COMP 1010- Summer 2015 (A01)

Jim (James) Young

young@cs.umanitoba.ca

jimyoung.ca

Hello!

James (Jim) Young
young@cs.umanitoba.ca
jimyoung.ca

office hours T / Th: 17:00 – 18:00
EITC-E2-582
(or by appointment, arrange by email)

Lab 6 is up

AS1 solution is up

Tuesday – midterm review

- Email me things you want me to go over
- Mid term is up to and including for loops (and nested for loops)

Casting

conversion between types (casting)

we have int, long, float, double, etc., how do they relate? how do we go between them?

Try ...

```
int i = 1234;
byte b = i;
```

What will happen?
1234 cannot fit into byte?

What about..

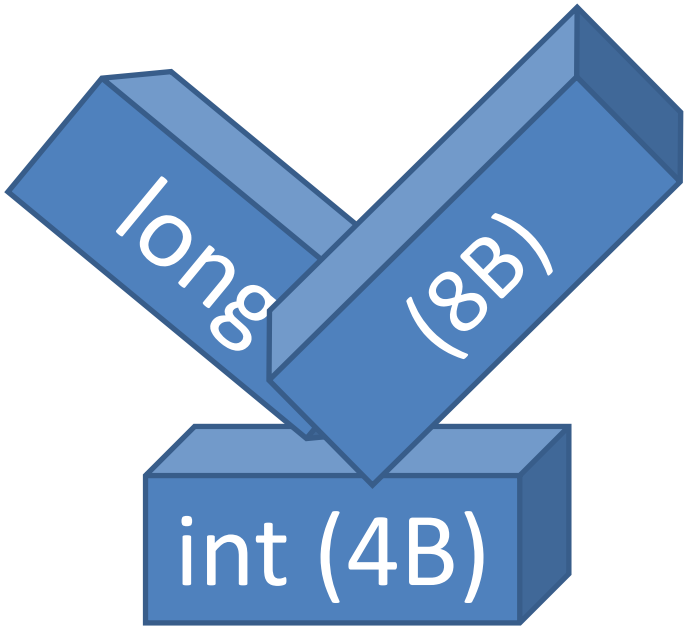
```
long l = 1234;
int i = l;
```

It just doesn’t fit!!

Processing knows that the int only has half the memory. It doesn’t even try

It’s dangerous!

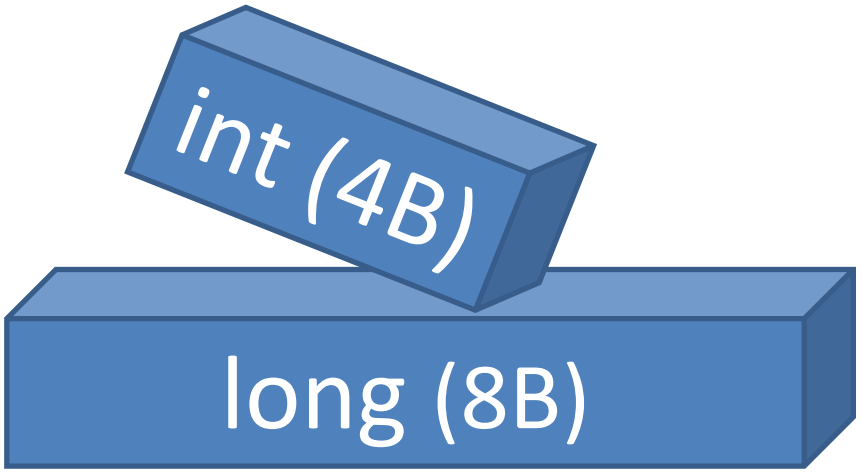
Narrowing conversion



Other direction

```
int small;
long large;
small = 15;
large = small; // Convert an int to a long!!
```

Widening conversion



casts

widening conversions automatically convert (cast) the data types – this is called an **implicit cast**

narrowing can result in the loss of data, so Processing requires that you **explicitly cast** the data to the new type

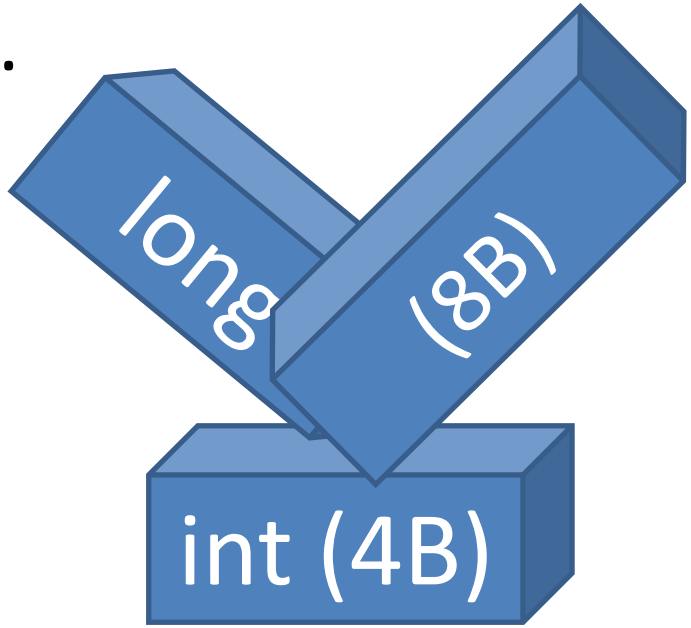
Example:

long large = 200;

int small = large;

Error: cannot convert

Processing is saying that you may lose data, so it doesn't want to do it.



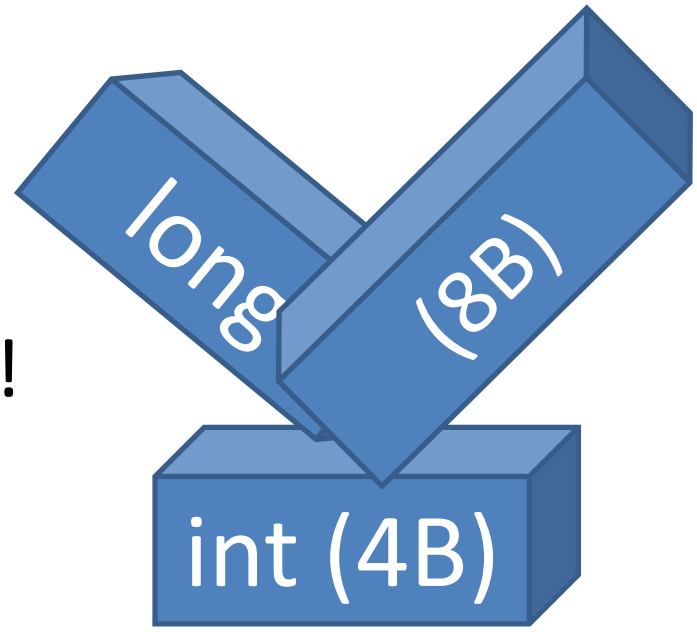
Explicit cast:

put (type) in front of a variable or value

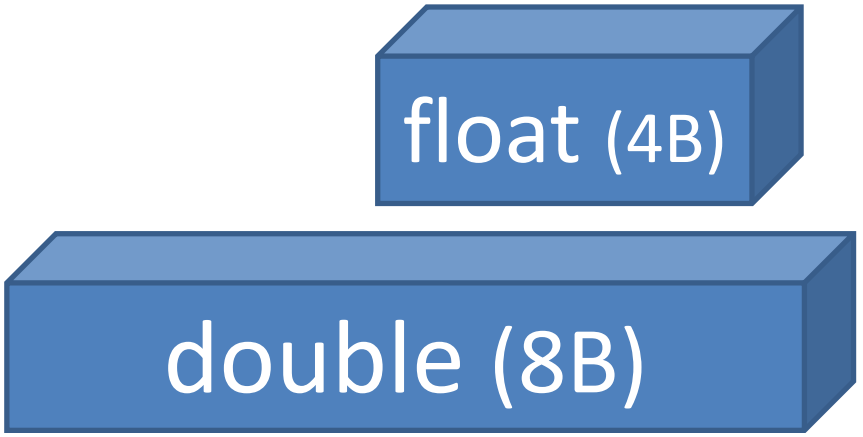
```
long large = 200;
int small = (int)large;
```

This tells Processing to convert

Just do it! I know what I'm doing!



floating point..



float is 4 bytes

double is 8 bytes

widening conversion is an **implicit cast**

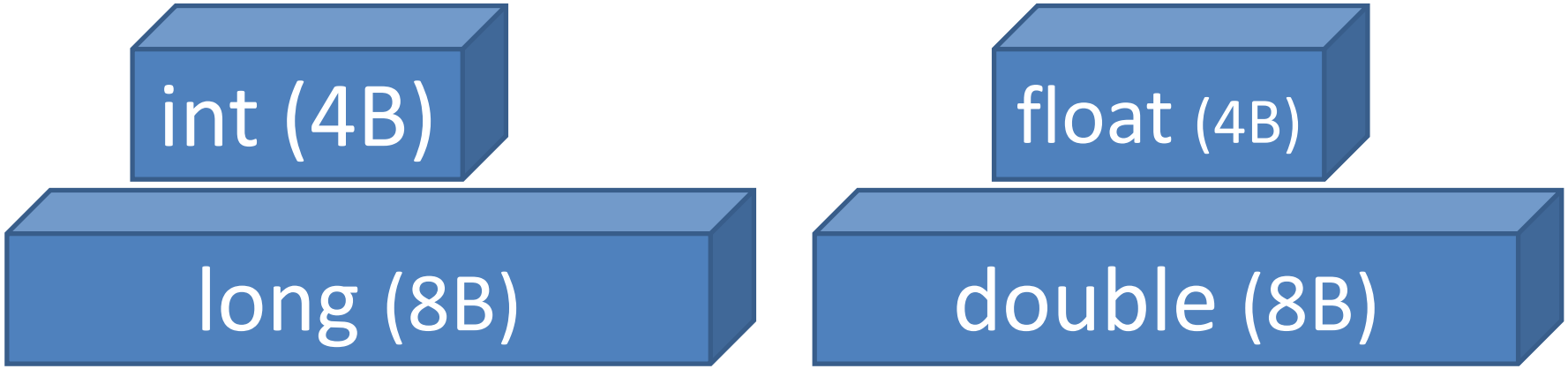
narrowing conversion requires an **explicit cast:**

```
float f = 1.234;
double d = f;
float floatVariable = (float)doubleVariable;
```

converting between integer and floating point types

floating point -> integer, data is lost so an **explicit cast** is needed to make Java happy.

integer -> floating point, **implicit cast** because floating point is more capable and can accommodate the integer.



Float -> integer

When explicitly cast to an integer, a floating point number gets **truncated.** The decimal portion is lost.

```
float f = 123.456;
int i = (int)f;
println(i);
```

order of operations with the explicit cast!

int integerVariable = (int)0.5*3.0;

Casts happen first!

the cast converts the 0.5 to an int first, = 0

second, the multiplication takes place: 0 * 3.0

the result is a floating point.

To fix this?

int integerVariable = (int)(0.5*3.0);

What we learnt…

Data types have a fixed amount of memory, which dictates how much information they can store

Different datatypes store information differently, e.g., floating point versus integer

you can convert between data types (floating point or integer) and between memory sizes (e.g., long<->int, float<->double) using casts.

Compiling and the Java Virtual Machine!!

what does this mean, anyway??

Processing is basically Java

computers can only understand binary!!

binary is a counting system that only has 0s and 1s.

“It was just a dream Bender, there’s no such thing as two.”

computers cannot understand programing languages like Processing!

programming languages are designed for people

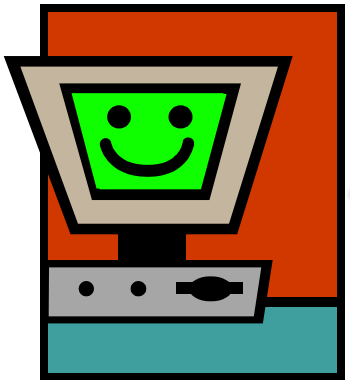
human-readable computer code

```
float left = 100;
float top = 100;
float dotSize =
50;

void setup()
{
size(canvasSize,ca
nvasSize);
}
```

compiler

a compiler program can read human-readable code, and translate to **machine language**



computers do not know how to human

machine language

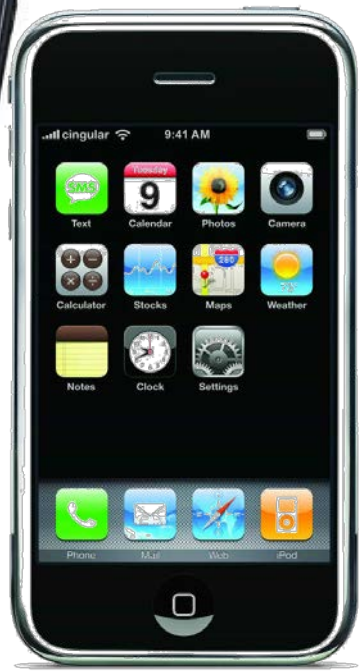
```
10101001101010
01010100100010
00101011101001
10101010001001
10101001010100
10001000101011
10100110101010
10101010010101
```

compilers are necessary

note: a program **must** be **compiled** before it can be run by a computer. When you buy software or download a program, it is usually already compiled and packaged to run.

complication…

different computers speak different languages..



there are **many** machine languages

```
float left = 100;
float top = 100;
float dotSize =
50;

void setup()
{
size(canvasSize,ca
nvasSize);
}
```

human-readable computer code

iPhone compiler

mac compiler

PC compiler

iPhone machine language

mac machine language

PC machine language

Not scalable

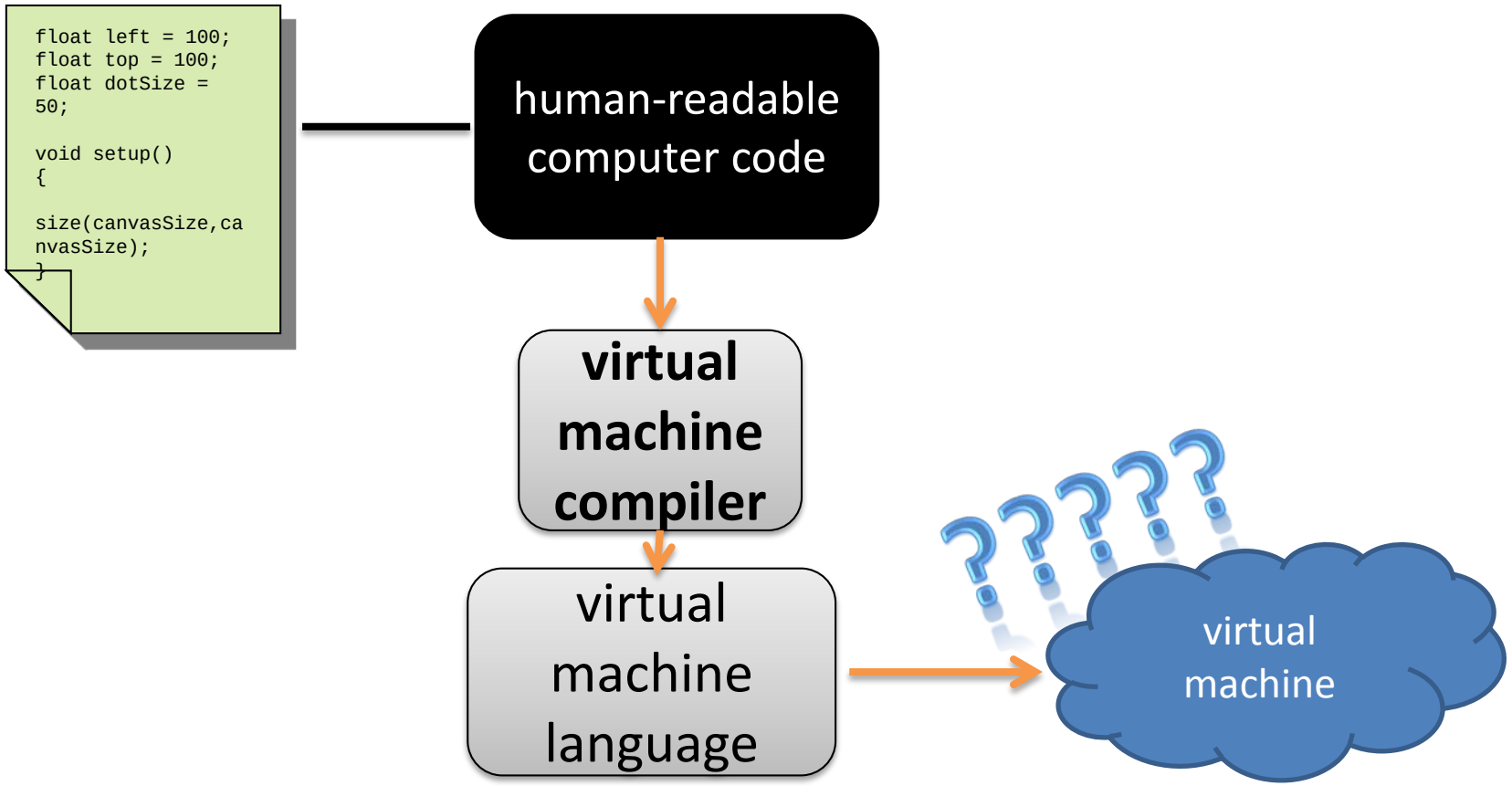
what if a new platform is introduced??

for new platforms, you need to make a new compiler to convert human-readable code to machine code

EVERY program must be re-compiled, debugged, updated, to make it work

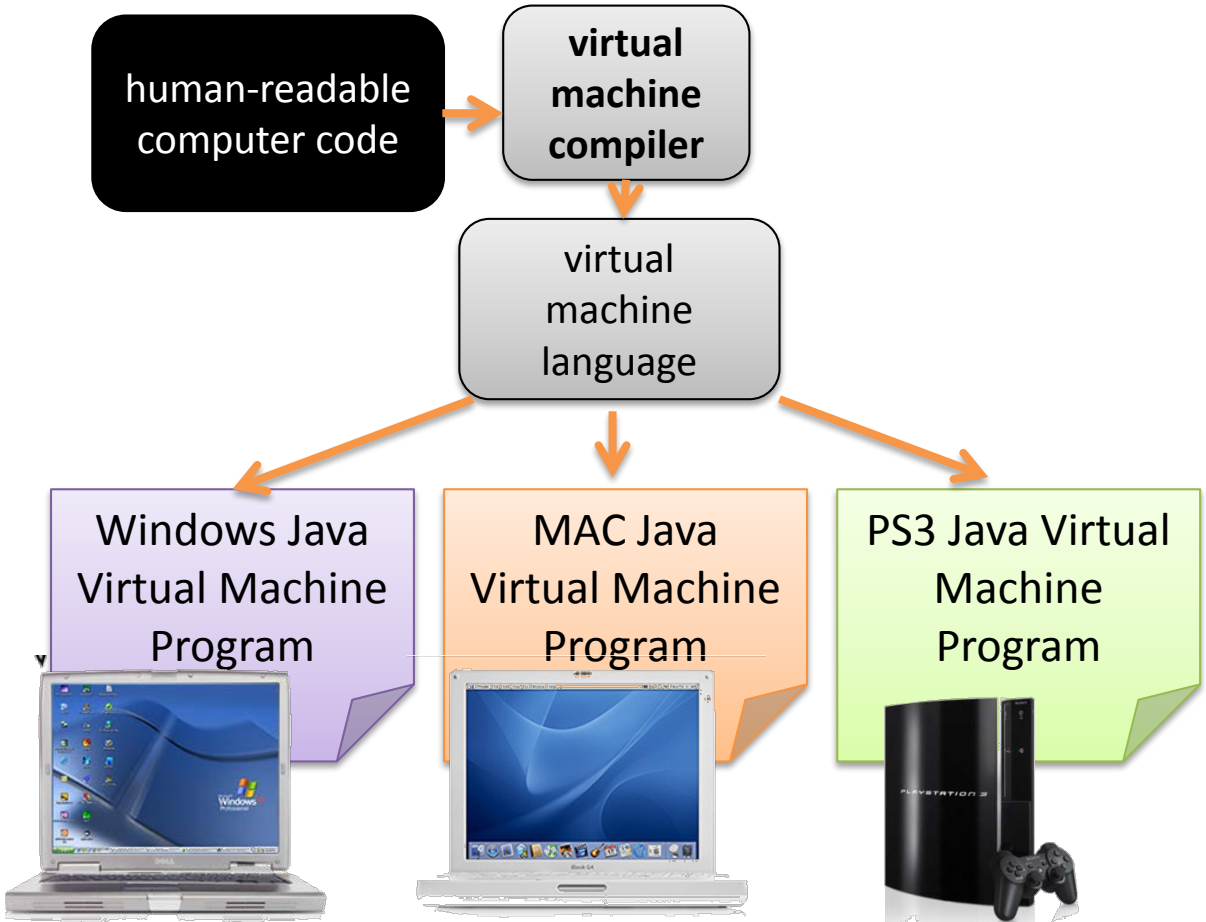
solution: a **virtual machine**

rather than compiling a program to run on a specific machine, we compile a program to run on some imaginary **virtual machine**.



JAVA has **virtual machine** programs, or emulators, for many platforms!

A **virtual machine** program can read and execute (run) **virtual machine code**



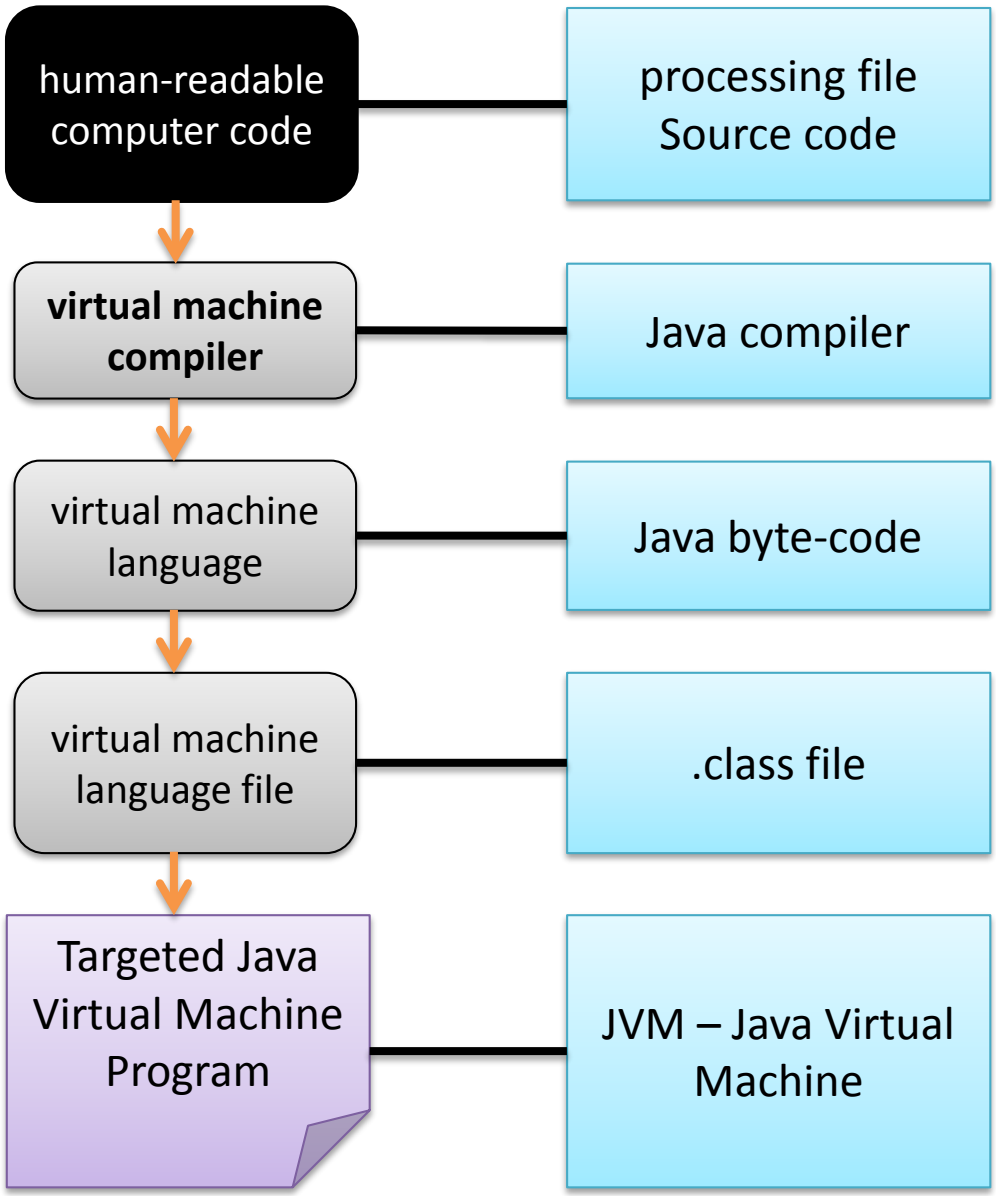
scalable!

what if a new platform is introduced??

for new platforms, you need to make a new Java virtual machine

Then, all your existing Java programs will run!! no need to recompile them!

terminology and convention



store a program written in the programming language in this file

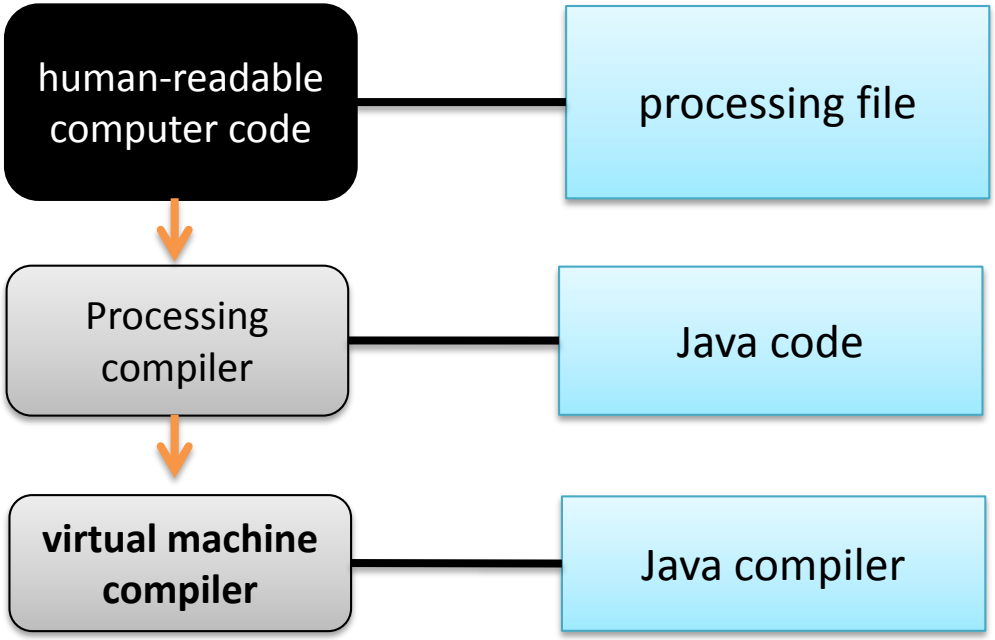
many versions! Sun (inventors), Eclipse compiler, others…

machine instructions for the virtual machine – not human readable

produced by the Java compiler, not human readable. Gets run on the virtual machine

specific to your platform (windows, etc..). Many versions available

Where does Processing fit in???



summary

programming languages are designed for humans – computers cannot understand them

a **compiler** converts human-readable programming into platform-specific **machine language**

the **Processing compiler** converts your program into Java

the **Java compiler** converts a Java program into **Java byte code-** the machine language for the **Java Virtual Machine (JVM)**

the **Java byte code** can be run on any **JVM –** these are available for many computers / platforms.

things to do!

just understand the basic concepts of the JVM and what compiling is