C++ Classes and Objects

The Building Blocks of OOP: Understanding Members and Access Specifiers.

What is an Object?

"An object is a real-world entity which has a state and behavior... It is the basic unit of object-oriented programming."

What is a Class?

- "A class is a **blueprint** from which you can create individual objects."
- "They are **user-defined data types** which hold their own data members and member functions."
- "A class is a logical entity. It doesn't consume any space in memory until an object is created."
- Syntax: The `class` keyword followed by the class name.

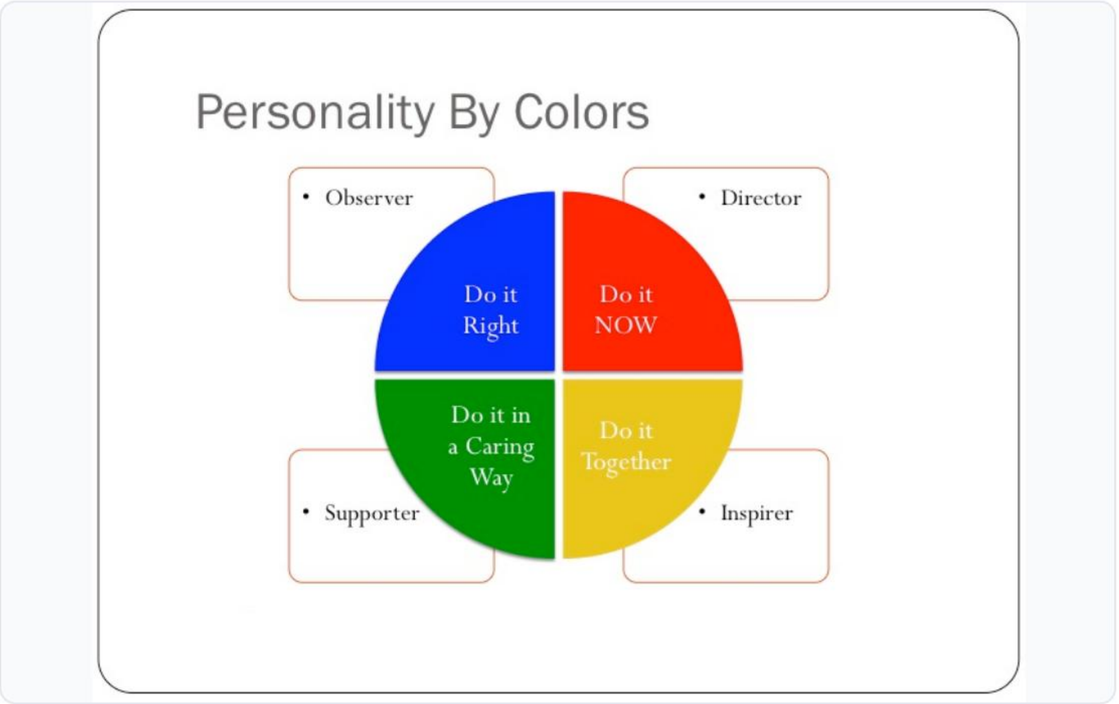
Class vs. Object: The Blueprint Analogy

The Class is the Blueprint

It defines the properties (e.g., `color`, `make`) and behaviors (e.g., `start_engine()`). It's a logical template.

The Object is the Car

It's a physical instance of the blueprint (e.g., `my_car`, `your_car`). It has real state (`color = "blue"`) and can perform actions. It occupies space in memory.



Inside a Class: The Members

Data Members

"These are the variables that define the properties or state of an object. (e.g., `int year`, `string make`)."

Member Functions

"These are the functions that define the behavior of an object. They operate on the data members. (e.g., `set_year()`, `display_info()`)."

Access Specifiers

"Access specifiers define how the data members and member functions of a class can be accessed from outside the class."

The 3 Access Specifiers

- `public` : Members are accessible from anywhere outside the class.
- `private` : (Default) Members are only accessible from within the class itself.
- `protected` : Members are accessible within the class and by any child (derived) classes.

Example: `private`

Private Access

Members are **accessible only inside the class**.

By default, all members in a C++ class are `private` .

```
#include
using namespace std;

class A {
    int a; // 'a' is private by default
public:
    A() { a = 1; } // Constructor
    void showA() {
        cout << "a = " << a; // OK
    }
};

int main() {
    A objA;
    // cout << objA.a; // ERROR!
    objA.showA(); // OK
    return 0;
}
```

Protected Access

Members are **accessible inside the class** and in any **derived child classes** (inheritance).

```
#include
using namespace std;

class Parent {
protected:
    int id_protected;
};

class Child : public Parent {
public:
    void setId(int id) {
        id_protected = id; // OK
    }
    void displayId() {
        cout << "id is: " << id_protected;
    }
};

int main() {
    Child obj1;
    obj1.setId(81);
    obj1.displayId();
    return 0;
}
```

Example: `public`

Public Access

Members are **accessible from anywhere** outside the class using the dot (`.`) operator.

```
#include
using namespace std;

class Circle {
public: // Public access specifier
    double radius;
    double compute_area() {
        return 3.14 * radius * radius;
    }
};

int main() {
    Circle obj;
    obj.radius = 5.5; // OK: 'radius' is public
    cout << "Area: " << obj.compute_area();
    return 0;
}
```

Access Specifier Summary

Specifier	Accessible Inside Class	Accessible in Derived Class	Accessible Outside Class
`public`	Yes	Yes	Yes
`protected`	Yes	Yes	No
`private`	Yes	No	No