C# Lambdas

Here are quick examples of how to create anonymous lambdas in C# method blocks.

With Delegate Definition

When the lambda needs to be based on a delegate type, here are examples.

This lambda implements a delegate type, accepting an int and returning a composite:

```
// Declare the delegate type...
public delegate (int res, string data)? dGetKey(string kid);

public void Example()
{
    // Declare the lambda implementation...
    dGetKey callback = (k) =>
    {
        // Do stuff...
        // Return its signature...
        return (1, null);
    };

    // You can invoke it, like this...
    var result = callback("myKeyId");
}
```

And, this delegate type has no return:

```
// Declare the delegate type...
public delegate dGetKey(string kid);

public void Example()
{
    // Declare the lambda implementation...
    dGetKey callback = (k) =>
```

```
    {
        // Do stuff...
    };

    // You can invoke it, like this...
    callback("myKeyId");
}
```

Without a Delegate Definition

When you don't have a defining delegate type, here is how you can create anonymous lambdas.

This lambda accepts an int, and returns a composite (int and string):

```
Func<string, (int res, string? data)> krcb = (k) =>
{
    return (1, null);
};
```

This lambda accepts a string, with no return (action vs func):

```
Action<string> testaction = (k) =>
{
  // Do stuff...
};
```

Revision #4
Created 22 August 2025 22:59:58 by glwhite
Updated 23 August 2025 18:09:25 by glwhite