Делегаты (Delegates) в C# - это тип данных, который представляет ссылку на метод. Они позволяют передавать методы как параметры, хранить методы в коллекциях, вызывать методы асинхронно и многое другое.

Синтаксис создания делегата:

csharp

Copy

delegate returnType delegateName(parameters);

Здесь:

* returnType - тип данных возвращаемого значения метода, который будет вызываться через делегат
* delegateName - имя делегата
* parameters - параметры метода, который будет вызываться через делегат

Пример создания делегата:

csharp

Copy

delegate int Calculate(int a, int b); // объявление делегата

В этом примере делегат называется Calculate и принимает два параметра типа int и возвращает значение типа int.

Создание экземпляра делегата:

csharp

Copy

Calculate calc = Add; // Создание экземпляра делегата и присваивание ему метода

В этом примере мы создаем экземпляр делегата calc и присваиваем ему метод Add.

Пример использования делегата:

csharp

Copy

public static int Add(int a, int b)

{

return a + b;

}

public static int Multiply(int a, int b)

{

return a \* b;

}

static void Main(string[] args)

{

Calculate calc = Add; // Создание экземпляра делегата и присваивание ему метода Add

int result = calc(10, 5); // Вызов делегата

Console.WriteLine(result); // Вывод на экран результата (15)

calc = Multiply; // Присваивание делегату метода Multiply

result = calc(10, 5); // Вызов делегата

Console.WriteLine(result); // Вывод на экран результата (50)

}

Здесь мы создаем экземпляр делегата calc и присваиваем ему метод Add. Затем мы вызываем делегат, передавая ему параметры 10 и 5, и сохраняем результат в переменной result. Затем мы выводим результат на экран. Затем мы присваиваем делегату метод Multiply и вызываем его с другими параметрами.

Делегаты также могут использоваться для определения анонимных методов:

csharp

Copy

Calculate calc = delegate (int a, int b) { return a + b; }; // Определение анонимного метода

int result = calc(10, 5); // Вызов делегата

Console.WriteLine(result); // Вывод на экран результата (15)

Здесь мы определяем анонимный метод через делегат calc и вызываем его с параметрами 10 и 5.

В C# 3.0 и выше версиях можно использовать лямбда-выражения для определения анонимных методов:

csharp

Copy

Calculate calc = (a, b) => a + b; // Определение анонимного метода через лямбда-выражение

int result = calc(10, 5); // Вызов делегата

Console.WriteLine(result); // Вывод на экран результата (15)

Здесь мы определяем анонимный метод через лямбда-выражение и вызываем его с параметрами 10 и 5.

Таким образом, делегаты позволяют передавать методы как параметры, хранить методы в коллекциях, вызывать методы асинхронно и многое другое, что делает их мощным инструментом для функционального программирования в C#.