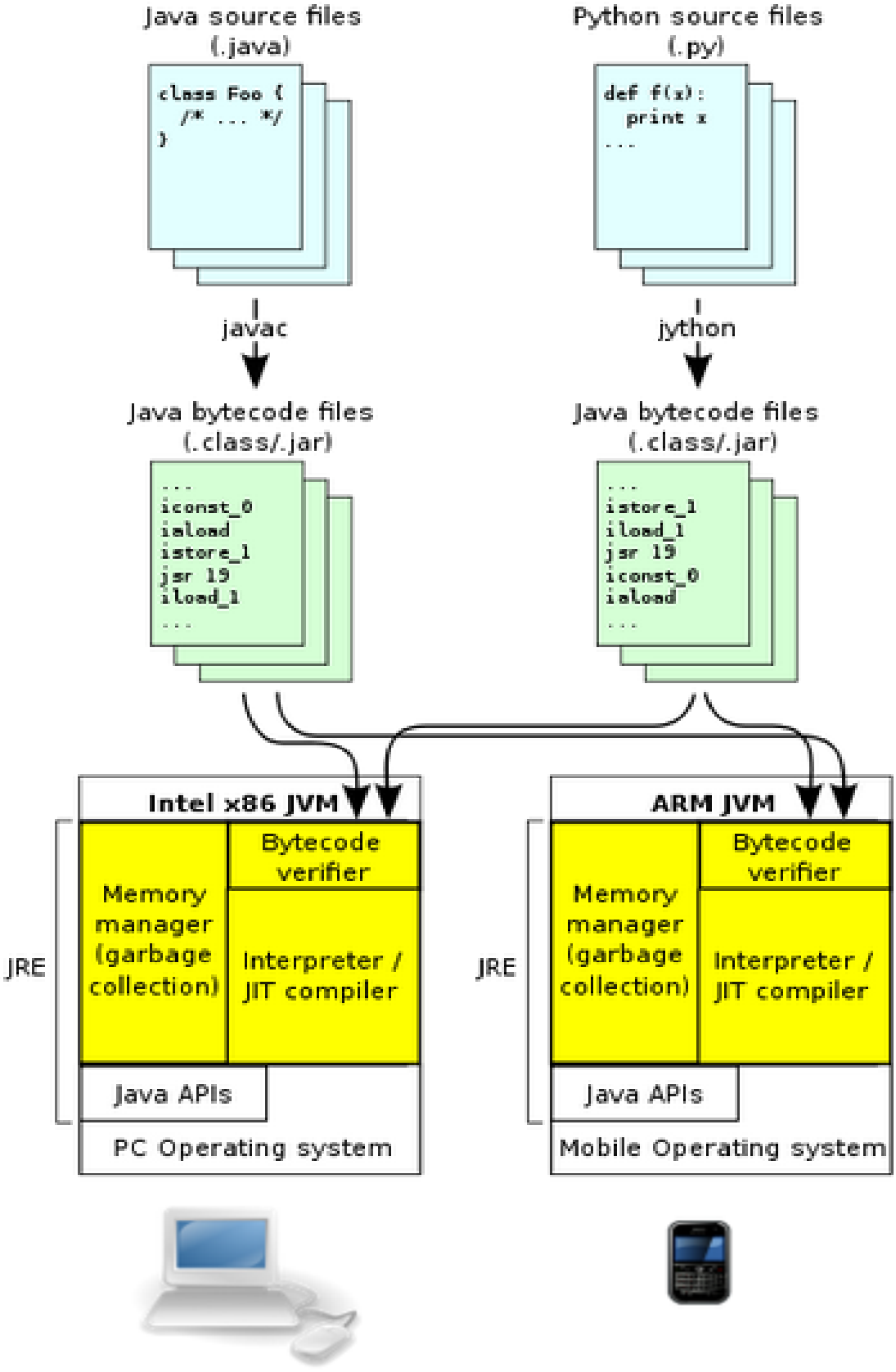

Overview of a Java virtual machine (JVM) architecture. Source code is compiled to Java bytecode, which is verified, interpreted or JIT-compiled for the native architecture. The Java APIs and JVM together make up the Java Runtime Environment (JRE).

Overview of JVM:

A Java virtual machine (JVM) interprets compiled Java binary code (called bytecode) for a computer's processor (or "hardware platform") so that it can perform a Java program's instructions.

JVM languages

Although the JVM was primarily aimed at running compiled Java programs, many other languages can now run on top of it.

Versions of non-JVM languages

Language	On JVM
Erlang	Erjang
JavaScript	Rhino
Pascal	Free Pascal
PHP	Quercus
Python	Jython
REXX	NetRexx[3]
Ruby	JRuby
Tcl	Jacl

Languages designed expressly for JVM

Language
BBj
Clojure
Fantom
Groovy
MIDletPascal
Scala
Kawa

Example: **PHP programs are to be converted to Quercus if they have to be run on JVM.**

Bytecode verifier

A basic philosophy of Java is that it is inherently safe from the standpoint that no user program can crash the host machine or otherwise interfere inappropriately with other operations on the host machine.

To provide this safety , the JVM verifies all bytecode before it is executed. This verification consists primarily of three types of checks:

1. Branches are always to valid locations.
2. Data is always initialized and references are always type-safe.
3. Access to private or package private data and methods is rigidly controlled.

JVM heap

The Java virtual machine heap is the area of memory used by the JVM, for dynamic memory allocation. The heap is divided into generations:

1. The young generation stores short-lived objects that are created and immediately garbage collected.
2. Objects that persist longer are moved to the old generation (also called the tenured generation).