CHAPTER 1: THE HISTORY AND EVOLUTION OF JAVA

OUTLINE:

INTRODUCTION

Computer language innovation and development occurs for two fundamental reasons:

1) to adapt to changing environments and uses

2) To implement refinements and improvements in the art of programming

The development of Java was driven by both in equal measures.

Many Java features are inherited from the earlier languages:

C → C++ → Java

BEFORE JAVA: C

Designed by Dennis Ritchie in 1970s.

Before C: BASIC, COBOL, FORTRAN, PASCAL

C- structured, efficient, high-level language that could replace assembly code when creating systems programs.

Designed, implemented and tested by programmers.

BEFORE JAVA: C++

Designed in 1979 initially called "C with Classes".

Response to the ***increased complexity of programs*** and respective improvements in the programming paradigms and methods:

1) assembler languages
2) high-level languages
3) structured programming
4) object-oriented programming (OOP)

OOP – methodology that helps organize complex programs through the use of inheritance, encapsulation and polymorphism.

C++ extends C by adding object-oriented features.

JAVA: HISTORY

In 1990, Sun Microsystems started a project called Green.

Objective: to develop software for consumer electronics.

Project was assigned to James Gosling, a veteran of classic network software design. Others included Patrick Naughton, ChrisWarth, Ed Frank, and Mike Sheridan.

The team started writing programs in C++ for embedding into

- toasters
- washing machines
- VCR’s

Aim was to make these appliances more “intelligent”.

JAVA: HISTORY (CONTD.)

C++ is powerful, but also dangerous. The power and popularity of C derived from the extensive use of pointers. However, any incorrect use of pointers can cause memory leaks, leading the program to crash.

In a complex program, such memory leaks are often hard to detect.

Robustness is essential. Users have come to expect that Windows may crash or that a program running under Windows may crash. ("This program has performed an illegal operation and will be shut down").

In computer science, robustness is the ability of a computer system to cope with errors during execution and cope with erroneous input.

JAVA: HISTORY (CONTD.)

However, users do not expect toasters to crash, or washing machines to crash.

A design for consumer electronics has to be *robust.*

Replacing pointers by references, and automating memory management was the proposed solution.

JAVA: HISTORY (CONTD.)

Hence, the team built a new programming language called Oak, which avoided potentially dangerous constructs in C++ that can avoid memory leaks.

Introduced automatic memory management, freeing the programmer to concentrate on other things.

Architecture neutrality (Platform independence)

JAVA: HISTORY (CONTD.)

Many different CPU's are used as controllers. Hardware chips are evolving rapidly.

As better chips become available, older chips become obsolete and their production is stopped.

Manufacturers of toasters and washing machines would like to use the chips available off the shelf, and would not like to reinvest in compiler development every two-three years.

So, the software and programming language had to be *architecture neutral*.

JAVA: HISTORY (CONTD)

It was soon realized that these design goals of consumer electronics perfectly suited an ideal programming language for the Internet and WWW, which should be:

- object-oriented (& support GUI)
- robust
- architecture neutral

Internet programming presented a BIG business opportunity. Much bigger than programming for consumer electronics.

Java was “re-targeted” for the Internet

JAVA: HISTORY (CONTD)

The team was expanded to include Bill Joy (developer of Unix), Arthur van Hoff, Jonathan Payne, Frank Yellin, Tim Lindholm etc.

In 1994, an early web browser called WebRunner was written in Oak. WebRunner was later renamed HotJava.

In 1995, Oak was renamed Java.

A common story is that the name Java relates to the place from where the development team got its coffee.

JAVA HISTORY (CONTD)

Designed by James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan at Sun Microsystems in 1991.

The original motivation was not Internet:

platform-independent software embedded in consumer electronics devices was original motto.

JAVA HISTORY (CONTD)

With Internet, the urgent need appeared to break the fortified positions of Intel, Macintosh and Unix programmer communities.

Java as an “Internet version of C++”? No.

Java was not designed to replace C++, but to solve a different set of problems.

HOW JAVA CHANGED THE INTERNET

- Java Applets

- Security
- Portability

JAVA BYTECODE

- **Bytecode** is the compiled format for **Java** programs.
- ***Bytecode is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM).***
- Once a **Java** program has been converted to **bytecode**, it can be transferred across a network and executed by **Java** Virtual Machine (JVM).
- **Bytecode** files generally have a .class extension.
- JVM gives Security and Portability
- Bytecode has been highly optimized → ?
- Interpreted not compile → Just-In-Time (JIT) compiler

THE JAVA BUZZWORDS

The key considerations were summed up by the Java team in the following list of buzzwords:

- Simple
- Object-oriented
- Robust
- Secure
- Portable
- Multithreaded
- Architecture-neutral
- Interpreted
- High performance
- Distributed
- Dynamic

Simple – Java is designed to be easy for the professional programmer to learn and use.

Object-oriented: a clean, usable, pragmatic approach to objects, not restricted by the need for compatibility with other languages.

Robust: restricts the programmer to find the mistakes early, performs compile-time (strong typing) and run-time (exception-handling) checks, manages memory automatically.

Multithreaded: supports multi-threaded programming for writing program that perform concurrent computations

Architecture-neutral: Java Virtual Machine provides a platform independent environment for the execution of Java byte code

Interpreted and high-performance: Java programs are compiled into an intermediate representation – byte code:

a) can be later interpreted by any JVM

b) can be also translated into the native machine code for efficiency.

Distributed:

- Java is designed for the distributed environment of the Internet because it handles TCP/IP protocols. In fact, accessing a resource using a URL is not much different from accessing a file.
- Java also supports Remote Method Invocation (RMI). This feature enables a program to invoke methods across a network.

Dynamic: Java programs carry with them substantial amounts of run-time type information to verify and resolve access to objects at run-time.

THE EVOLUTION OF JAVA

- Java 1.0
- Java 1.1
- Java 2 → version 1.2 → “2” indicates “second generation.”
  - With Java 2, Sun repackaged the Java product as J2SE
  - Added new features, such as Swing and the Collections Framework
- J2SE 1.3
- J2SE 1.4
- J2SE 5 → major new features
- JDK 5 (developer version number) and J2SE5 (*product version* number)
- J2SE 6
- Java SE 7 (July 28, 2011)
- Java SE 8 (March 18, 2014)
- Java SE 9 (September 21, 2017)
- Java SE 10 (March, 20, 2018)

THE EVOLUTION OF JAVA

1. JDK Alpha and Beta (1995)
2. JDK 1.0 (23rd Jan 1996)
3. JDK 1.1 (19th Feb 1997)
4. J2SE 1.2 (8th Dec 1998)
5. J2SE 1.3 (8th May 2000)
6. J2SE 1.4 (6th Feb 2002)
7. J2SE 5.0 (30th Sep 2004)
8. Java SE 6 (11th Dec 2006)
9. Java SE 7 (28th July 2011)
10. Java SE 8 (18th March 2014)
11. Java SE 9 (21st Sep 2017)
12. Java SE 10 (20th March 2018)

Reference: https://www.javatpoint.com/history-of-java

DISCLAIMER

- These slides are not original and have been prepared from various sources for teaching purpose.

Sources:

- Herbert Schildt, Java^{TM}: The Complete Reference

THANK YOU