

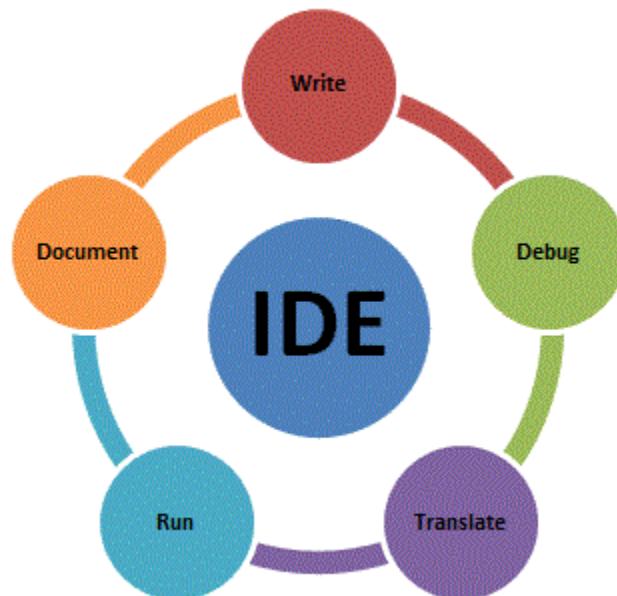


Introduction

Programmers write programs! There are a number of key tasks that they need to do every time they write a new program. These include actually writing the code, finding errors with it if there are any, translating it, running it and documenting the code, in case it has to be changed in the future. Because these jobs have to be done again and again, programs called **Integrated Development Environments (IDE)** have been developed. These programs allow a programmer to do all of the common tasks involved in writing a program in one place easily.

What is an IDE?

An IDE is a software application that provides a programmer with the tools to write, debug, translate, run and document applications.



Typical tools include:

- **A text editor**, so that you can write and edit programming code. You type the programming code into the text editor and also make any changes you need to make in this tool.



- **Debugging tools** also called '**error diagnostics**', so you can find any errors in programs. This is also known as '**debugging a program**'. These tools help you locate and identify syntax problems with the code.
- **Translators**, so you can turn the source code into object code.
- **A runtime environment**, so you can actually run the code and see the results.
- **Auto-documentation tools**, IDEs may automatically create documentation about the code as it is being written, including what variables close variable memory location within a computer program where values are stored., arrays and subroutines are used for.

Why are IDEs important?

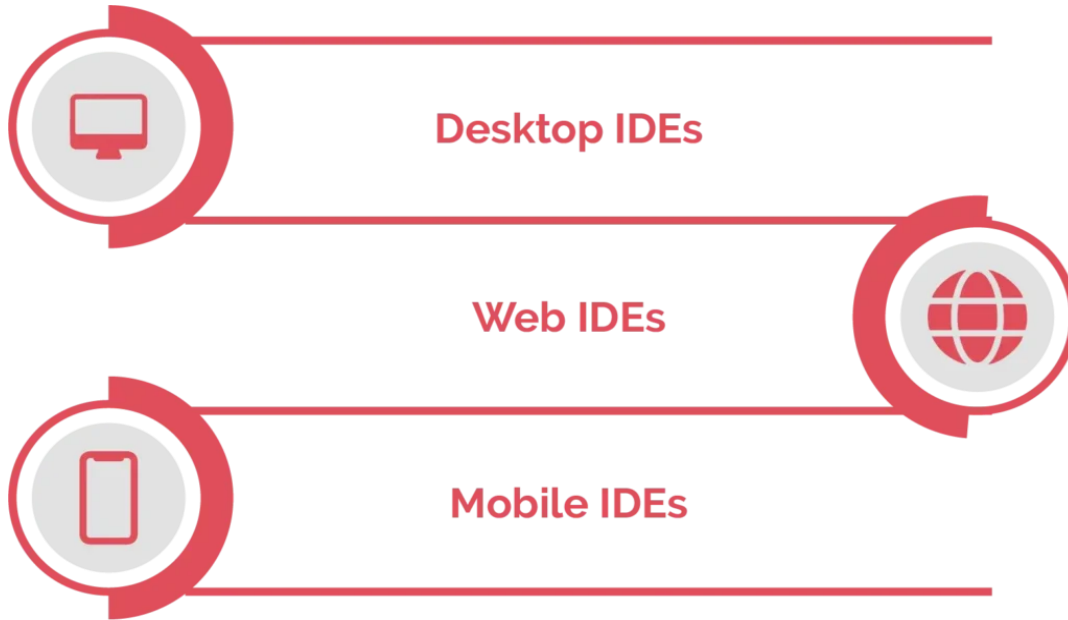
One certainly does not need an IDE to code or develop applications. Even a simple text editor like **notepad** can be used to write code. However, IDEs offer some stunning features that go beyond ordinary editing. By providing frequently used **developer tools** all in one simple interface, one can directly get on to building their applications without going through the hardship of manually configuring and integrating the development environment.

Why do Developers Use IDE?

By providing a single, unified environment for managing all aspects of the development process, IDEs can help improve a developer's productivity, code quality, and overall development experience.

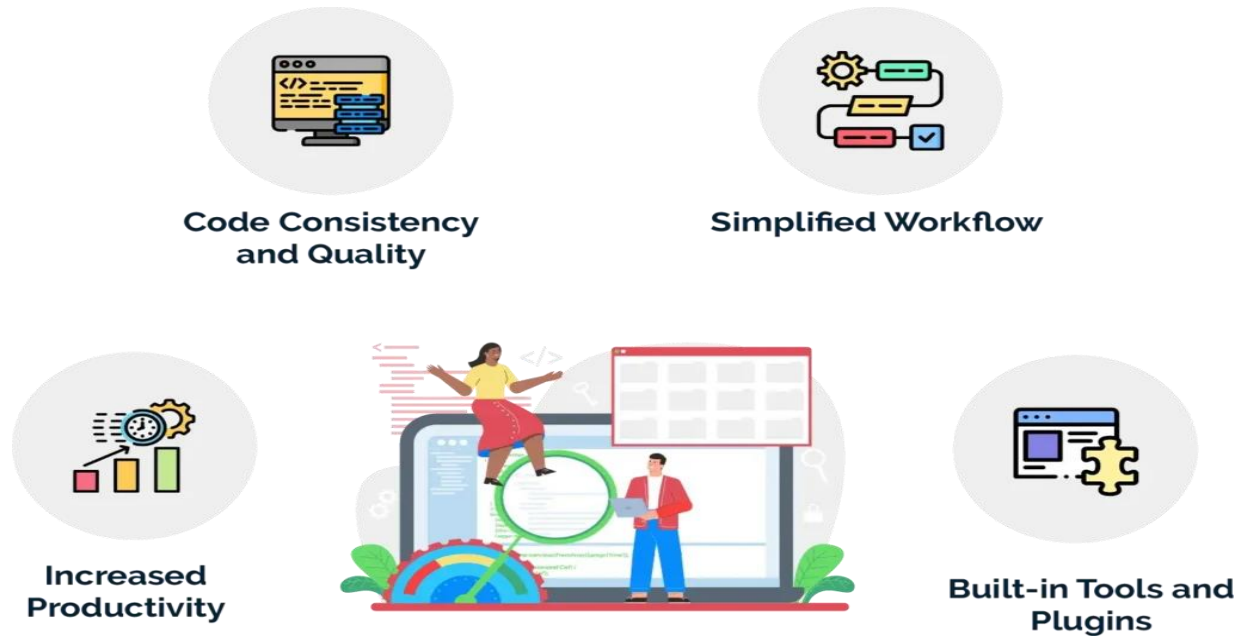
Types of IDEs

When used to describe toolkits, it is clear that 'IDEs are not a single entity – they come in various forms, tailored to suit specific programming languages, platforms, and development environments. Here are some of the most popular types:



- **Desktop IDEs:** This type includes the IDEs that can be configured locally. They do not need an active internet connection to build/run programs. Desktop IDEs are highly customizable to suit developer-specific needs and provide performance irrespective of the internet speed. Examples: include Microsoft Visual Studio, Eclipse, Netbeans, etc.
- **Web IDEs:** Web IDEs have also been adopted, with the emergence of cloud computing as the main technology ideal for handling web applications. These IDEs are also web-based and can be used in your browser. They enable you to write, test, and share your code with your team and others irrespective of the place, provided that there is an internet connection. Such platforms for development as Cloud-based and Web IDEs like Cloud9 and Repl. It is perfect for software as a service project.
- **Mobile App Development IDEs:** These IDEs are specially designed for creating mobile applications. They include features like Emulator support and integration for developing and testing mobile applications. Mobile Development IDEs can be in the form of Desktop IDE or Cloud IDE. Examples: Include Android Studio, Flutlab.io, etc.

Advantages of Using an IDE



Increased Productivity

Using an integrated development environment (IDE) increases your productivity. By integrating many software development tools into a single interface, many IDEs streamline your workflow. You can focus on writing code without juggling multiple tools. This is especially useful in mobile development and [Software-as-a-Service \(SaaS\)](#) applications. Additionally, features like syntax highlighting and code completion help developers work more efficiently. The importance of IDEs in modern application development cannot be overstated.

Code Consistency and Quality

IDEs help maintain code quality and consistency. Many IDEs come with built-in code formatting and linting tools. These tools ensure your code adheres to best practices and coding standards. Whether you are involved in web development or mobile apps, these features are invaluable. They help developers identify and fix errors early in the development process. Consequently, standardized development practices improve the overall quality of your projects. Without an IDE, maintaining such consistency would be challenging.



Simplified Workflow

An IDE simplifies your workflow significantly. You can easily navigate through your codebase, making it simpler to find and fix errors. Additionally, IDEs often include project management tools and features that aid in organizing your development tasks. The integration of various development components within the IDE makes it easier for developers to manage their projects. This streamlined approach allows developers to test and deploy their applications more efficiently. The benefits of using IDEs in complex projects are immense.

Built-in Tools and Plugins

Just about every IDE has a vast selection of incorporated tools and it also offers support to most of the plugins. All of them contribute to the IDE which extends the functionality and enables developers to personalize the IDE based on their working requirements. For instance, Visual Studio is a development environment for building applications using different programming languages, and it also supports numerous plugins.

Disadvantages of using IDE

Some of common disadvantages of IDEs or Integrated Development Environments are as follows:

- **Not beginner Friendly:** IDEs are complex tools and generally provide quite an intimidating UI, which can be a little tough to comprehend for a beginner. Maximizing their benefit generally needs a dedicated effort.
- **Frequent Updates:** It's difficult to keep up with constant updates, such as new samples, templates, and features.
- **Only Assist in writing code:** As a tool, IDEs (Integrated Development Environments) are limited to being an assistant. They can't write code or automatically fix errors, one still needs the knowledge to write clean code.



- **Cost:** Some IDE is Expensive or needs additional costs to unlock some add-ons, which can be a barrier for small-scale companies or learning Individuals.

Not: In this course, we will use the *Visual Studio* IDE with the *C++* programming language.

