### Class

Class is a building block of an application.

In real world application consists of multiple classes each responsible for particular behavior in the application and overall all the class together will provide the behavior accepted from the application.

|  |
| --- |
|  |

**Real World Example of Classes**

|  |
| --- |
|  |

**Anatomy of a Class**

* Data (represented by fields).
* Behavior (represented by methods/functions).

**UML class diagram example**

UML is unified modeling language which is a graphical language represents the class and its collaboration.

|  |
| --- |
|  |

**Declaring Classes**

Naming convention of class name in C# we use Pascal case.

|  |
| --- |
|  |

### Object

Object is an instance of a class that resides in memory.

Person class defines the blue print from which we can create objects and at runtime this objects in the memory talk to each other, as a result get the behavior accepted from the application.

**Creating Objects**

|  |
| --- |
|  |

**Using Objects**

|  |
| --- |
|  |

|  |
| --- |
|  |

### Class Members

Two types of Members:-

1. Instance: accessible from an object.

|  |
| --- |
|  |

1. Static: accessible from the class.

Where Console is a class and WrileLine() is a static method.

|  |
| --- |
|  |

### Why use static members

To represent concepts that are singleton (means we should have one instance of that concept in the memory)

**Declaring Static Members**

|  |
| --- |
|  |

### Inheritance

If we have multiple classes that should have some common behavior and properties, instead of implementing the common behavior or properties in multiple places, define them once in parent or base class and those other classes inherit those behavior from their base class.

### Static Method

When a method is defined as static using static key word, it can be access directly using class name and don’t need to create an instance of the class.

|  |
| --- |
| 1. **class** Program 2. { 3. **public** **static** **void** withoutObj() 4. { 5. Console.WriteLine("Hello"); 6. } 7. **static void Main()** 8. { 9. Program. withoutObj(); 10. Console.ReadKey(); 11. } 12. } |

### Property

They look like a field from outside but internally they are like methods

### Interface

Interface is used to define the shape of object

### Design Patterns

1. **Singleton** – (we should have one instance of that concept in the memory)

Eg:- DateTime.Now, Console.Writeline()