**Compilation**

First, the source ‘.java’ file is passed through the compiler, which then encodes the source code into a machine independent encoding, known as Bytecode. The content of each class contained in the source file is stored in a separate ‘.class’ file. While converting the source code into the bytecode, the compiler follows the following steps:

* **Parse**: Reads a set of \*.java source files and maps the resulting token sequence into AST (Abstract Syntax Tree)-Nodes.
* **Enter**: Enters symbols for the definitions into the symbol table.
* **Process annotations**: If Requested, processes annotations found in the specified compilation units.
* **Attribute**: Attributes the Syntax trees. This step includes name resolution, type checking and constant folding.
* **Flow**: Performs dataflow analysis on the trees from the previous step. This includes checks for assignments and reachability.
* **Desugar**: Rewrites the AST and translates away some syntactic sugar.
* **Generate**: Generates ‘.Class’ files.

Precompile

Some compilers have a precompile or pre-processor to process statements that are not part of the computer programming language. If your source program contains EXEC CICS® statements or EXEC SQL statements, then it must first be pre-processed to convert these statements into COBOL, PL/I or Assembler language statements, depending on the language in which your program is written.