Runtime

29 January 2022 09:30

Delegate: It is a reference type which in-directs a call to a method with a particular list of parameter types and a particular return type.
A delegate object is actually an instance of a (compiler generated) class derived from System.MulticastDelegate and it has following characteristics

1. A delegate object can refer to one (through assignment) or more (through addition) methods whose return type and parameter types are compatible with those specified in the type of that delegate object.
2. The type of delegate object implements a compatible *Invoke* method, a call to which is equivalent to sequentially invoking the methods referred by that delegate object.

Event: It is a notification relayed by an object called an *event source* describing a certain change in its state to other objects called *event sinks* which are interested in taking some action when that change occurs. A .NET application can support events using following steps

1. In the event source class define an event member of System.EventHandler<E> delegate type where E is a class derived from System.EventArgs. Invoke this delegate to relay the event.
2. In the event sink class define a method compatible with above System.EventHandler<E> delegate type and add this method to the event member of the event source. When the event is relayed by the event source this method will be invoked.

Language INtegrated Queries (LINQ)**:** It is a feature of .NET runtime which enables a high-level language such as C# to provide syntactical support for querying different sources of data using data-source independent declarative statements.
LINQ specifies a set of *functional* (chainable or fluent) methods called *query operators* which can be applied to a compatible data-source. The BCL provides built-in implementations for query operators in form of extension methods defined in following static classes of System.Linq

namespace

1. **Enumerable** whose methods target *System.Collections.Generic.IEnumerable*<V> interface and pass query operations to the implementation of this interface in form of *delegates* (containing references to the instructions of those operations).
2. **Queryable** whose methods target *System.Linq.IQueryable*<V> interface and pass query operation to the LINQ provider exposed by the implementation of this interface in form of *expression-trees* (which can be decomposed into the instructions of those operations)

Reflection in .NET: Under CLR, the meta-data (type information) of any type is loaded into the memory when required as an instance of System.Type whose reference can be obtained using following methods:

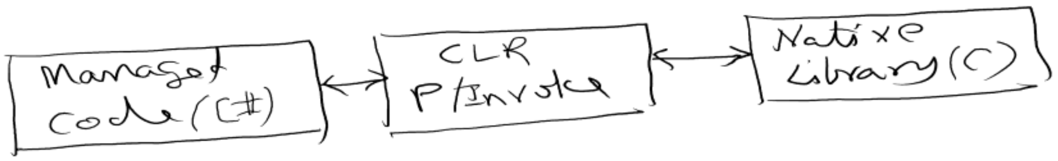
1. From the literal name of type T which is known at compile time
   ```
   Type t = typeof(T);
   ```
2. From an instance obj of the type
   ```
   Type t = obj.GetType();
   ```
3. From fully qualified name N (Namespace.Type,assembly) of the type which is discovered at runtime
   ```
   Type t = Type.GetType(N);
   ```

Attribute: It is a *programming language neutral* modifier which can be applied to a *declaration target* (such as class, field, property, method etc) in order to extend its meta-data. There are two types of attributes.

1. **Custom Attribute** which is inserted into the meta-data of its declaration target as an instance of a class derived from Sytstem.Attribute.
2. **Pseudo Attribute** which is inserted into the meta-data of its declaration target as a built-in IL modifier.

Platform Invocation (P/Invoke)**:** It is a mechanism built within the CLR that allows managed (IL) code to *invoke* unmanaged (native) functions exported by a *platform* specific library which supports *dynamic linking*. Platform invocation can be applied in C# using following steps:

1. Define an *extern static* method with System.Runtime.InteropServices.DllImportAttribute which identifies the native library along with its unmanaged entry-point function.
2. Call the above method and the CLR will automatically invoke the specified entry-point function performing required conversions between managed (C#) and unmanaged (C) data-types.


Managed code (C#)
CLR P/Invoke
Native Library (C)