what is delegates?

A Delegate is a type that references a method. Once a delegate is assigned a method, it behaves exactly like that method. The Delegate method can be used like any other method with parameter and return value".

Why delegates

Unlike c or C++ function pointer, delegates are,

1. Type safe
2. Object Oriented
3. Secure

Delegates have following properties,

1. Delegates are similar to C++ function pointer but it is type safe in nature.
2. Delegate allows method to pass as an argument.
3. Delegate can be chained together.
4. Multiple methods can be called on a single event.

**Important Points About Delegates:**

* Provides a good way to encapsulate the methods.
* Delegates are the library class in System namespace.
* These are the type-safe pointer of any method.
* Delegates are mainly used in implementing the call-back methods and events.
* Delegates can be chained together as two or more methods can be called on a single event.
* It doesn’t care about the class of the object that it references.
* Delegates can also be used in “anonymous methods” invocation.

. **Syntax:**

[modifier] delegate [return\_type] [delegate\_name] ([parameter\_list]);

*modifier: It is the required modifier which defines the access of delegate and it is optional to use.*

*delegate: It is the keyword which is used to define the delegate.*

*return\_type: It is the type of value returned by the methods which the delegate will be going to call. It can be void. A method must have the same return type as the delegate.*

*delegate\_name: It is the user-defined name or identifier for the delegate.*

*parameter\_list: This contains the parameters which are required by the method when called through the delegate.*

**Step 1 Creating a Delegate**

*Syntax*

*Access modifier delegate return type Delegate name ([arguments list]);*

*Eg*

1. Public delegate **void** Delegate (**int** x, **int** y);

When we create a delegate, Access modifier, return type, number of arguments, and their data types of the delegate must and should be the same as Access modifier, return type, number of arguments and their data types of the function that we want to refer to.

**Step 2 Instantiating the Delegate**

*Syntax*

*Delegate name object name = new Delegate name (Targe Function name);*

*Eg*

1. Delegate obj= **new** Delegate(Add);

What exactly we are doing in this step is that a reference will be maintained from the delegate object to the function that we want to refer to.

**Step 3 Invoking the Delegate**

*Syntax*

*Object Name ([Arguments Values])*

*Eg*

1. Obj(10,20)

At this step, the function that is referred by the delegate will be called for the execution. So now, I hope you are clear about what delegate is,when to use delegate, and steps to work with delegates.