

Шпаргалка по теме: C#. Лямбда-выражения

Лямбда-выражения — это краткий способ записи анонимных функций. Они широко используются в C# для упрощения кода, особенно при работе с делегатами, событиями и LINQ.

1. Базовый синтаксис лямбда-выражений

Лямбда-выражение состоит из:

- Входных параметров (если есть).
- Лямбда-оператора (\Rightarrow).
- Тела выражения (одна строка или блок кода).

Формы лямбда-выражений:

- Однострочное выражение:

```
(параметры) => выражение
```

- Блочное выражение:

```
(параметры) => { тело }
```

2. Примеры лямбда-выражений

Пример 1: Простое лямбда-выражение

```
Func<int, int> square = x => x * x;  
Console.WriteLine(square(5)); // Вывод: 25
```

Пример 2: Лямбда-выражение с несколькими параметрами

```
Func<int, int, int> add = (x, y) => x + y;  
Console.WriteLine(add(3, 4)); // Вывод: 7
```

Пример 3: Лямбда-выражение без параметров

```
Action greet = () => Console.WriteLine("Hello, World!");  
greet(); // Вывод: Hello, World!
```

Пример 4: Блочное лямбда-выражение

```
Func<int, int, int> multiply = (x, y) =>  
{  
    int result = x * y;  
    return result;  
};  
Console.WriteLine(multiply(3, 4)); // Вывод: 12
```

3. Лямбда-выражения и делегаты

Лямбда-выражения часто используются с делегатами, такими как Func, Action и Predicate.

Пример 1: Использование с Func

```
Func<int, int, int> sum = (a, b) => a + b;  
Console.WriteLine(sum(10, 20)); // Вывод: 30
```

Пример 2: Использование с Action

```
Action<string> print = message => Console.WriteLine(message);  
print("Привет, мир!"); // Вывод: Привет, мир!
```

Пример 3: Использование с Predicate

```
Predicate<int> isEven = num => num % 2 == 0;  
Console.WriteLine(isEven(4)); // Вывод: True
```

4. Лямбда-выражения и LINQ

Лямбда-выражения широко используются в LINQ для фильтрации, сортировки и преобразования данных.

Пример 1: Фильтрация списка

```
List<int> numbers = new List<int> { 1, 2, 3, 4, 5 };  
var evenNumbers = numbers.Where(n => n % 2 == 0).ToList();  
// evenNumbers: [2, 4]
```

Пример 2: Сортировка списка

```
List<string> names = new List<string> { "Анна", "Иван", "Мария" };  
var sortedNames = names.OrderBy(n => n).ToList();  
// sortedNames: ["Анна", "Иван", "Мария"]
```

Пример 3: Преобразование списка

```
List<int> numbers = new List<int> { 1, 2, 3, 4, 5 };  
var squaredNumbers = numbers.Select(n => n * n).ToList();  
// squaredNumbers: [1, 4, 9, 16, 25]
```

5. Лямбда-выражения и события

Лямбда-выражения можно использовать для подписки на события.

Пример:

```
button.Click += (sender, e) => MessageBox.Show("Кнопка нажата!");
```

6. Захват переменных в лямбда-выражениях

Лямбда-выражения могут захватывать переменные из окружающего контекста.

Пример:

```
int multiplier = 2;  
Func<int, int> multiply = x => x * multiplier;  
Console.WriteLine(multiply(5)); // Вывод: 10
```

7. Лямбда-выражения и асинхронные методы

Лямбда-выражения можно использовать с асинхронными методами.

Пример:

```
Func<Task> asyncAction = async () =>  
{  
    await Task.Delay(1000);  
    Console.WriteLine("Завершено через 1 секунду");  
};  
await asyncAction();
```

Итоговая таблица

Конструкция	Описание	Пример
Однострочное выражение	Краткая форма лямбда-выражения.	<code>x => x * x</code>
Блочное выражение	Лямбда-выражение с несколькими строками.	<code>(x, y) => { return x + y; }</code>
C Func	Лямбда-выражение для методов с возвращаемым значением.	<code>Func<int, int> square = x => x * x;</code>
C Action	Лямбда-выражение для методов без возвращаемого значения.	<code>Action<string> print = message => Console.WriteLine(message);</code>
C Predicate	Лямбда-выражение для проверки условия.	<code>Predicate<int> isEven = num => num % 2 == 0;</code>
В LINQ	Использование лямбда-выражений для фильтрации, сортировки и проекции.	<code>numbers.Where(n => n % 2 == 0).ToList();</code>
Захват переменных	Лямбда-выражения могут использовать переменные из внешнего контекста.	<code>int multiplier = 2; Func<int, int> multiply = x => x * multiplier;</code>
Асинхронные лямбды	Лямбда-выражения с асинхронными методами.	<code>Func<Task> asyncAction = async () => { await Task.Delay(1000); };</code>