Configuring Your Compiler: Choosing a Language Standard

Choosing a C++ Language Standard

With many versions of C++ available (`C++98`, `C++03`, `C++11`, `C++14`, `C++17`, `C++20`, `C++23`, etc.), the question arises: **how does a compiler know which version to use?**

Compiler Defaults

Compilers typically have a **default standard** they use when compiling programs. However:

- The latest version is usually **not** the default.
- Many compilers still default to `C++14`.

The most commonly recommended standard for use is one or two versions back from the latest finalized standard. This ensures stability, as bugs in the standard or compiler implementation are often resolved over time.

For **personal projects and learning**, it is best to use the latest standard to stay up-to-date and learn modern C++ features.

Understanding the C++ Standards Document

Each C++ language standard is described in a **standards document**, which serves as the authoritative reference for:

- Rules and requirements of the language.
- Implementation guidance for compiler developers.

Important Notes:

- The standards document is not designed for learning C++; it is highly technical.
- People often quote the standards document to explain specific behaviors in C++.
- The approved standards documents are not freely available. They can be purchased, with a link provided to buy the latest version.

Compiler Support and Compatibility

Even after a language standard is finalized, compilers often have:

- Missing features.
- Partial implementations.
- Buggy support for certain features.

To track compiler support for features in each C++ language standard, you can refer to the **CPPReference website**. Their home page includes a section called *Compiler Support* (located in the top-right corner) with tables showing feature support by compiler and language standard.

Dealing with Unsupported Features

If you encounter unsupported features in your compiler:

- Upgrade to the latest version of your compiler if it supports the feature.
- If the issue persists, try using a different compiler that provides better support.
- Alternatively, find a solution using a different set of features supported by your current compiler.