**Metadata in .NET**

Metadata in .NET is data that describes other data. It is embedded in assemblies (DLLs and EXEs) and contains essential information about the types, members, and references in a .NET application. Metadata is a fundamental part of the Common Language Runtime (CLR) and plays a crucial role in type safety, reflection, and interoperability.

**Key Components of Metadata**

Metadata in .NET includes the following elements:

1. **Assembly Metadata**
   * Contains information about the assembly, such as name, version, culture, and public key (for strong-named assemblies).
2. **Type Metadata**
   * Defines classes, structs, enums, delegates, and interfaces.
   * Stores details such as class names, visibility (public, private), and inheritance.
3. **Member Metadata**
   * Includes methods, properties, fields, events, and their attributes.
   * Stores method signatures, return types, parameters, and access modifiers.
4. **Custom Attributes**
   * Allows developers to attach additional metadata using attributes ([Attribute]).
   * Example: [Obsolete], [Serializable], [DllImport], etc.
5. **Security Information**
   * Defines security permissions and roles using .NET Code Access Security (CAS).
6. **Manifest**
   * Contains assembly-level metadata, including referenced assemblies and module definitions.

**How Metadata is Used in .NET**

1. **Reflection**
   * Metadata allows runtime inspection of types and objects using the System.Reflection namespace.
   * Example:
   * Type type = typeof(MyClass);
   * MethodInfo[] methods = type.GetMethods();
   * foreach (var method in methods)
   * {
   * Console.WriteLine(method.Name);
   * }
2. **Language Interoperability**
   * Metadata enables different .NET languages (C#, VB.NET, F#) to work together seamlessly.
3. **Code Compilation & Execution**
   * The CLR uses metadata to verify type safety, perform Just-In-Time (JIT) compilation, and execute the code.
4. **Serialization**
   * Metadata provides information about objects, enabling serialization and deserialization.
5. **IntelliSense & Documentation**
   * Metadata is used by IDEs (e.g., Visual Studio) to provide code completion, tooltips, and documentation.

**Metadata vs. Reflection**

* **Metadata** is stored in the assembly and describes types and members.
* **Reflection** is the process of reading metadata at runtime to inspect and manipulate objects dynamically.

**Viewing Metadata**

You can inspect metadata using:

1. **ILDASM (Intermediate Language Disassembler)**
   * Run ildasm.exe on a .NET assembly to view metadata and IL code.
2. **dotPeek / ILSpy (Decompilers)**
   * Tools that allow you to browse metadata and source code.

**Conclusion**

Metadata is a vital component of .NET that provides essential type information, supports reflection, and enables interoperability between different parts of a .NET application. Understanding metadata helps in debugging, performance tuning, and dynamic programming in .NET.