Classes & Abstraction in C++

By:
freeCSIT

C++ course with PDF Notes

What is OOP?

It is faster and easier to execute

It provides a clear structure for the programs

keep the C++ code DRY

The major aim of developing OOP language is to remove some of the flaws encountered in programming a language of procedural approach.

The main aim of OOP is to bind together data and functions.

Object **P**rogramming **O**riented

Overview of OOP Concept

Classes and objects are the two main aspects of object-oriented programming.

Features of OOP

1. Classes
2. Object
3. Inheritance
4. Polymorphism
5. Encapsulation

C++ is a general-purpose Object-Oriented Programming Language

C++ includes features of OOP as well as conventional procedural programming

An Object is the programming representation of the real-world entity.

The major aim of developing OOP language is to remove some of the flaws encountered in programming a language of procedural approach.

Overview of OOP Concept

Classes and objects are the two main aspects of object-oriented programming.

Features of OOP

1 Classes

It is a user-defined data types, which holds its own data member functions which can be accessed and used by creating an instance of the classes. We can use in our program.

a class is a template for objects

Classes provide users a way to create user defined data types.

Classes provide a convenient way to group related data and the methods that operate on data together

A class defines the structure and behavior (attributes and methods) of objects.

It is a blueprint for creating an object

Overview of OOP Concept

Classes and objects are the two main aspects of object-oriented programming.

Features of OOP

1. Classes

Characteristics of Classes

1. A class is a template that units data and operations
2. A class is an abstraction of the real world entities with similar properties
3. A class identifies a set of similar objects
4. A class is implementation of an abstract data type.

Overview of OOP Concept

Classes and objects are the two main aspects of object-oriented programming.

Features of OOP

1 **Classes**

Syntax of Class

```
class className
{
    Access specifier:      // it can be public ,private or  protected
    Variable declaration;  //
    function declaration;
};
```

Example of Class

```
class student
{
    private:
    int rollNumber;
    string name;
    float grade;
            void print( ){
              cout<<“Hello”<<endl;
            }
            void myFunction( );
};
```

Overview of OOP Concept

Features of OOP

2 **Object**

An Object is the programming representation of the real-world entity.

When a class is defined no memory is allocated but when instantiated memory is allocated.

An **object** is an instance of a class. It represents a real-world entity with attributes (data) and behaviors (functions).

The Objects can be any one of the following:

a) External entities
b) Things
c) Occurrences or events
d) Roles
e) Organizational units
f) Places
g) Data Structures

Objects can have both attributes (data) and behaviors (functions or methods). Attributes describe the object with respect to certain parameters and behavior or functions to describe the functionality of the object.

Example of Object

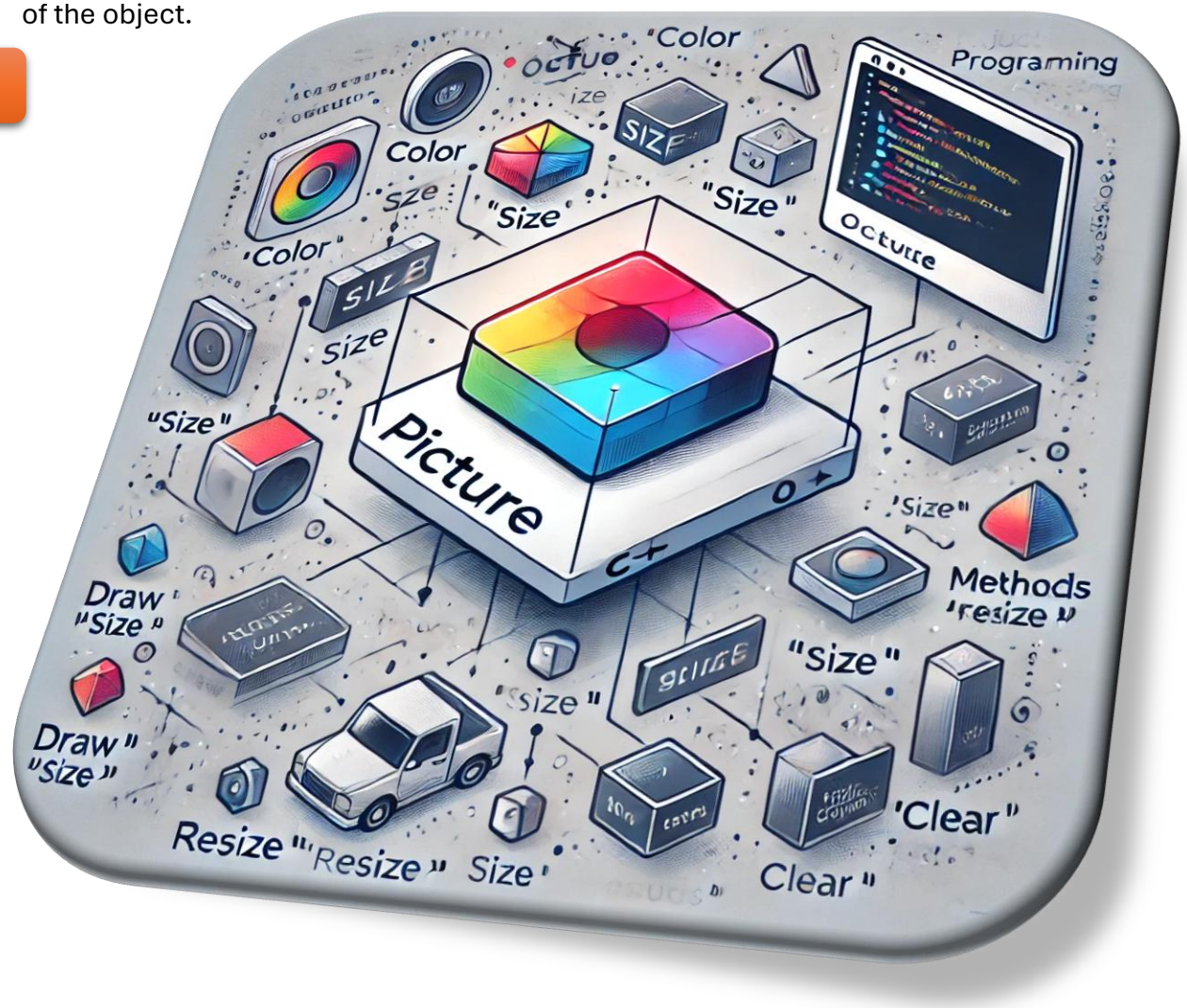
Polygon Object		Bank Account	
Attributes	Behavior	Attributes	Behavior
Position	Move	Account number	Deduct funds
Fill color	Erase	Balance	Transfer funds
Border color	Change color		Deposit funds
			Show balance

Overview of OOP Concept

Features of OOP

2 **Object**

Objects can have both attributes (data) and behaviors (functions or methods). Attributes describe the object with respect to certain parameters and behavior or functions to describe the functionality of the object.



The Objects can be any one of the following:

a) External entities
b) Things
c) Occurrences or events
d) Roles
e) Organizational units
f) Places
g) Data Structures

Creation of Class & Object

1 Code demo

classAndObject.cpp > ...

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

class myClassName {       // The class
  public:                // Access specifier
    int myNumber;          // Attribute (int variable)
    string myStringName;   // Attribute (string variable)
};

int main(){

    myClassName myObjectName;  // Create an object of myClassName

  // Access attributes and set values
  myObjectName.myNumber = 7;
  myObjectName.myStringName = "Santosh";

  // Print attribute values
  cout << myObjectName.myNumber << "\n";
  cout << myObjectName.myStringName;

    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS F:\Youtube Post\C++ course with Notes\C++ classes and Abstractions> g++ .\classAndObject.cpp
PS F:\Youtube Post\C++ course with Notes\C++ classes and Abstractions> .\a.exe
7
Santosh
PS F:\Youtube Post\C++ course with Notes\C++ classes and Abstractions>
```

Overview of OOP Concept

2 Code demo

class.cpp > ...

```cpp
#include<iostream>
using namespace std;
class student{
        //private:
        public:
        int rollNumber;
        string name;
        float grade;
    void print( ){
        cout<<"hello"<<endl;
    }
     void myFunction( int x,int y){
        cout<<x+y<<endl;
     }
};

int main(){
    student st;

        st.rollNumber=123;
        st.name="santosh";
        st.grade=9.67;
        cout<<st.rollNumber<<endl;
        cout<<st.name<<endl;
        cout<<st.grade<<endl;
        st.print();
        st.myFunction(7,8);

    return 0;
}
```

Output:
123
santosh
9.67
hello
15

Overview of OOP Concept

Class Methods

1. Inside class definition
2. Outside class definition

Methods are function that belongs to the class

class methods (also referred to as member functions) are functions that are defined within a class and operate on objects of that class

These methods typically manipulate or access the data members of the class and perform actions related to the class's behavior.

Key points:

A class method is declared inside a class definition and may be defined either within the class or outside the class.

Inside the class, methods are declared with their prototypes

Example:

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

class MyClass {
public:
    void display();  // Declaration of a class method
};

// Definition of the class method
void MyClass::display() {
    cout << "Hello from MyClass!" << endl;
}

int main() {
    MyClass obj;
    obj.display();  // Calling the class method
    return 0;
}
```

Thanks for watching

Please Subscribe