1.7 Integrated development environment

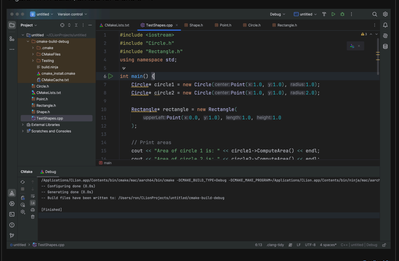
Introduction to IDEs

When developing software, a programmer must write, compile, test, and debug code. Each of these tasks can be performed with different software. Ex: A text editor is needed to write code, and a compiler is needed to compile the code. An ***integrated development environment*** (***IDE***) is software that integrates a text editor and a compiler, often with additional tools. Ex: Most IDEs include the ability to edit multiple files, a file manager for finding and organizing files, shortcut buttons, a debugger to help a programmer find bugs, and a console for entering commands and executing programs.

IDEs support multiple programming languages and are widely used in software development. Common IDEs for C++ include Visual Studio Code, XCode, and CLion.

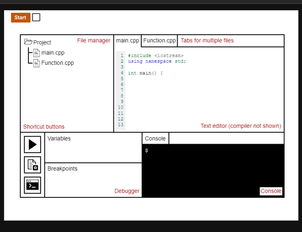
Most IDEs do not come pre-installed with an operating system and must be installed separately. Some IDEs are run locally on a computer, while others are cloud-based and are run using a web browser.

Figure 1.7.1: CLion, an IDE for C and C++.



Feedback?

PARTICIPATION ACTIVITY 1.7.1: An IDE integrates various programming tools.



Captions

Feedback?

PARTICIPATION ACTIVITY 1.7.2: IDEs.

1) What does IDE stand for?
   - Identity environment
   - Integrated development environment
   - Integrated development executable

2) Which software is necessary when programming either with or without an IDE?
   - Text editor only
   - Text editor and compiler
   - Text editor, compiler, and debugger

3) Which of the following is an advantage of using an IDE?
   - Fewer steps are needed when compiling and running a program using an IDE.
   - An IDE uses less memory and fewer system resources than separate tools.
   - An IDE is less expensive than separate tools.

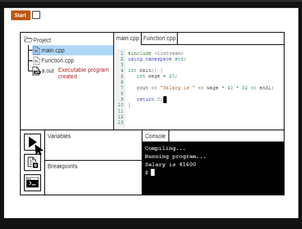
Feedback?

Common features of an IDE

IDEs contain many features that can make programming more efficient and convenient. Common IDE features include the following.

- ***Syntax highlighting*** uses different colors for keywords, variables, strings, and other code syntax elements. Syntax highlighting can help a programmer identify errors such as unclosed quotes and parentheses, missing semicolons, and misspellings.
- ***Automatic delimiter completion*** adds a matching closing quotation mark, parenthesis, bracket, or brace when a corresponding opening symbol is typed, which can help errors be more easily identified.
- ***Automatic indentation*** adds indents to lines after braces, which can make code more readable.
- Shortcut buttons allow various predefined operations to be performed with a single click. Ex: A common shortcut button allows a program to be compiled and run quickly without having to enter a command to compile a program into an executable and then running that executable.
- A file manager and multiple text editor tabs for separate files allow different programs, input files, and output files to be easily accessed and edited.

PARTICIPATION ACTIVITY 1.7.3: Common features of an IDE.



Captions

Feedback?

PARTICIPATION ACTIVITY 1.7.4: Common features of an IDE.

1) What is syntax highlighting?
   - Automatic correction of syntax errors
   - Appearance of different code elements in different colors
   - Addition of the matching closing parenthesis when an open parenthesis is typed

2) In the animation above, the programmer was able to identify an error because of the wrong colors in the following code:

```
cout << "Salary is " wage * 52 << endl;
```

Use syntax highlighting to identify the error.
   - cout is misspelled.
   - The semicolon at the end of the statement is missing.
   - A quotation mark is missing.

3) A programmer finds that the following code does not compile.

```
#include <iostream>
using namespace std;

int main()
int salary = 50000;

cout << "Salary is " << salary << endl;

return 0;
}
```

How could automatic delimiter completion or automatic indentation have helped the programmer identify the error?
   - Automatic delimiter completion should have added >> symbols corresponding to the << symbols, but the >> symbols are missing.
   - Automatic indentation should have indented the code following `int main()` {, but that code is not indented.
   - Automatic delimiter completion should have added quotation marks around the variable salary, but salary lacks quotation marks.

4) In the animation above, the programmer switched to edit a different file by clicking a different filename in the file manager pane. How else could the programmer have switched between files?
   - By clicking the tabs along the top of the text editor
   - By entering the filename in the console
   - By typing the filename in the text editor

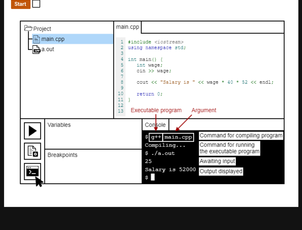
Feedback?

Console and command-line interface in an IDE

An IDE typically includes a console for user input and program output. A ***console*** (or ***terminal***) is a text-based interface that allows a user to run programs, enter input, and view program output.

A console's ***command-line interface*** (***CLI***) is an interface that allows a user to enter commands to run programs and work with files. One advantage of a CLI is the ability to run a program with command-line arguments. A ***command-line argument*** is a value entered by a user after the program name when running a program from a command line. Ex: In the command `ls -a`, ls is the name of the program and -a is a command-line argument.

PARTICIPATION ACTIVITY 1.7.5: Console and command-line interface.



Captions

Feedback?

PARTICIPATION ACTIVITY 1.7.6: Console and command-line interface.

1) Which of the following is displayed in a console?
   - Input only
   - Both input and output
   - Output only

2) In the following command, what are the command-line arguments?

```
salary.exe 0.5 25
```

   - salary.exe
   - salary.exe 0.5 25
   - 0.5 25

3) What is an advantage of running a program through a CLI in an IDE?
   - A CLI allows a program to easily be run with many different arguments.
   - Using the CLI to run a program is faster than using the shortcut button in the IDE.
   - The CLI allows an already-compiled executable program to be run without recompiling each time changes are made to the source file.

Feedback?

How was this section? | Provide section feedback