COMP 322: Introduction to C++

Chad Zammar, PhD
Jan 08, 2021

Comp 322

- 1 credit crash course
- Basics of C++
- C++ from C perspective
- C++ from Java perspective

Prerequisites

- You took at least some of the following:
  - Comp 250
  - Comp 206 or Comp 208
- You already know C
- You already know Java
- You already had a cup of coffee (or tea)

Objective

COMP 322 is:

- Introduction to C++
- Basics of Object Oriented Programming in C++
- Highlighting main differences between C++, C and Java

COMP 322 is not:

- Introduction to programming
- Advanced C++ techniques

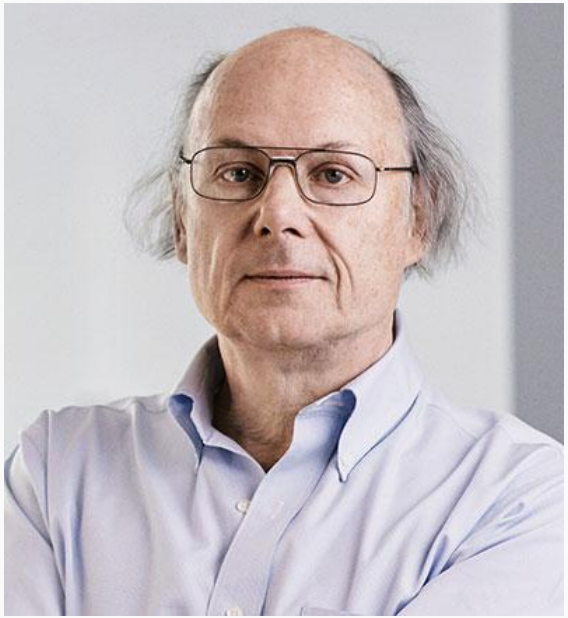
Resources

- No official textbook
- Recommended books:
  - The C++ Programming Language by Bjarne Stroustrup
  - Accelerated C++: Practical Programming by Example by Andrew Koenig
  - C++ Primer by Stanley Lippman and Barbara Moo
- MyCourses (Course webpage)
- C++ tutorials on the Internet (www.cplusplus.com, www.learncpp.com)
- E-mail: chad.zammar@mcgill.ca
- TAs contact info: check website
- Office hours: TR or by appointment

Evaluation

- **Two** quizzes during lecture time (20% each)
- **Three** homework assignments (20% each)
  - To be submitted via myCourses before the deadline
    - Penalties apply for late submission
  - No email submission
  - Code must compile and execute
  - Academic integrity: https://www.mcgill.ca/deanofstudents/plagiarism

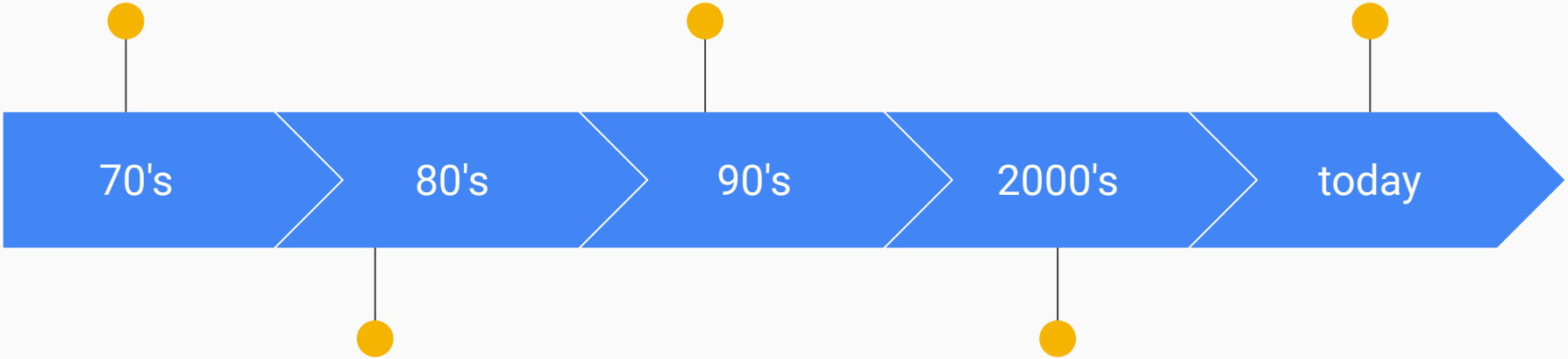
C++ was created by Bjarne Stroustrup



From Assembly to C language by Dennis Ritchie. Origin is tied to the development of the Unix Operating System.

In 1998, the C++ standards committee published the first international standard for C++ ISO/IEC 14882:1998.

Latest revision (C++20) was supposed to be released in December 2020.



From C to C++ by Bjarne Stroustrup. Combining features found in Simula with the performance of C.

C++11 (the new C++ standard) was released

Is C++ still relevant after 40 years?

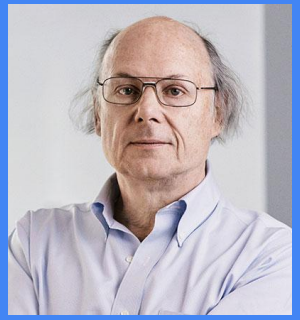
- **Most popular programming languages as of now:**
  - **Java, C, Python, C++, C# and Javascript**
- **C and C++ are the most performant languages from the above list**
- **C++ is used in:**
  - **Almost everything and everywhere**

“There are only two kinds of languages: the ones people complain about and the ones nobody uses.”

— Bjarne Stroustrup, The C++ Programming Language

“What the h*** is C++ anyway?”

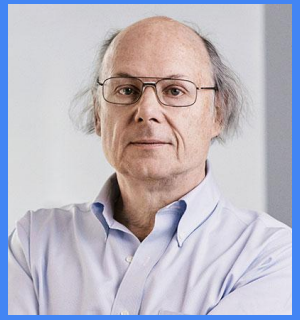
— My Wife, couple days ago



C++ vs C design

- C++ derives directly from C, so it is C plus plus more stuff
- Object Oriented via Classes
  - In C data and functions are separated
  - In C++ they are encapsulated within a class where data can be hidden
- Generic programming via Templates
- Redesigned memory management via new/delete
- Operator (and functions) overloading
- Namespaces
- Reference variables
- Exception handling

C++ vs Java design



C++

- Compiled language
  - Runs as native binary on a target
- Compatible with C code (very few rare exceptions)
- Allows multiple programming paradigms without discrimination
- Allows multiple inheritance of classes
- Allows manual low level memory management via pointers

Java

- Interpreted language
  - Runs through a virtual machine
- Uses JNI (Java Native Interface) to call C/C++ code
- Allows multiple programming paradigms but strongly favors Object-Oriented
- Single inheritance for classes
- No pointers and provides automatic garbage collection mechanism
- James Gosling is Canadian :)

Compiling and running C++ code (1/2)

- C++ code can be contained
  - in a single file or
  - it can span over multiple files
  - Common practice to separate header files (.h) containing declarations from implementation files (.cpp)
  - Remember that, unlike C++, Java requires a different file for each public class and requires that the name of the file matches the name of the class
- C++ common file extensions:
  - .cpp, .c++, .c, .cxx, .cc, .hpp …
- Use any file editor or IDE to write C++ code

Compiling and running C++ code (2/2)

- C++ code need to be compiled in order to run as an executable
- gcc / g++ under Linux, MSVC under windows, Clang under OSX
- Example: ***g++ example.cpp -o example***
- Usually when we compile we invoke two operations (actually 3 if we consider pre-processing):
  - *Compilation*: transforms source code to object file (intermediate step between source code and final executable file)
  - *Linking*: producing one executable file from multiple object files
- Common practice is to use special script called *Makefile* for compiling complex projects
- www.cpp.sh : simple frontend gcc compiler for quick coding and debugging

```
/*
============================
 Name        : helloWorld.cpp
 Author      : Chad
 Description : Hello World in C++
============================
*/

#include <iostream>

using namespace std;

// main function
int main()
{
    cout << "Hello World!" << endl;
    return 0;
}
```

- /* … */ block comment
- // single line comment
- # … preprocessor command
    - #include <iostream> dumps in the content of iostream
- using namespace std
    - Means using namespace std :)
- main() is entry point function
- Operator << to write to cout object
- endl: end line and flush stream
- return 0; to signal that code has completed successfully