Java Review Session 3

CS 5004

Topics Covered

Java Syntax

- Type casting
- String formatting
- Interfaces (quick review)
- Inheritance (quick review)
- Comparable Interface
- More Java Collections

What is the `new` keyword? Object allocation (create a new instance of the object)

Package - a set of related classes

How do our programs actually run?

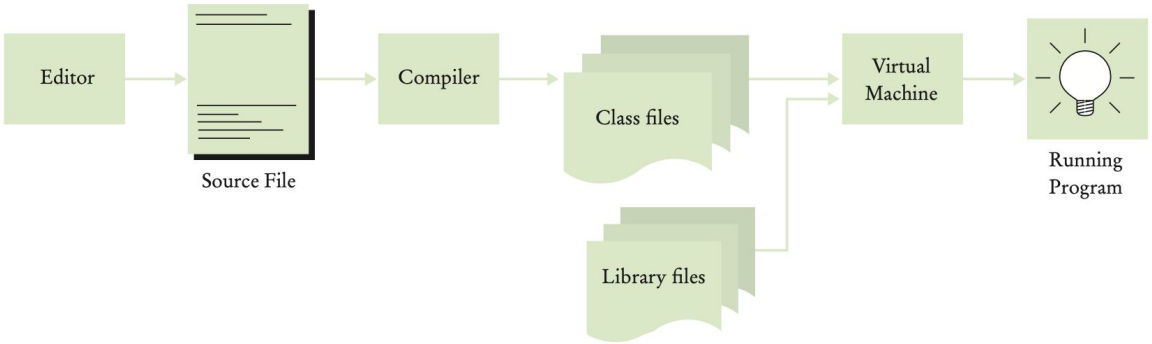

Figure 6 From Source Code to Running Program

Image: Cay Horstmann

When you compile a `.java` source file (human readable code), the Java compiler translates it into platform-independent bytecode stored in a `.class` file (a set of instructions that can be executed by the JVM)

Library files contain implementation of classes and methods commonly used in Java programs, can be loaded in during runtime to provide functionality to your program (standard library to bytecode)

In IDEs the compiler and the JVM are automatically executed when you run your Java program

Type casting

What happens if you want to convert a double into an int? Will this work?

```
double account_bal = 1250.12

int dollars = account_bal
```

Is this a good idea?

```
double account_bal =
1250.12

int dollars = (int)
account_bal
```

Syntax 4.2 Cast

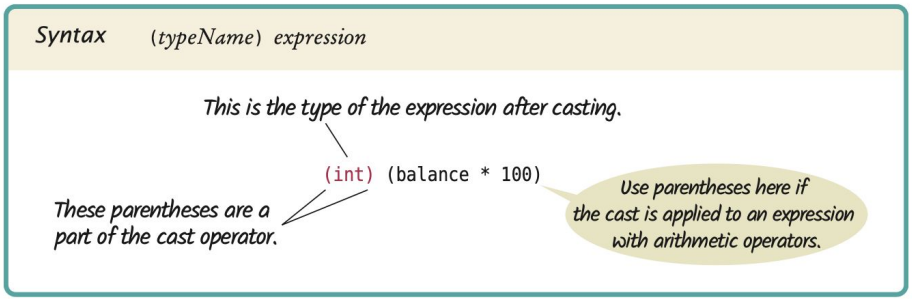

Image: Cay Horstmann

Formatting output

```
String.format("%.2f", value);
```

Table 6 Format Specifier Examples

Format String	Sample Output	Comments
`"%d"`	`24`	Use `d` with an integer.
`"%5d"`	`  24`	Spaces are added so that the field width is 5.
`"Quantity:%5d"`	`Quantity:  24`	Characters inside a format string but outside a format specifier appear in the output.
`"%f"`	`1.21997`	Use `f` with a floating-point number.
`"%.2f"`	`1.22`	Prints two digits after the decimal point.
`"%7.2f"`	`  1.22`	Spaces are added so that the field width is 7.
`"%s"`	`Hello`	Use `s` with a string.
`"%d %.2f"`	`24 1.22`	You can format multiple values at once.
`"Hello%nWorld%n"`	`Hello` `World`	Each `%n` causes subsequent output to continue on a new line.

Image: Cay Horstmann

Inheritance - brief review

Object - the cosmic superclass

.equals() compares whether to objects have the same contents

== tests whether two references are identical (if they are referring to the same object)

Savings account instanceOf bank account?

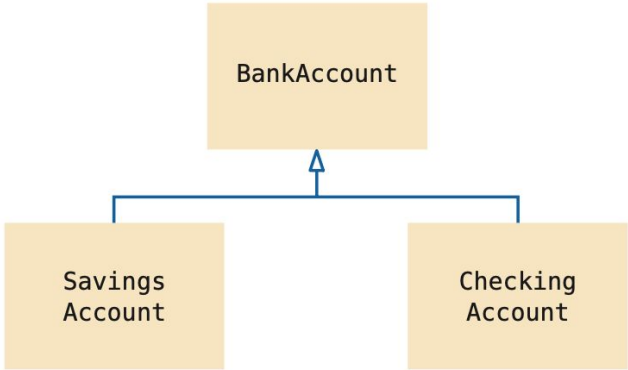

Image: Cay Horstmann

Interfaces

Interfaces are similar to classes with four key differences

- Interfaces do not have instance variables (interfaces have no state)
- All methods in an interfaces must be abstract (no implementation details)
- All methods in an interface are automatically public
- An interface does not have a constructor
    - Interfaces are not classes you cannot construct objects of an interface type

- Classes implement interfaces

- Interfaces don’t have instance variables, but can have constants

    - all variables in interfaces are automatically public static final

    - although allowed, it’s not very common

Interfaces - why?

- Interfaces are used to define a contract for classes - specifies the methods that an implementing class must provide (without specifying how)
- Java does not allow multiple inheritance of classes, but you can implement multiple interfaces (classes can only extend a single super class)

Inheritance/Interface Demo

Demo of a Bank account, Savings account, and Checking account

The Comparable Interface

- We implement the comparable interface so that objects of your class can be compared (useful for applications such as sorting)
- An interface of the standard Java library
- This is how we compare two objects (the interface declares the `compareTo()` method)
- Specified by: a.compareTo(b)
- Returns a negative number if a comes before b, 0 if a and b are the same, and a positive number if b comes before a

More Java Collections

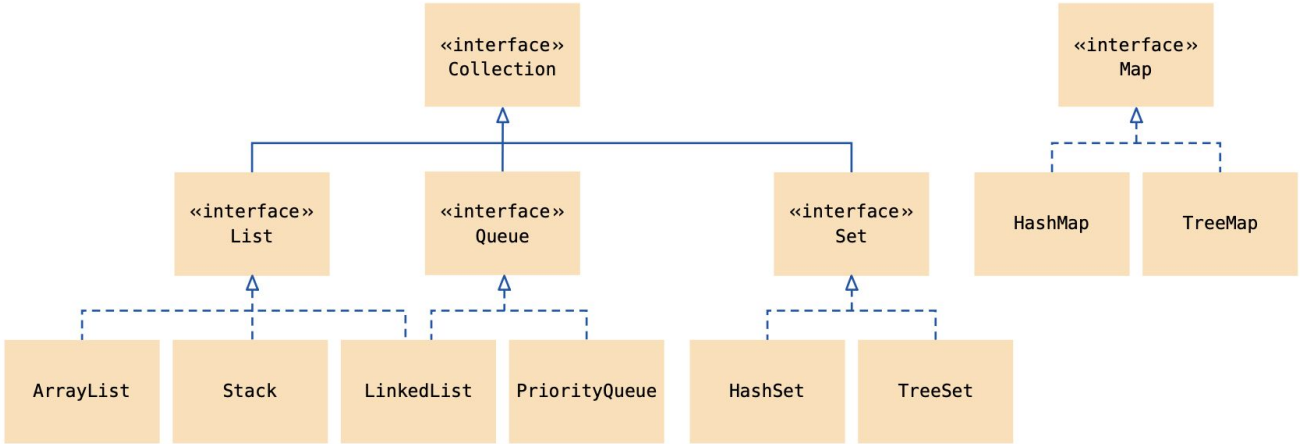

Figure 1 Interfaces and Classes in the Java Collections Framework

Image: Cay Horstmann