C++ Access Specifiers

Access Specifiers

By now, you are quite familiar with the public keyword that appears in all of our class examples:

Example

```
class MyClass { // The class
  public: // Access specifier
  // class members goes here
};
```

The public keyword is an **access specifier.** Access specifiers define how the members (attributes and methods) of a class can be accessed. In the example above, the members are public - which means that they can be accessed and modified from outside the code.

However, what if we want members to be private and hidden from the outside world?

In C++, there are three access specifiers:

- public - members are accessible from outside the class
- private - members cannot be accessed (or viewed) from outside the class
- protected - members cannot be accessed from outside the class, however, they can be accessed in inherited classes. You will learn more about Inheritance later.

In the following example, we demonstrate the differences between public and private members:

Example

```
class MyClass {
  public: // Public access specifier
        int x; // Public attribute
  private: // Private access specifier
        int y; // Private attribute
};

int main() {
 MyClass myObj;
 myObj.x = 25; // Allowed (public)
 myObj.y = 50; // Not allowed (private)
 return 0;
}
```

If you try to access a private member, an error occurs:

error: y is private

Note: It is possible to access private members of a class using a public method inside the same class. See the next chapter (Encapsulation) on how to do this.

Tip: It is considered good practice to declare your class attributes as private (as often as you can). This will reduce the possibility of yourself (or others) to mess up the code. This is also the main ingredient of the Encapsulation concept, which you will learn more about in the next chapter.

Note: By default, all members of a class are private if you don't specify an access specifier:

Example

```
class MyClass {
  int x; // Private attribute
  int y; // Private attribute
};
```