While there are many programming languages out there, when it comes to games programming, one particular language stands out from the rest - *C++.* Why? Well, read on! But before we start, let's first determine what *C++* is.
C++ is a multi-paradigm, object-oriented programming language that was developed by Danish computer scientist Bjarne Stroustrup in 1979 and released in 1983. It is an enhancement to the *C* programming language, one of the most widely used programming languages of all time.
When it comes to developing games, there are many factors that influence developers in choosing their desired programming language. When looking at *C++*, it is clear to see why it is the choice of so many developers – it's very fast, efficient, extremely flexible and well supported. It has also inspired other programming languages to evolve, including *Java*, *Flash* and *C#*. Let's now take a look at some of this programming language's features in more detail.
First of all, what makes *C++* so fast? When coding in this language, you can write games that compile into a binary format, enabling the game to communicate directly with the operating system and hardware. It also has high and low level language features, allowing developers to write code in more human-readable form, or in assembly language. This enables the games developer to have full control over their game's behaviour.
If we briefly compare this to languages like *Java* or *C#*, you will see that games made with these languages run a little slower. Oracle's *Java Virtual Machine*

and Microsoft's *Common Language Runtime* are virtual runtime environments that allow for games made with *Java* and *C#* respectively to run on any kind of computer system that has any of these environments installed. This means games are compiled into managed code, which runs inside the virtual machine, which is then re-compiled into binary code that the operating system and hardware can understand. These types of games can now run on any machine, making them multi-platform. However, because of this "recompilation", the games run slower. The virtual engines that *Java* and *C#* offer are stable, safe and manageable environments that make it easier to handle and run the games, but they fail to offer the direct access between the game and the operating system or hardware, which game programmers favour so much.
Secondly, *C++* permits games to have direct access to memory. Although programmers have to make sure all memory allocated is equally de-allocated, they have control over what memory chunks are used and where and how they are used. *C#* and *Java* have what is called a *Garbage Collector* inside the virtual machine, which automatically removes unused data, thereby preventing

any unnecessary memory leaks. This may be helpful, but at the same time it gives very little memory control to the developer**.**

Another great feature of *C++* is that it's a *multi-paradigm programming language*, offering the programmer varying styles of development. One of the most popular and important styles included is *object-oriented programming*, or OOP for short, which makes use of *objects*, breaking down large scale game development into smaller segments. While this is by far the best way to make games, developers may also choose to program in the old *procedural* way, where code is executed one after the other. *C++* offers the best of both worlds by allowing an interchange between the different paradigms, where as other languages like *Java* force the developer to focus only on object orientation, thereby taking away that flexibility that game developers prefer.

It is also safe to state that *C++* has been around for a long time, and been used by many games development teams for nearly three decades. It is for this reason that there are a vast amount of external tools, third party libraries and 2D and 3D engines available that can be embedded with the native *C++* code to make games. Graphics APIs, such as *Simple DirectMedia Layer* or SDL for short, can be used together with native *C++* code to create games on the *Windows*, *Mac OS* and *Linux* platforms. This wonderful API can be used to draw images on screen, read player input and control gaming events that run constantly. On top of that, *OpenGL* can be added to *SDL* to further make use of 3D graphic libraries for more complex games. Most of these APIs and pre-built engines almost always have in some way or another been coded in *C* or *C++,* so if developers ever have to delve behind the scenes they would easily comprehend what is going on.

You will always find a compiler for *C++* on any system. Tools like *Visual Studio* will compile games for *Windows*, *Xcode* will compile for all *MacOS* and *iOS* platforms, and now programmers are even able to make use of Google's *Dalvik Virtual Machine* to incorporate *C* and *C++,* in order to create games for the *Android* platform.

Overall, this remarkable language is still the language of choice when it comes to games development. It may come with a slightly cryptic syntax, but it offers faster results and direct access. Having been around for just about 30 years, the libraries, tools and support that have evolved around it are immense. It is an extremely powerful tool that has helped to form other languages as well as create game engines and various other useful tools. It is therefore clear to see why *C++* is the game developer's tool of choice.

Useful Websites

C++
Oracle
Microsoft
Bjarne Stroustrup

All images have been sourced from Google Images