In this method, there is one variable. The name of the variable is phil. Its type is String. It is also a parameter. When (not now!) the method is CALLED, phil's value is the argument in the method call.
3.9 Stack diagrams

Parameters and other variables only exist inside their own methods. Within the confines of main, there is no such thing as phil. If you try to use it, the compiler will complain. Similarly, inside printTwice there is no such thing as argument.

One way to keep track of where each variable is defined is with a stack diagram. The stack diagram for the previous example looks like this:

![Stack Diagram]

For each method there is a gray box called a frame that contains the methods parameters and local variables. The name of the method appears outside the frame. As usual, the value of each variable is drawn inside a box with the name of the variable beside it.
3.9 Stack diagrams

Parameters and other variables only exist inside their own methods. Within the confines of main, there is no such thing as phil. If you try to use it, the compiler will complain. Similarly, inside printTwice there is no such thing as argument.

One way to keep track of where each variable is defined is with a stack diagram. The stack diagram for the previous example looks like this:

For each method there is a gray box called a frame that contains the methods parameters and local variables. The name of the method appears outside the frame. As usual, the value of each variable is drawn inside a box with the name of the variable beside it.
Parameter vs. Argument

- VARIABLE declared in the ( ... ) of a method CALL.
  - Receives a value each time the method is called.
  - That value is usually used by the method.
  - Like any other variable, the method may change its value.

- VALUE expressed or computed in the ( ... ) of a method CALL.
  - It is copied to a parameter during the calling operation.
  - It is the value that is used.
  - A variable expressing an argument doesn't change value when a parameter is assigned.
Parameter vs. Argument

- VARIABLE declared in the ( ... ) of a method CALL.

- Lives (uses memory) in the frame of the CALLED method.

- VALUE expressed or computed in the ( ... ) of a method CALL.

- When it comes from a variable, that variable lives (uses memory) in the frame of the CALLING method.

- AS A CONSEQUENCE: A variable expressing an argument doesn't change value when a parameter is assigned.

> (Parameters are sometimes called "Dummy Arguments")
Parameter vs. Argument

- VARIABLE declared in the ( ... ) of a method CALL.
- Comp. Sci. Jargon: **FORMAL PARAMETER or ARGUMENT**
- Sometimes called "**Dummy Arguments**"

- VALUE expressed or computed in the ( ... ) of a method CALL.
- Comp. Sci. Jargon: **ACTUAL PARAMETER or ARGUMENT**
• What happens if you invoke a method and you don’t do anything with the result (i.e. you don’t assign it to a variable or use it as part of a larger expression)?

• What happens if you use a print method as part of an expression, like `System.out.println("boo!") + 7`?

• Can we write methods that yield results, or are we stuck with things like `newLine` and `printTwice`?

Yes, we can. Use `return`

For the other two questions, TRY THEM (Downey's advice) ....