

40

LINQ Methods

With One-line Descriptions



Keivan Damirchi

Intersect

Returns the **common** elements between two sequences.

OfType

Filters a sequence to only include elements of a specified type.

Except

Returns the elements in the first sequence that are **not in the second** sequence.

Repeat

Generates a sequence that **repeats** a specified value a specified number of times.

ThenBy

Performs a **secondary** sort on a sequence based on a key.

ToLookup

Creates a **lookup** table from a sequence based on a key.

ElementAt

Returns the element at a **specified index** in a sequence.

Range

Generates a sequence of integers within a **specified range**.

Distinct

Returns the **unique** elements in a sequence.

Empty

Returns an **empty** sequence of a specified type.

GroupJoin

Performs a **left outer join** on two sequences based on a key and groups the results.

DefaultIfEmpty

Returns a **default value** if a sequence is empty.

Where

Filters a sequence based on a **condition**.

Union

Returns the **distinct** elements from two sequences.

Select

Projects each element of a sequence into a **new form**.

Count

Returns the **number of elements** in a sequence.

OrderBy

Sorts a sequence in ascending order based on a key.

Concat

Concatenates two sequences.

First

Returns the first element of a sequence.

ElementAtOrDefault

Returns the element at a specified index in a sequence or a default value if the index is out of range.

FirstOrDefault

Returns the **first element** of a sequence or a **default value** if the sequence is empty.

Last

Returns the **last element** of a sequence.

LastOrDefault

Returns the **last element** of a sequence or a **default value** if the sequence is empty.

Single

Returns the **only element** of a sequence, or throws an exception if there is **not exactly one element**.

SingleOrDefault

Returns the **only element** of a sequence, or a **default value** if the sequence is empty or if there is not exactly one element.

Take

Returns a **specified number** of elements from the start of a sequence.

TakeWhile

Returns elements from a sequence while a **condition** is true.

Skip

Skips a **specified number** of elements in a sequence.

SkipWhile

Skips elements in a sequence while a **condition** is true.

Sum

Computes the **sum of a sequence** of numeric values.

ThenByDescending

Performs a **secondary sort** in descending order based on a key.

Min, Max

Returns the **minimum** or **maximum** value in a sequence of numeric values.

ToDictionary

Creates a dictionary from a sequence of key-value pairs.

Any

Determines whether any element of a sequence satisfies a condition.

OrderByDescending

OrderByDescending method is used to sort the data in Descending order.

Reverse

Reverse method is used to reverse the data stored in a data source.

All

Determines whether all elements of a sequence satisfy a **condition**.

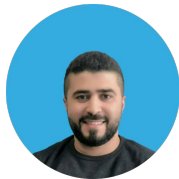
Average

Computes the **average** of a sequence of numeric values.

Contains

Contains Method is used to **check** whether a sequence or collection (i.e. data source) contains a **specified element** or not.

Close();



Keivan Damirchi