Chapter 1: Java Building Blocks.

Object: a runtime instance of a class in memory. All the various objects of all the different classes represent the state of your program.

Elements of a java class: methods and fields. They are the members of a class. Variables hold the state of the program and methods operate on that state.

Keyword: reserved words of the Java language.

Method signature: the full declaration of a method. E.g.: `public int numberVisitors (int month)`

There are two types of comments:

```
// Single-line comment
/*
Multiple-line comment
*/
```

```
/* And
   * // they can be
   * combined
   */
```

Classes and files.

Classes are .java extension. Public classes are not required. You can put multiple classes in one file, at most one of those classes is allowed to be public. The name of the file needs to match the public class.

The main() method is the entry point of the Java program. This is managed by the JVM. The JVM calls the OS to allocate memory and CPU time, access files and so on.

```
public class Zoo {
      public static void main(String[] args) { }
}
```

Public is the access modifier (it can be protected or private)
Static binds the method to its class.
Void is the return type (can be any other type, primitive or user defined). It's a good practice to use the a void method to change an object's state.
Main is the name of the method.
Inside the parenthesis is the argument. Args is the name. String is an array of java.lang.String type. These are acceptable as well for main: String args[] or String… args.

To compile a java file: `javac Zoo.java`. The result will be a file of bytecode by the same name, but with a .class extension.
To run the file: `java Zoo`
To run passing parameters: java Zoo Bronx "San Diego" Zoo

Package declarations and imports.

Java come with thousands of built-in classes which are organized in packages. In order to use them, you would need to import them otherwise an error will be thrown.

There are a few things to keep in mind. Wildcards access all classes inside of a package. Java only looks at classnames in a package, it will not read / import classes contained in other sub-packages. Static imports can import other types.
A classname has priority over wildcard when importing multiple packages and classes.
Java.lang is automatically imported.
Java.nio.file for files and paths.

When the need to import different classes with the same naming convention, it's a good approach to import one of them and use the other's fully qualified name, or the fully qualified name for both.

```
import java.util.Date;
public class Conflicts {
      Date date;
      java.sql.Date sqlDate;
}

public class Conflicts {
      java.util.Date date;
      java.sql.Date sqlDate;
}
```

Importing both classes using the classnames or the wildcard, would be recognized by java as code error and will not compile.

Creating Objects.
To create an instance of a class, the keyword 'new' is used. E.g.: Random r = new Random();
Random() is a constructor. The purpose of the constructor is to initialize fields. It must match the name of the class and there isn't any return type in the method signature.

Order of initialization: fields and instance initializers blocks are run in the order in which they appear in the file. However, the constructor runs after all fields and instance initializers blocks have run.
Instance Initializer example: ***{ System.out.println("setting constructor"); }***

Data types.
Java contains two types of data: primitives and reference types. There are 8 primitives types built in the Java language:

boolean	true or false		true
byte	8-bit integral	2^8 = 2*2=4*28*2= 16*2=32*2=64*2= 128*5=256 This means the range for byte is -128 to 127	123
short	16-bit integral	2^{16}	123
int	32-bit integral	2^{32}	123
long	64-bit integral	2^{64}	3123456789L
float	32-bit floating-point	2^{32}	123.45f
double	64-bit floating-point	2^{64}	123.456
char	16-bit Unicode		'a'

You can now the max value of a primitive type by calling the MAX_VALUE field. E.g.: `System.out.println(Integer.MAX_VALUE);` and it will print ***2,147,438,647***

A number defined in a variable is called a *literal*. E.g.: ***Long max = 3123456789L;***
In literals, you can use underscore for big numbers. This is NOT accepted by Java → ***double a = _1000_._00_;*** They can go anywhere between the numbers in order to make easier the value to read. E.g.: ***double a = 1_000_000.0_0;***

Base 10 numbers – decimal number system (0-9).
Base 8 numbers / octal (0-7), uses 0 as the prefix. E.g.: 017
Base 16 numbers / hexadecimal (0-9, A-F), uses I followed by x or X as a prefix. E.g.: 0xFF
Base 2 numbers / binary (0-1), uses 0 followed by b or B as a prefix. E.g.: 0b10

Reference Types.

These types refer to an object (instance of a class). They hold the address where the object is located in memory – pointer. A value can be assigned in one of two ways: a reference can be assigned to another object of the same type or a new object using the new keyword.

Reference types can be assigned null value, while primitives cannot. Reference types can be used to call methods when they do not point to null. Primitives do not have methods.

Identifiers. There are three rules for identifiers' names:

- The name must begin with a letter or the symbol $ or _.
- Subsequent characters may also be numbers.
- You cannot use a java reserved word for this.

Understanding default initialization of variables.
Local variables are defined within a method. They do not have a default value.
Instance (also called fields) and class variables are not local. Class variables have the *static* keyword. These are not required to initialize them, they get assigned a default value in that case.

boolean → false
byte, short, int, long → 0
float, double → 0.0
char → '\u0000' (NUL)
all objects reference type → null

Variable scope.
Always verify the scope of local, class and instance variables.

- Local variables – in scope from the declaration to the end of the block.
- Instance variables – in scope from the declaration until the object is garbage collected.
- Class variables – in scope from the declaration until the program ends.

Ordering elements in a class.
This the following order for the elements in a class:

Package // may not be required
Import // may not be required – goes immediately after package
Class // required – goes immediately after import
{ fields and methods } // may not be required – anywhere inside the class
// comments – anywhere in the file

Garbage collection.
All java objects are stored in the program memory's heap. The heap (free store) is a large pool of unused memory allocated to your java app. The Garbage Collector deletes the objects from memory that are no longer reachable by the app.

System.gc(); → a request for Garbage Collector to run, but this request can be ignored by Java.
An object is not reachable when:

1) No reference points to it.
2) The reference to the object goes out of scope.

Finalize() can be implemented but is only run when the object is eligible for the garbage collector. The method can run zero or one time. If garbage collector fails to collect the object and runs a second time, finilaize() won't be called again.

Benefits of java.

- Object oriented. Organized in classes.
- Encapsulation. Supports access modifiers to protect data from unintended access and modifications.
- Platform independent. Java code compiles to bytecode. Can be read by the JVM on any OS.
- Robust. Prevents memory leaks.
- Simple. No pointers, no operator overloading.
- Secure. Java code runs in the JVM, which creates a sandbox to execute the code.