A brief history of C++

Victor Eijkhout, Susan Lindsey

Fall 2022

last formatted: October 13, 2022

1. C with classes

Bjarne Stroustrup

1980 or so: adding classes to C.

2. Query the standard

If you want to detect what language standard you are compiling with, use the `__cpluscplus` macro:

Code:

```
cout << "C++ version: " <<
     __cplusplus << '\n';
```

Output

```
[basic] version:

C++ version: 201703
```

This returns a `long int` with possible values 199711, 201103, 201402, 201703, 202002.

3. C++98/C++03

Of the C++03 standard we only highlight deprecated features.

- `auto_ptr` was an early attempt at smart pointers. It is deprecated, and C++17 compilers will actually issue an error on it. For current smart pointers see chapter **??**.

4. C++11

- `auto`
- Range-based for.
- Lambdas. See chapter **??**.
- Variadic templates.
- Unique pointer.
- `constexpr`

5. C++14

C++14 can be considered a bug fix on C++11. It simplifies a number of things and makes them more elegant.

- Auto return type deduction:

```
auto f() {
  SomeType something;
  return something;
}
```

- Generic lambdas. Also more sophisticated capture expressions.
- Unique pointer
- `constexpr`

6. C++17

- Optional; section **??**.
- Structured binding declarations as an easier way of dissecting tuples
- Init statement in conditionals

7. C++20

- modules
- coroutines, another form of parallelism.
- concepts
- spaceship operator
- ranges
- calendars and time zones
- text formatting
- `span`.
- `numbers`.
- Safe integer/unsigned comparison; section **??**.