5 - How C++ Works

sábado, 25 de janeiro de 2025 17:12

How C++ Works

- Code -> Compile -> Binary
- Everything that starts with a # is a preprocessed statement
  - Happens just before the compilation
  - include ->
    - Looks for a file
    - past all the content in the place it's called
- Main function
  - Every c++ program has na entry point
  - execution in order but we can break this depending on the code
  - main funcgion don't need to return nothing, but it's defined as integer
  - << overload operator, thing about it as functions
- The compile changes the c++ code into binary code
- We have default projects as Debug and Release
  - A set of rules that applies to a project
  - As compilation target x86, x32...
  - There are more rules like
    - Configuration
    - Platform
    - SDK
    - Output directory
    - Configuration, set to lib or exe
  - Comp9ile settings are in c/c++
    - a lot kkk but basecly libs
  - Debug mode is slower than release mode, because the compiler has more optimizations
- Each c++ file in the project gets compiled
  - Compiled in something called object file .obj
  - Some way to put everything together, here is where the linker comes
    - Take all .obj files and put it together in one .exe file

- Header files does NOT get compiled
- Compile the cpp file
  - CRTL + ALT +7
  - OR CLICK ON BUILD
- eRROR LIST IS MORE LIKE NA OVERVIEW, because it query the output to the word error. The best way is to read the output directly
- What files a compiler creates?
  - .obj file that the compiler generates (one file for each c++)
- We can pass declarations or definitions
  - Declarations:
    - Just tell the compiler that the function exists and it belives it
    - But the linker error comes up... because here is where it'll look for that defined function
  - Definitions:
    - Has the actual implementation with the body implementation