# Day 53/60 - Must-Know LINQ Methods Every C# Developer Should Master

LINQ (Language Integrated Query) is a powerful feature in C# that simplifies data manipulation. Knowing the right LINQ methods can improve performance and readability. Here are essential LINQ methods every developer must know:

## Where() - Filtering Data

Filters elements based on a condition.

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

Best for: Filtering lists efficiently.

## Select() - Transforming Data

Projects each element into a new form.

```
var names = new List<string> { "John", "Alice" };
var upperNames = names.Select(n => n.ToUpper());
// Output: JOHN, ALICE
```

Best for: Converting objects or modifying properties.

## OrderBy() and OrderByDescending() - Sorting Data

Sorts elements in ascending (OrderBy) or descending (OrderByDescending) order.

```
var sorted = numbers.OrderBy(n => n);
// Output: 1, 2, 3, 4, 5
```

Best for: Sorting collections easily.

## GroupBy() - Grouping Data

Groups elements based on a key.

```
var grouped = names.GroupBy(n => n.Length);
// Groups names by length
```

Best for: Aggregating data by categories.

## FirstOrDefault() vs SingleOrDefault() - Retrieving Elements

FirstOrDefault() returns the first element or null if none exist.
SingleOrDefault() ensures only one element exists, otherwise throws an error.

```
var result = numbers.FirstOrDefault(n => n > 3);
// Output: 4
```

Best for: Fetching specific elements safely.

Any() and All() - Boolean Checks

✅ Any() checks if at least one element matches a condition.
✅ All() checks if all elements match a condition.

```
bool hasEven = numbers.Any(n => n % 2 == 0); // true
bool allPositive = numbers.All(n => n > 0); // true
```

Best for: Quick validation of lists.

Aggregate() - Custom Aggregation

✅ Reduces a collection into a single value.

```
int sum = numbers.Aggregate((a, b) => a + b);
// Output: 15 (sum of all numbers)
```

Best for: Custom calculations like sums, concatenations, or combining elements.

Final Thoughts

Mastering LINQ improves your ability to write efficient, readable, and maintainable C# code.

Which LINQ method do you use the most? Drop a comment!

#dotnet #csharp #linq #codingtips #performance