chorus() {
    System.out.println(”Hey soul sister, Ain’t that Mister Mister");
    System.out.println(”On the radio, Stereo…");
}
```

When to use methods

- Statements are closely related

- Statements are repeated
- Watch out for weakly-related statements

- You can always change your decomposition!