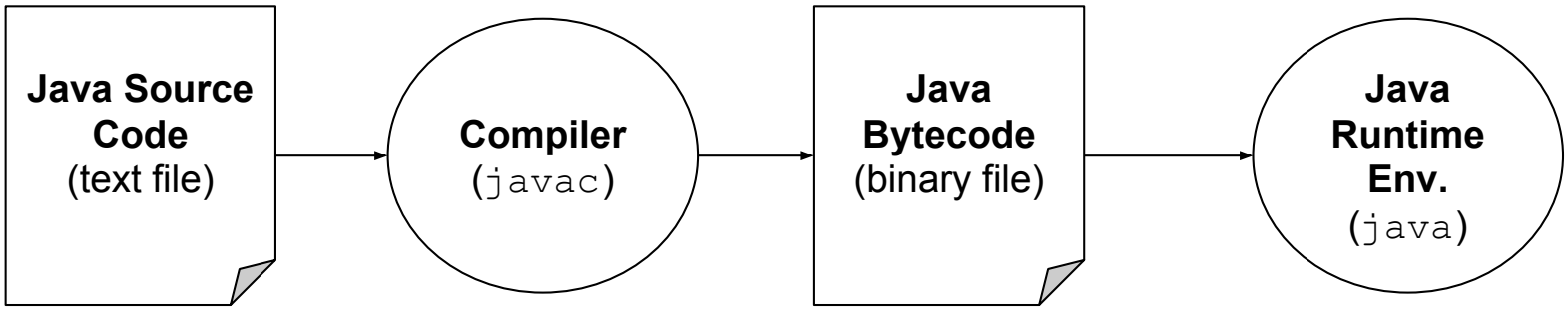
Lecture 31: From Python to Java

Java is Compiled



Java is (in many senses) a compiled langauge.

- Code you write is translated to bytecode
- Bytecode is run in a Java Virtual Machine
- There is no REPL):

Let's Run a Java Program

```
$ atom HelloWorld.java
$ ls
HelloWorld.java
$ javac HelloWorld.java
$ ls
HelloWorld.class  HelloWorld.java
$ java HelloWorld
Hello World!
```

Builtin Java Types

Java has builtin types that are similar to builtin Python types

- `byte`: 8-bit integer
- `short`: 16-bit integer
- `int`: 32-bit integer
- `long`: 64-bit integer
- `float`: 32-bit real number
- `double`: 64-bit real number
- `boolean`: `false` and `true`
- `char`: single character (declared with single quotes)

Builtin Java Types

Java has builtin types that are similar to builtin Python types

- `byte`: 8-bit integer
- `short`: 16-bit integer
- `int`: 32-bit integer
- `long`: 64-bit integer
- `float`: 32-bit real number
- `double`: 64-bit real number
- `boolean`: `false` and `true`
- `char`: single character (declared with single quotes)

Built-in numeric Java types have a fixed size.

- We must know the largest value that we plan to represent

Static Typing

The compiler checks type compatibility for us before we run our code.

- We must *declare* the type of every variable *before* assigning it a value, and
- once we declare a variable's type, its type cannot change.

```
1    int a;
2    String s = "text";
3    Vector<Float> v;
4    a = 10
5    byte b = (byte) a;
```

Booleans and Conditionals

- if/elif/else → if/else if/else
- Boolean expression *must* be enclosed in parenteses
- Use braces instead of colon to define scope

```
1  if (a) { // check if a == true
2      i = i + 1;
3      System.out.prinln("This multiline block required braces");
4  } else if (i < j) {
5      i = 0;
6  } else {
7      return i;
8  }
```

For Loops

There are multiple ways to define a for loop:

- for (initialization; stopping condition; update)
- for (type element : iterable_collection)

```
1  Vector<Integer> v = new Vector<Integer>();
2
3  for (int i = 0; i < 10; i++) {
4      v.add(i);
5  }
6
7  for (int i : v) {
8      System.out.println(i);
9  }
```

An Example Class

```
1  class ExampleClass {
2      private int a;
3      private int b;
4
5      public ExampleClass(int a, int b) {
6          this.a = a;
7          this.b = b;
8      }
9
10     public static int addNums(int a, int b) {
11         return a + b;
12     }
13
14     public int addNums() {
15         return a + b;
16     }
17
18     public static void main(String[] args) {
19         ExampleClass ex = new ExampleClass(3, 4);
20         System.out.println(ExampleClass.addNums(1, 2));
21         System.out.println(ex.addNums());
22     }
23 }
```

Practice

Write Java functions to complete the following tasks:

- Print all odd integers between `rangeMin` and `rangeMax`
- Sum all elements in Vector v
- Return a reversed copy of Vector v.

Practice

Write Java functions to complete the following tasks:

- Print all odd integers between `rangeMin` and `rangeMax`
- Sum all elements in Vector v
- Return a reversed copy of Vector v.

```
1   class Practice {
2
3       /* print all odd integers between rangeMin and rangeMax */
4       public static int printOdds(int rangeMin, int rangeMax) {}
5
6       /* sums all elements in v */
7       public static int sumVector(Vector<Integer> v) {}
8
9       /* returns copy of v, in reverse */
10      public static Vector<Integer> reverseVector(Vector<Integer> v) {}
11  }
```