# Java

* src > main > java contains package folders. A package is a namespace.
* Each .java file must have a class that has the same name.
  + This class must have a method ‘main’ that is run by the compiler
* Java is strongly typed language, i.e. every variable must explicitly have a type.
* Strings have double quotes “string”. Strings are concatenated with +, i.e. “Hello world” is equal to “Hello ” + “World”
* Methods for every string are called via .’…’. These can be chained
* ctrl + space brings up possible methods. ctrl + click goes to the source code.
* Everything in Java is an object.
* A class file is ‘the blueprint’. An object is an instance of a class.
* All java > lang packages are imported automatically. Others need to be imported manually.
* Static classes can ‘survive’ without an instantiation. Anything called within a static class must also be static.
* Primitive types are the only ones that aren’t classes. These are 1 byte size large. Difference between Double and double.
* Difference to C/C++
  + Java has garbage collection, i.e. it automatically assigns and de-assigns memory. C++ doesn’t.
  + Java is an interpreted language, C/C++ is a compiled language.
* Comments
  + /\*\* …. \*/ is a class comment. This shows up when ‘hovering’ over a class or its method
  + /\* … \*/ is a multiline comment
  + // is a single line comment
* We use the ‘this’ to prevent ‘clashes’ between a local variable and a class ‘property’
* There is a difference between the Java runtime environment, JRE, needed to run programs, and Java development kit, JDK, which you need for development.
* A constructor, a method constructing the object defined by the class, is a method with the same name as the class.
* We can define data structures, like:
  + Lists, which stores objects in a number. A downside is that to find a specific item within a list, you need to access all of them
  + Maps (Hash/Dictionary), which has a constant access time. It works with a key and
* A ‘final’ variable cannot be overwritten, but can be modified / accessed.
* An abstract class creates a class that has some methods that are not yet implemented.
* A .jar file is a way of ‘compiling’ programs to be able to run them outside of an IDE. Can also create .exe files (for Windows).
* Interfaces dictate a number of methods an implementation of that interface should have. Then, any class can implement the methods specified by the interface. The