Chapter 3 Notes (Wrappers, Static Methods, Strings, Math)

- Static methods can be called without instantiating objects. These are also called class methods. The API has the keyword “static” before the return type.

- One reason a class may define static methods is to provide some quick, one-time functionality without requiring the client to instantiate an object.

-Class or static methods are invoked using the ClassName, rather than an object reference:

ClassName.staticMethodName(argument list);

absValue = Math.abs(some number);

In this above statement, the class name is “Math”, and the static method is “abs”, which returns the absolute value of “some number”, the argument. We use the class name rather than the object name because there is no object and we don’t need objects to call static methods.

- Because static methods can be called without an object being instantiated, static variables cannot access instance variables of the class (because instance variables are object data and exist only once an object has been instantiated). Static methods can access static data, however, and classes often declare static data to be used with static methods. Static data belong to the class, rather than to a particular object, or instance of a class.

- System is an existing Java class. One of its fields is a static constant, “out”, which represents the Java console by default. Another of its fields is a static constant, “in”, which represents the keyboard by default. Because in and out are static, we refer to them by the class name, System, and the dot notation.

- The “exit” method of the System class is useful if we want to stop execution at a place other than the usual end of the program.

- System.out is an object of the PrintStream class, which is also an existing Java class; it can be found in the java.io package. The out object refers to the standard output device, which by default is the Java console.

- The methods print and println belong to the PrintStream class and take arguments of any primitive type, a String, or an object reference. The only difference between print and println is that println will also print a newline character, “\n” after it writes the output.

Text

Description automatically generated

Text, letter

Description automatically generated

- All classes have a toString method, which converts the object data to a String for printing. The toString method is called automatically whenever an object is used as a String. Notice that our SimpleDate class, introduced earlier in the chapter, had a toString method that returned the object data as a String in the format mm/dd/yyyy.

- The toStringmethods API is

String toString ()

- Wrapper classes convert primitive data types into objects.

- The wrapper is important because some programs will only accept objects as arguments. Conversely, there are also times where we will need to convert an object into a primitive data type.

- Java not only converts from primitive data types to objects (autoboxing), but also vice versa (unboxing).

- There are also methods which allow us to convert Strings to numbers: parseInt, parseDouble, and valueOf are methods that are static. They are called using the Integer or Double class name and the dot notation.

- The parse methods convert a String to a primitive type, and the valueOf methods convert a String to a wrapper object.

A close-up of a document

Description automatically generated with medium confidence

A picture containing text, whiteboard

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

A piece of paper with writing

Description automatically generated with medium confidence