

From C to C++

- Version 1: Dr. Ofir Pele
- Version 2: Dr. Miri Ben-Nissan
- Version 3: Dr. Erel Segal-Halevi

Why C++ is much more fun than C (C++ FAQ)?

1. Classes & methods - OO design
2. Generic programming - Templates allow for code reuse
3. Function and operator overloading
4. Stricter type system (e.g. function args)
5. Some run-time checks & memory control

**A common and mature language that
gives you high level and low level control**

Have fun 🎵

Why C++ is much more fun than C (C++ FQA)?

1. Tons of corner cases
2. Duplicate features
3. Cryptic syntax
4. Undecidable syntax (uncompilable programs!)
5. No two compilers agree on it

Probably one of the hardest computer languages to master.

Have fun 🎵

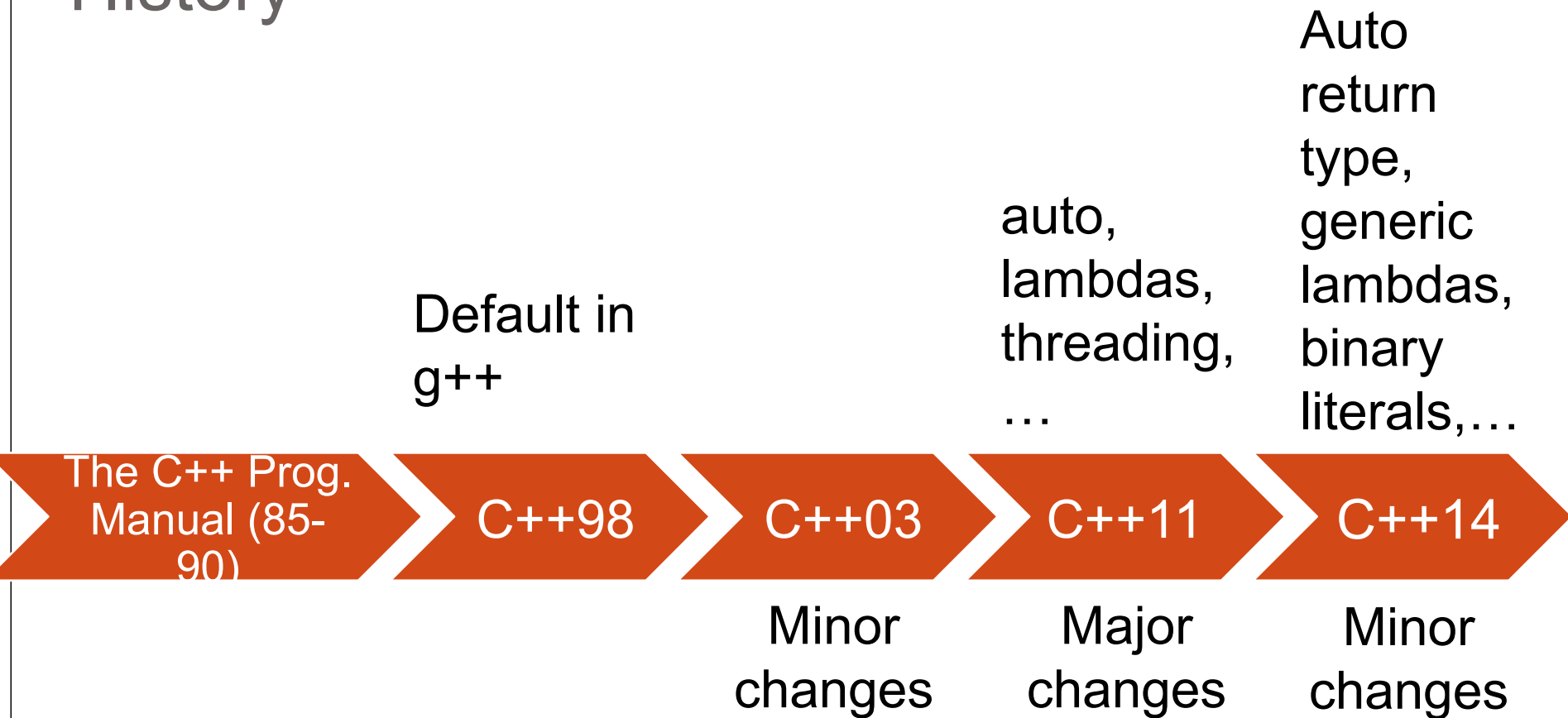
C++ vs. Java

Java is much simpler to program -
removes many ambiguous and duplicate features.

So why use C++ at all?

1. *Basis to other languages.* Many other languages and operating systems are written in C/C++ (e.g Java, Python, Windows, Unix/Linux, Mac...).
2. *Understanding.* With C++ you must understand what the computer is doing under the hood.
3. *Zero-cost abstraction.* tight memory and time management – important in embedded systems and real-time systems.
4. *Common in big systems.* Google, Facebook and other big companies use (also) C++.

History



**We'll learn parts of C++-11, 14, 17,
Mostly parts that makes C++ more “pythonic” while keeping it
efficient**

Future



C++17

C++20

...

Basic Unix commands

ls

ls -latr

cd [dirname]

mkdir [dirname]

cat [filename]

nano [filename]

grep [word]

source [filename]

rm [filename]

grep ... < input.txt

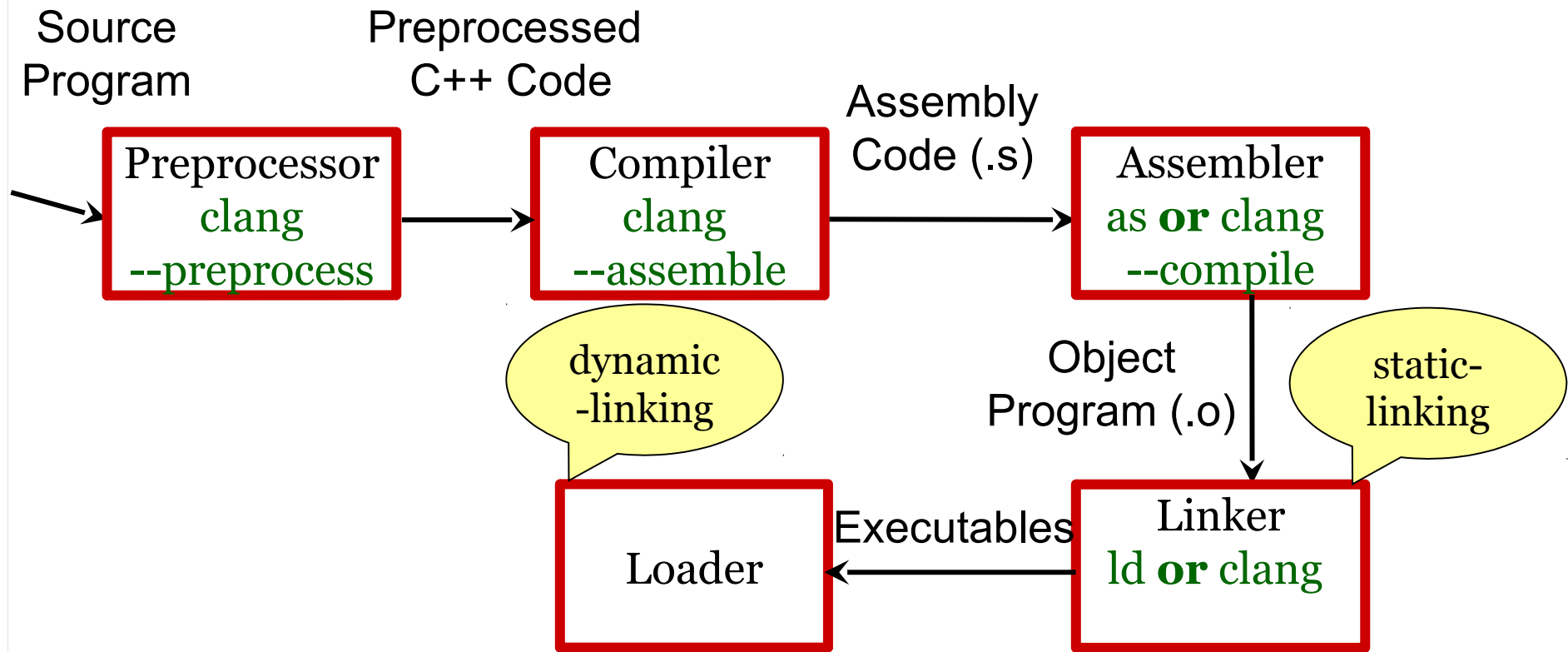
grep ... > output.txt

ls -latr | grep bash

su

exit

sudo apt install ...



The missing types

Fill in missing types from C,
in somewhat crude way

strings in C++

```
#include <iostream>
```

```
#include <string>
```

```
int main()
```

```
{
```

```
    std::string str;
```

```
    int a;
```

```
    double b;
```

```
    std::cin >> str >> a >> b;
```

```
    if(std::cin.fail())
```

```
    {
```

```
        std::cerr << "input problem \n";
```

```
        return 1;
```

```
    }
```

```
    std::cout << "I got: " << str << ' '
```

```
    << a << ' ' << b << std::endl;
```

```
}
```

:More about string functions

<http://www.cppreference.com/cppstring>

Boolean variables

```
#include <iostream >
```

```
int main()
```

```
{
```

```
    int a = 5;
```

```
    bool isZero = (a == 0);
```

```
    // same conditions
```

```
    if(!isZero && isZero == false &&
```

```
        isZero != true && !!! isZero && a )
```

```
    {
```

```
        std::cout << "a is not zero\n";
```

```
    }
```

```
}
```

Good
style



C++-11 enum class

```
enum class Season : char {  
    W I N T E R, // = 0 by default  
    S P R I N G, // = W I N T E R + 1  
    S U M M E R, // = W I N T E R + 2  
    A U T U M N // = W I N T E R + 3  
};
```

```
Season curr_season;
```

```
curr_season= Season::AUTUMN;
```

```
curr_season= SUMMER; //w on 't com pile! (good)
```

```
curr_season= 19; //w on 't com pile! (good)
```

```
int prev_season= Season::SUMMER; //w on 't com pile!  
(good)
```

Overloading

Understand and remember.

- More than syntactic sugar.
- This is how a lot of stuff works under the hood (e.g. inheritance)

Function overloading - C

```
#include <stdio.h>
void foo ()
{
    printf ("foo()\n");
}
void foo (int n)
{
    printf ("foo(%d)\n", n);
}
int main ()
{
    foo(12);
    foo();
    return 0;
}
```

Compilation output:

**Error:
Multiple
definition of
foo**

Function overloading – C++

```
#include <iostream>
void foo() {
    std::cout << "foo()\n";
}
void foo(int n) {
    std::cout << "foo(" << n << ")\n";
}
int main() {
    foo(12);
    foo();
}
```

Output:

Compile, and print:

foo(12)

foo()

Default parameters

```
#include <iostream>
```

```
void foo(int n=5)  
{  
    std::cout << n;  
}
```

```
int main()  
{  
    foo();  
}
```

Output:

Compile, and print:
foo(5)

Overload resolution

1. Find all functions with same name “candidates”. Let’s call them O1.
2. Find O2 subset of O1 which have the correct number of arguments - “viable candidates”
3. Find O3 subset of O2 with best matching arguments.
if $|O3|=1$
 use that function.
else (0 or more than 1):
 emit compilation error.