Access Specifier	Access Level	Usage
`public`	Accessible **anywhere** in the project.	Used for global access.
`private`	Accessible **only inside the class.**	Used for data hiding.
`protected`	Accessible **within the class and derived (child) classes.**	Used for inheritance.
`internal`	Accessible **within the same assembly (project).**	Used to restrict access to the project.
`protected internal`	Accessible **within the same assembly and in derived classes.**	Combination of `protected` + `internal`.
`private protected`	Accessible **within the same class and derived classes in the same assembly.**	More restrictive than `protected internal`.

Constructor Type	Purpose
Default Constructor	Initializes with default values.
Parameterized Constructor	Allows passing values for initialization.
Copy Constructor	Creates a copy of another object.
Static Constructor	Initializes static members, runs once.
Private Constructor	Restricts object creation, used in Singleton.

Feature	Constructor	Method
Belongs to a Class	✅ Yes	✅ Yes
Can Have Parameters	✅ Yes	✅ Yes
Can Have Access Modifiers	✅ Yes (public, private, etc.)	✅ Yes
Can Be Overloaded	✅ Yes	✅ Yes

Feature	Fields	Properties
Definition	Variables declared inside a class.	Special methods (getters & setters) that control access to fields.
Encapsulation	No control over access.	Provides controlled access (read-only, write-only, validation, etc.).
Direct Access	Can be accessed directly (if public).	Accessed through getters and setters.
Flexibility	No logic can be applied.	Can have logic (e.g., validation, calculations).
Usage	Used for internal data storage.	Used to expose data safely.

Method	Description
`ToString()`	Returns a string representation of the object.
`Equals(object obj)`	Compares two objects for equality.
`GetHashCode()`	Returns a unique integer value for the object.
`GetType()`	Returns the runtime type of the object.
`MemberwiseClone()`	Creates a shallow copy of the object.

Feature	Object	Constructor
Definition	An instance of a class that allows access to its members.	A special method that initializes an object when it is created.
Creation	Created using the `new` keyword.	Called automatically when an object is created.
Purpose	Allows access to class methods and properties.	Initializes object properties and sets default values.
Return Type	Has no return type; it is just an instance.	No return type (not even `void`).
Calling	Explicitly created using `new ClassName();`.	Automatically invoked when an object is instantiated.
Types	No types—just an instance of a class.	Different types: Default, Parameterized, Copy, Static, Private.

Class Type	Can Be Instantiated?	Can Be Inherited?	Special Features
Abstract Class	❌ No	✅ Yes	Contains abstract and concrete methods
Sealed Class	✅ Yes	❌ No	Prevents inheritance
Static Class	❌ No	❌ No	Only contains static members
Partial Class	✅ Yes	✅ Yes	Defined across multiple files
Nested Class	✅ Yes	✅ Yes	Defined inside another class
Generic Class	✅ Yes	✅ Yes	Works with multiple data types
Concrete Class	✅ Yes	✅ Yes	A normal class with full implementation

Method	Description
`ToString()`	Returns a string representation of the object.
`Equals(object obj)`	Compares two objects for equality.
`GetHashCode()`	Returns a unique integer value for the object.
`GetType()`	Returns the runtime type of the object.
`MemberwiseClone()`	Creates a shallow copy of the object.