**Reflection:**

* Reflection is the feature that enables you to obtain information about a type.
* The term reflection comes from the way the process works: A Type object mirrors the underlying type that it represents.
* To obtain information, you ask the Type object questions, and it returns (reflects) the information associated with the type back to you.
* Reflection is a powerful mechanism because it allows you to learn and use the capabilities of types that are known only at runtime.
* Many of the classes that support reflection are part of the .NET Reflection API, which is in the System.Reflection namespace.
* Thus, you will normally include the following in programs that use reflection:

using System.Reflection;

The Reflection Core: System.Type

* System.Type is at the core of the reflection subsystem because it encapsulates a type.
* It contains many properties and methods that you will use to obtain information about a type at runtime. Type is derived from an abstract class called System.Reflection.MemberInfo.

**MemberInfo defines the following read-only properties:**

|  |  |
| --- | --- |
| Property | Description |
| Type DeclaringType | Obtains the type of the class or interface in which the  member is declared. |
| MemberTypesMemberType | Obtains the kind of the member. This value indicates if the  member is a field, method, property, event, or constructor. |
| intMetadataToken | Obtains a value associated with a specific metadata. |
| Module Module | Obtains a Module object that represents the module (an  executable file) in which the reflected type resides. |
| string Name | The name of the type. |
| Type ReflectedType | The type of the object being reflected. |

* Notice that the return type of MemberType is MemberTypes.
* MemberTypes is an enumeration that defines values that indicate the various member types. Among others, these include
* MemberTypes.Constructor
* MemberTypes.Method
* MemberTypes.Field
* MemberTypes.Event
* MemberTypes.Property
* Thus, the type of a member can be determined by checking MemberType.
* For example, if MemberType equals MemberTypes.Method, then that member is a method.
* MemberInfo includes two abstract methods: GetCustomAttributes( ) and IsDefined( ).
* These both relate to attributes. The first obtains a list of the custom attributes associated with the invoking object. The second determines if an attribute is defined for the invoking object.
* To the methods and properties defined by MemberInfo, Type adds a great many of its own.

For example, here are several commonly used methods defined by Type:

|  |  |
| --- | --- |
| ***Method*** | ***Purpose*** |
| ConstructorInfo[ ] GetConstructors( ) | Obtains a list of the constructors for the specified type. |
| EventInfo[ ] GetEvents( ) | Obtains a list of events for the specified type. |
| FieldInfo[ ] GetFields( ) | Obtains a list of the fields for the specified type |
| Type[ ] GetGenericArguments( ) | Obtains a list of the type arguments bound to a closed constructed generic type or the type parameters if the specified type is a generic type definition. For an open constructed type, the list may contain both type arguments and type parameters |
| MemberInfo[ ] GetMembers( ) | Obtains a list of the members for the specified type. |
| MethodInfo[ ] GetMethods( ) | Obtains a list of methods for the specified type. |
| PropertyInfo[ ] GetProperties( ) | Obtains a list of properties for the specified type. |

Here are several commonly used, read-only properties defined by Type:

|  |  |
| --- | --- |
| **Property** | **Purpose** |
| Assembly Assembly | Obtains the assembly for the specified type. |
| TypeAttributes Attributes | Obtains the attributes for the specified type. |
| Type BaseType | Obtains the immediate base type for the specified type. |
| string FullName | Obtains the complete name of the specified type |
| bool IsAbstract | Is true if the specified type is abstract. |
| bool isArray | Is true if the specified type is an array |
| bool IsClass | Is true if the specified type is a class. |
| bool IsEnum | Is true if the specified type is an enumeration |
| bool IsGenericParameter | Is true if the specified type is a generic type parameter |
| bool IsGenericType | Is true if the specified type is a generic type |
| string Namespace | Obtains the namespace of the specified type |