**Booleans and Data Representation**

Types in programming language:

* specify length of data (how many bits)
* help prevent errors
* enforce program invariants

In assembly and C:

* possible to case integer into a pointer
* enables program to write to random memory locations

Types in JVM

* as in (early version of) Java (sometimes simpler types)
* goal: JVM byte code verifier ensures memory safety
  + cannot overwrite memory locations
  + can avoid doing certain checks while running JVM

One difference Java vs Byte codes:

* java treats boolean as int
* they are implicitly converted into integers [primitive conversions](http://java.sun.com/docs/books/jvms/second_edition/html/Concepts.doc.html#23435)
* and they are implicitly converted back from integers :)

Little support for boolean operations

* intermediate boolean values are often invisible (encoded in program points)
* can manipulate them as integers

**Booleans in Bytecode are Fiction**

Using [cafebabe](http://lara.epfl.ch/w/compilation:cafebabe), our wonderful bytecode generation library, we generate this class:

public boolean test(int, boolean);

Code:

0: iload\_1

1: iload\_2

2: bipush 37

4: iadd

5: iadd

6: ireturn

public static void main(java.lang.String[]);

Code:

0: getstatic #19; //Field java/lang/System.out:Ljava/io/PrintStream;

3: new #3; //class HW

6: dup

7: invokespecial #20; //Method "<init>":()V

\* 10: iconst\_3

\* 11: iconst\_2

\* 12: invokevirtual #22; //Method test:(IZ)Z

15: invokevirtual #28; //Method java/io/PrintStream.println:(I)V

18: return

}

Using

java HW

in Java HotSpot(TM) Server VM (build 1.6.0\_03-b05, mixed mode), this:

* successfully passes bytecode verifier
* happily returns ‘the’ answer

**Convention on Booleans**

We adopt the following convention in code generation for JVM

The generated code will use ‘int’ to represent boolean values in

* local variables
* parameters
* intermediate stack values (if needed–often not needed)

In such cases, the ‘int’ variables will always be either

* 0, representing false
* 1, representing true

Other conventions are possible