Introduction to .NET

Florin Olariu

“Alexandru Ioan Cuza”, University of Iași

Department of Computer Science

Agenda

- Generic dictionaries
- LINQ
- Interview questions

Generic dictionaries

Generic dictionaries

- What is a dictionary?
- How can we manipulate a dictionary?

Generic dictionaries

- What is a dictionary?

It is a data structure that enables us to access an element based on a specified key.

Generic dictionaries

- What is a dictionary?

It is a data structure that enables us to access an element based on a specified key.

It is a strongly typed collection of keys and values.

Generic dictionaries

- What is a dictionary?

It is a data structure that enables us to access an element based on a specified key.

It is a strongly typed collection of keys and values.

- ***Key***

Generic dictionaries

- What is a dictionary?

It is a data structure that enables us to access an element based on a specified key.

It is a strongly typed collection of keys and values.

- *Key*
  - *Must be unique*

Generic dictionaries

- What is a dictionary?

It is a data structure that enables us to access an element based on a specified key.

It is a strongly typed collection of keys and values.

- ***Key***
  - *Must be unique*
  - *Must not be changed*

Generic dictionaries

- What is a dictionary?

It is a data structure that enables us to access an element based on a specified key.

It is a strongly typed collection of keys and values.

- ***Key***
  - *Must be unique*
  - *Must not be changed*
  - *Cannot be null*

Generic dictionaries

- Declaration

Generic dictionaries

- Declaration

```
Dictionary<TKey, TValue>
```

Generic dictionaries

- Declaration

```
Dictionary<TKey, TValue>
```

- Samples

```
Dictionary<int, int>
Dictionary<int, string>
Dictionary<string, Product>
```

Generic dictionaries

- Initialization

```
Dictionary<string, string> states;
states = new Dictionary<string, string>();

Dictionary<string, string> states = new Dictionary<string, string>();

var states = new Dictionary<string, string>();
```

Generic dictionaries

- Manipulating a dictionary

```
states.Add("NY", "New York");

 var states = new Dictionary<string, string>
     {
         {"NY", "New York"},
         { "CA", "California"}
     };
  states.Remove("CA");
```

Generic dictionaries

- Performance

Generic dictionaries

- Performance
  - Many collection classes offer the same functionality as others; for example, SortedList offers nearly the same features as SortedDictionary.

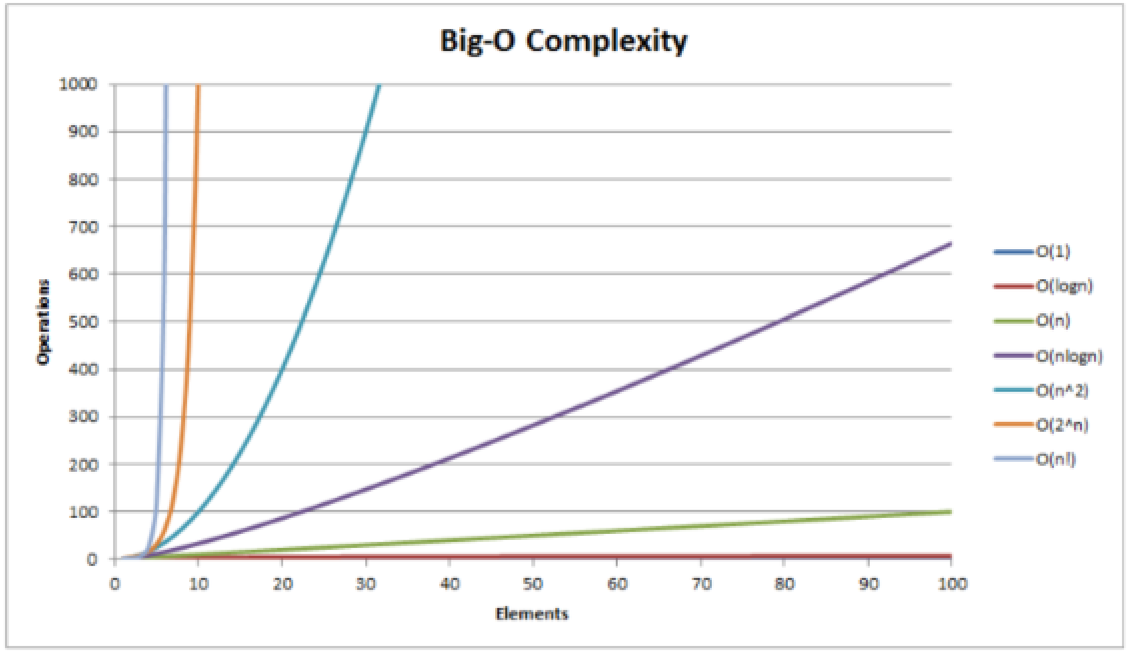
Generic dictionaries

- Performance
  - Many collection classes offer the same functionality as others; for example, SortedList offers nearly the same features as SortedDictionary.
  - However, often there’s a big difference in performance. Whereas one collection consumes less memory, the other collection class is faster with retrieval of elements.

Generic dictionaries

- Details about big O algorithm complexity in attached pdf for the course.

Big-O Complexity Chart



LINQ

LINQ

- Intro
- Building a LINQ query using Query syntax - Demo
- Building a LINQ query using Method syntax - Demo
- Lambda expression in action - Demo
- Using LINQ with collections

LINQ

- Stands from **L**anguage **IN**tegrated **Q**uery

LINQ

- Stands from **L**anguage **IN**tegrated **Q**uery
- Definition

LINQ

- Stands from **L**anguage **IN**tegrated **Q**uery
- Definition
  - ***A way to execute queries against a data source directly from .NET***

LINQ

- Stands from **L**anguage **IN**tegrated **Q**uery
- Definition
  - ***A way to execute queries against a data source directly from .NET***
- Data sources :
  - LINQ to objects => should implement an IEnumerable interface
  - LINQ to SQL => works with SQL databases
  - LINQ with Entities => works with Entity Framework
  - LINQ to XML => works with any XML Document
  - …

LINQ

- There are 2 ways to express queries in LINQ:

LINQ

- There are 2 ways to express queries in LINQ:
  - Query syntax:

LINQ

- There are 2 ways to express queries in LINQ:
  - Query syntax:

```
var product = from p in products
                     where p.Name == productName
                     select p;
```

  - Method syntax

LINQ

- There are 2 ways to express queries in LINQ:
  - Query syntax:

```
var product = from p in products
              where p.Name == productName
              select p;
```

  - Method syntax

```
var product = products.Where(p => p.Name ==
productName).FirstOrDefault();
```

or

LINQ

- There are 2 ways to express queries in LINQ:
  - Query syntax:

```
var product = from p in products
                          where p.Name == productName
                          select p;
```

  - Method syntax

```
var product = products.Where(p => p.Name ==
productName).FirstOrDefault();
```

or

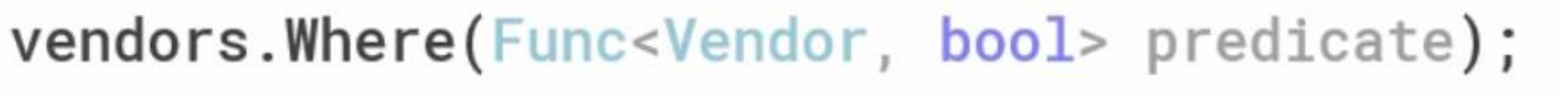
```
var product = products.FirstOrDefault(p => p.Name ==
productName);
```

LINQ

- Delegate

LINQ

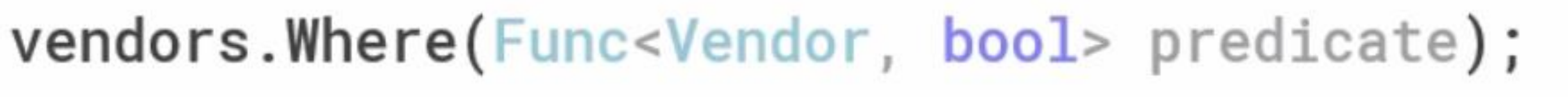
- Delegate
  - *Is a type that represents a reference to a method with a specific parameter list and a return type.*



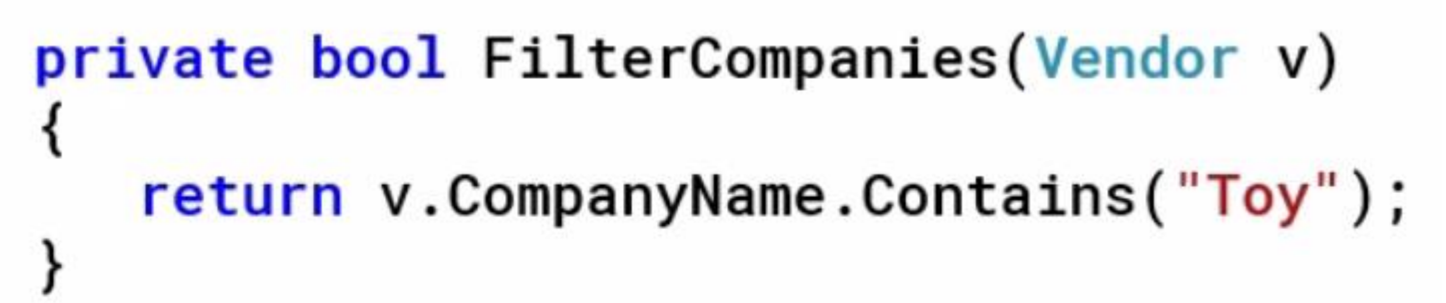
```
vendors.Where(Func<Vendor, bool> predicate);
```

LINQ

- Delegate
  - *Is a type that represents a reference to a method with a specific parameter list and a return type.*



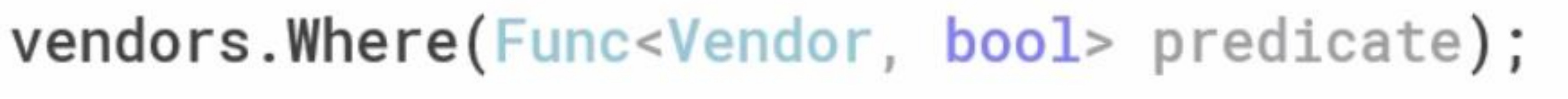
```
vendors.Where(Func<Vendor, bool> predicate);
```



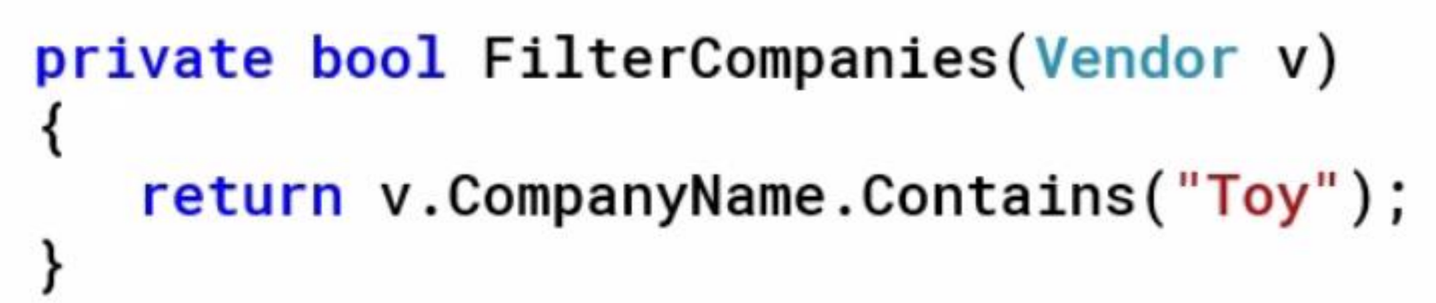
```
private bool FilterCompanies(Vendor v)
{
    return v.CompanyName.Contains("Toy");
}
```

LINQ

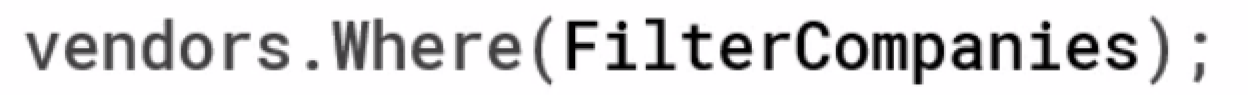
- Delegate
  - *Is a type that represents a reference to a method with a specific parameter list and a return type.*



```
vendors.Where(Func<Vendor, bool> predicate);
```



```
private bool FilterCompanies(Vendor v)
{
    return v.CompanyName.Contains("Toy");
}
```



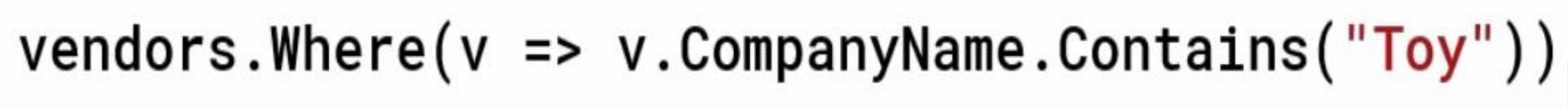
```
vendors.Where(FilterCompanies);
```

LINQ

- Lambda expression

LINQ

- Lambda expression
  - *Is a method that can be passed as an argument to a method when that argument is expecting a delegate type.*



```
vendors.Where(v => v.CompanyName.Contains("Toy"))
```

LINQ

- Demo – Query syntax vs Method syntax and Lambda expressions

LINQ

- LINQ with collections

DO	AVOID
Use LINQ!	Iterating the collections multiple times
Consider using method syntax over query syntax	Use FirstOrDefault and LastOrDefault instead of First and Last
Wait to cast a result after all queries are defined	

What’s next …

- Entity Framework Core

Interview questions

One more thing…

- *“Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live”*
  — John Woods

Questions

- Do you have any other questions?

Thanks!

See you next time! ☺