Comparing C++ to Java

CSC 340 - Appendix K, Credit to Hui Yang

February 3, 2016

Overview

❖ **Brief history**

❖ Why is C++ important?

❖ Main differences

Brief History of C++

❖ C was developed by Dennis Ritchie at AT&T Bell Labs in the 1970s

  ❖ To implement the Unix operating system

  ❖ Efficiency

❖ C++ was developed by Bjarne Stroustrup at AT&T Bell Labs in the 1980s

  ❖ Incorporate object oriented programming

  ❖ Stay compatible with C as much as possible (C remains a subset of C++)

  ❖ Efficiency: run as fast as C

*Not really "pure" but closer than C++

Brief History of Java

* Java was developed by Sun Microsystems in the early 1990s

    * Marked as a modern alternative to C++

    * Syntactically similar to C++ ("C++ Lite")

    * Pure* object oriented language

    * Designed to be hard for programmers to write incorrect programs

Overview

* Brief history

* **Why is C++ important?**

* Main differences

Why is C++ Important?

* Why does a 35 year old language still dominate "real" development?

* A recent interview with Bjarne Stroustrup by Info World
http://www.infoworld.com/article/2608770/application-development/application-development-stroustrup-why-the-35-year-old-c-still-dominates-real-dev.html

TL;DR

* C++ remains vital and relevant … because of **its ability to handle complexity** ("nothing that can handle complexity runs as fast as C++"), making it the go to solution for telecom, financial, and embedded applications and online systems such as Amazon and Google.

* He used C++ for projects that "required a real programming language and **real performance**" [as C++ is for] high performance, high reliability, small footprint, low energy consumption, all of these good things.

* C++ is **NOT FOR** small apps or hobbyists

Why learn C++?

* Still widely used in industry and academia

    * Many job openings require the candidates to master C++

* C++ is, in general, much more efficient in both space and time

    * Most high performance computing systems are written in C/C++

* C/C++ is supported by many parallel programming standards

Don't generics handle templates?

Why learn C++?

* Pointers

    * Allow programmers to optimize the performance of a program

* Templates: A missing feature in Java

    * Allows programmers to write generic, type independent code

* Operator Overloading: Another missing feature in Java

    * Allows programmers to define operators (+, =) for user specified data types (or classes)

Overview

* Brief history

* Why is C++ important?

* **Main differences**

Main Differences

* C++ is Compiled
  * C++ code is compiled into native code
  * Poor cross-platform compatibility
  * Runtime efficiency
* Java is Interpreted
  * Java code is compiled into bytecode that is interpreted by the JVM at runtime
  * Good compatibility (runs on any system with a pre-existing JVM)
  * Less than ideal runtime efficiency

Main Differences

* C++ is organized by libraries

* Java is organized by packages

Main Differences

	C++	Java
API	small	huge
Compiler Checks	some	numerous
Garbage Collection	no	yes
Operator Overloading	yes	no
Check array indices	no	yes
Standard GUI library	no	yes
Exception handling	some	yes
Pointer variables	yes	no
Multi-threading	platform specific	yes

Main Differences
Boolean Variables

❖ C++

❖ Data type: `bool`

❖ Values: `true` and `false` (but they are actually integers 1 and 0)

❖ In a predicate test clause, any non-zero will be considered true

❖ Java

❖ Data type: `boolean`

❖ Values: `true` and `false`

Main Differences
Strings

❖ C++

❖ Provided as a library

❖ `#include <string>`

❖ Java

❖ Integrated into Java core

❖ Class `String`

Main Differences
Constants

❖ C++

  ❖ Declared outside of a class using the keyword `const`
    ```
    const MAX_SIZE = 20;
    ```

❖ Java

  ❖ Declared within a class or method using `static final`
    ```
    public static final int MAX_SIZE = 20;
    ```

Main Differences
Arrays

❖ C++

  ❖ Arrays are sequences of objects

  ❖ ```
    int data[MAX_SIZE]; // fixed size array
    int data[] = new int[MAX_SIZE]; // dynamic array
    delete data; // programmers must deallocate memory
    ```

  ❖ The C++ equivalent of Java Array is the vector
    ```
    #include <vector>
    ```

❖ Java

  ❖ Arrays are objects with a public field `length`

  ❖ Exception is thrown if attempting to access a position outside of array bounds

  ❖ ```
    int[] data = new int[MAX_SIZE];
    ```

Not sure those unsigned ranges are correct….

Main Differences
Unsigned Integers

❖ C++

  ❖ Any numeric data type can be declared as unsigned
    ```
    // in the range [-231, 231]
    int x;
    // in the range [0, 232]
    unsigned int y;
    ```

❖ Java

  ❖ Does not exist

Main Differences
Functions

+ C++

  + Do not need to be inside of a class

  + Function prototype
```
double is_even(int x);
```

  + Types of functions

    + Stand alone

    + Member

    + Non-members functions in a class

+ Java

  + Methods only (members)

Function prototypes used to tell the compiler about a function signature before its definition

---

Main Differences
Classes

+ C++

  + Require ; at the end of a class declaration

  + Separate compilation: header and implementation file

+ Java

That ; will bite you a few times this semester!
We'll cover headers and implementation files later this semester

---

Main Differences
Input and Output

+ C++

  + Does not have System.in and System.out

  + Uses `cin` and `cout`
```
#include <iostream>
using namespace std;
cin >> number;
cout << number;
```

  + Other ways
```
scanf("%d%s", &number, string);
print("%d%s", number, string);
```

Main Differences
Input and Output

✤ Java

  ✤ System.in and System.out

```
Scanner keyboard = new Scanner(System.in);
System.out.println("How old are you?");
int age = keyboard.nextInt();
System.out.println("You are " + age + " years
old");
```

Main Differences
Compilation

✤ C++

  ✤ All the classes, functions, and variables must be declared before they are used

  ✤ Uses a pre-processor directive so that classes can be *specified* in a separate header file
```
#include "myClass.h" // pre-processor directive
```

✤ Java

  ✤ Automatically searches for class implementation in an imported package (or on CLASSPATH)

Java's Unique Features

✤ Keyword `synchronized`

✤ JavaDocs

  ✤ doxygen.org for C++

✤ Operator `instanceof`

✤ Interfaces

✤ Large standardized class library

C++ allows classes containing only pure virtual methods (requiring implementation in derived classes)

C++ Unique Features

- Constant and Macros
  `#define min(a, b) (((a) < (b)) ? (a) : (b))`

- Compilation directives
  ```
  #ifdef HOST_SPARC
  #include "myheader.h"
  #endif
  ```

- Templates

- Life outside of a class - it's common to declare and define functions and variables outside of a class

- Pointers and Memory Management

  - `new` operator allows the programmer to request memory

  - `delete` operator allows the programmer to return memory

Conclusion

- C++ and Java share many similarities in syntax

- Designed with different goals

- Essential and practical to master both languages