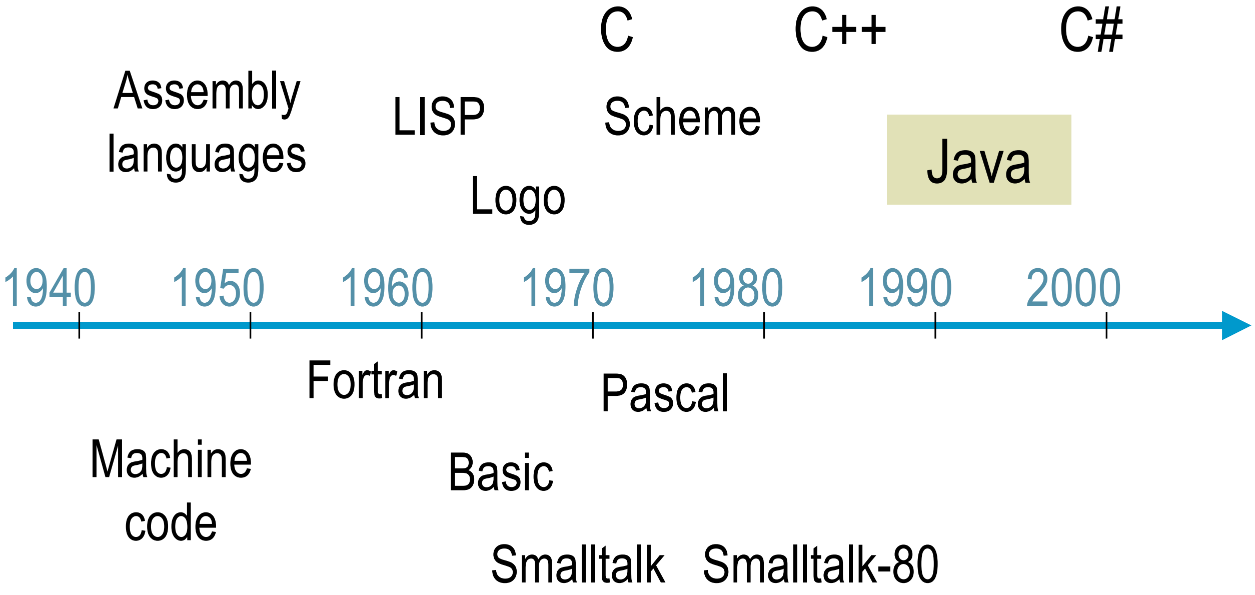
Step Into Java: Some Basics

Mr. Neat

Java

Programming Languages



C C++ C#

Assembly languages

LISP

Scheme

Java

Logo

1940 1950 1960 1970 1980 1990 2000

Fortran

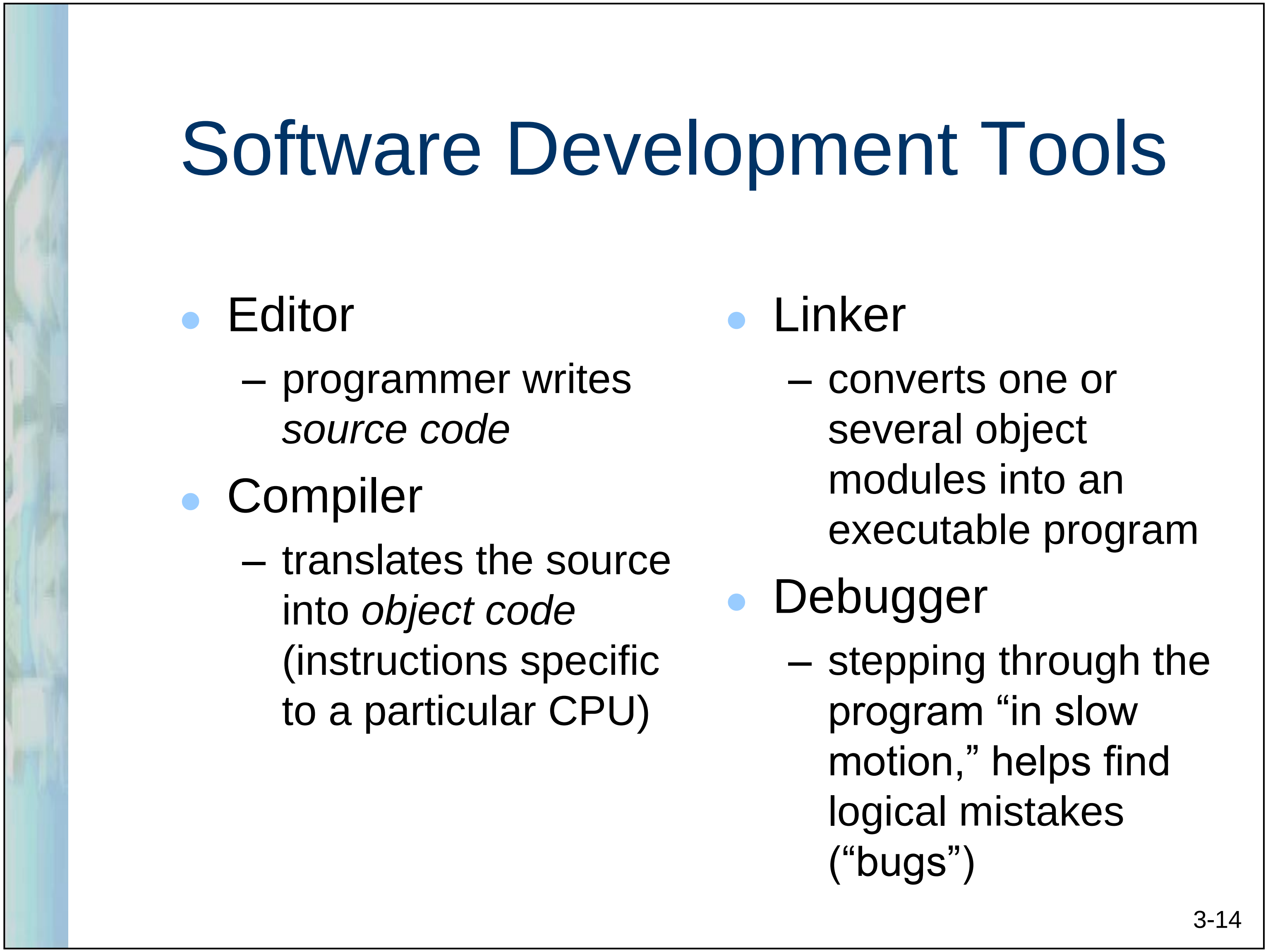
Pascal

Machine code

Basic

Smalltalk Smalltalk-80

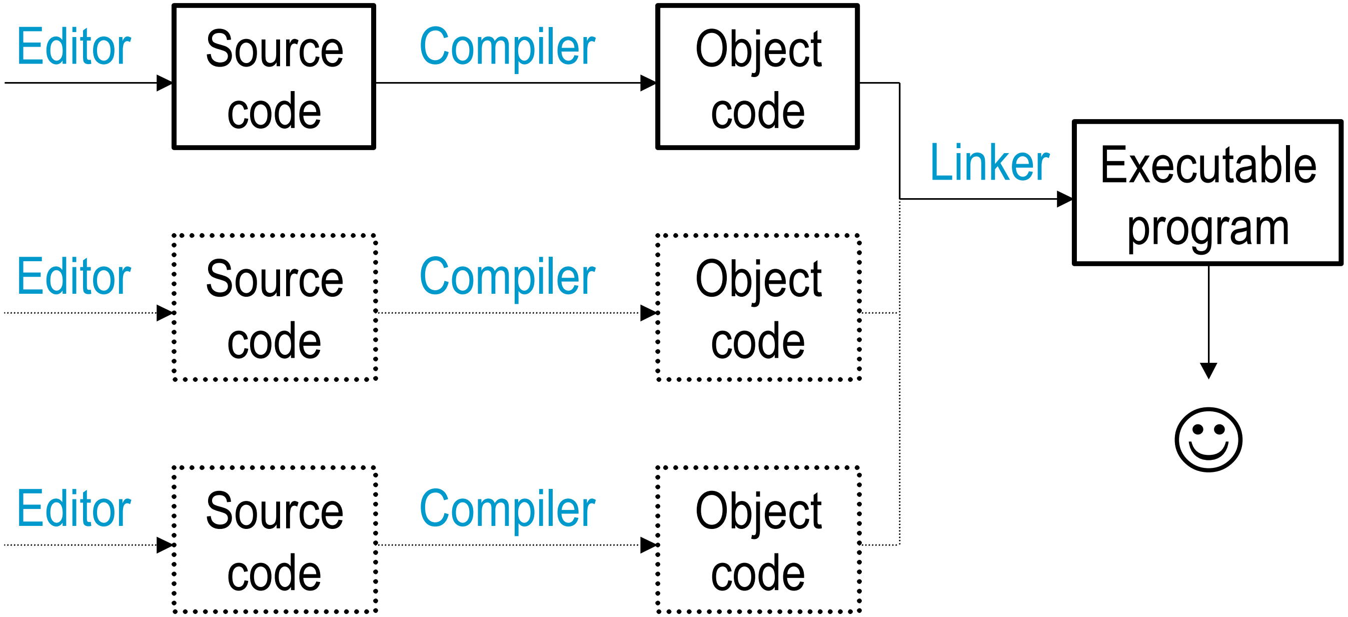
Compiled Languages



Software Development Tools

- Editor
  - – programmer writes *source code*
- Compiler
  - – translates the source into *object code* (instructions specific to a particular CPU)
- Linker
  - – converts one or several object modules into an executable program
- Debugger
  - – stepping through the program “in slow motion,” helps find logical mistakes (“bugs”)

Compiled Languages: Edit-Compile-Link-Run



Editor

Source code

Compiler

Object code

Editor

Source code

Compiler

Object code

Editor

Source code

Compiler

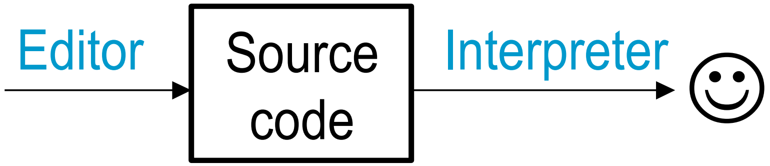
Object code

Linker

Executable program

☺

Compiler vs. Interpreter

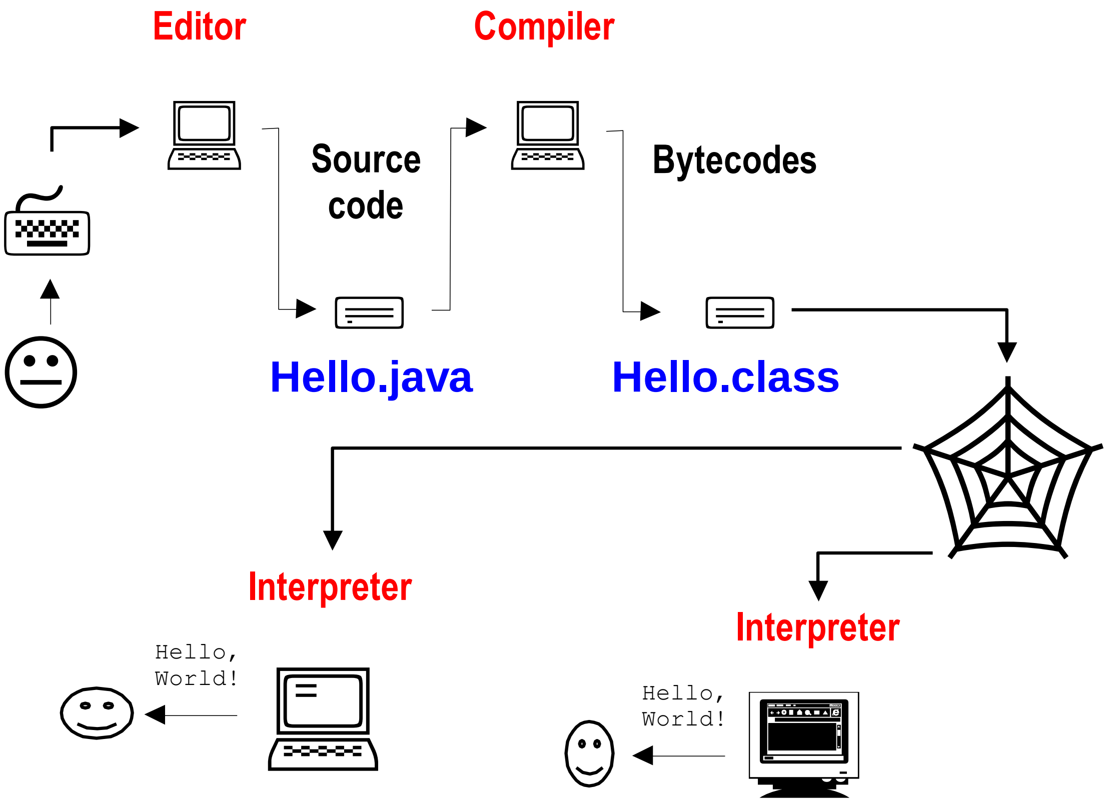


Compiler:		Interpreter:
checks syntax	⟷	checks syntax
generates machine-code instructions	⟷	executes appropriate instructions while interpreting the program statements
not needed to run the executable program	⟷	must remain installed while the program is interpreted
the executable runs faster	⟷	the interpreted program is slower

Java’s Hybrid Approach: Compiler + Interpreter

- A Java compiler converts Java source code into instructions for the *Java Virtual Machine.*
- These instructions, called *bytecodes*, are the same for any computer / operating system.
- A Java interpreter executes bytecodes on a particular computer.

Java’s Compiler + Interpreter



Why Bytecodes?

- Platform-independent.
- Load from the Internet faster than source code.
- Interpreter is faster and smaller than it would be for Java source.
- Source code is not revealed to end users.
- Interpreter performs additional security checks, screens out malicious code.