

# Day 8

## Anonymous object

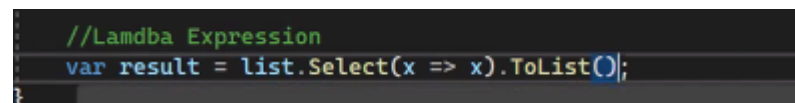
objects you can create without a previous template

```
var person = new {  
    Name = "John",  
    Age = 30,  
    Occupation = "Software Engineer"  
};
```

Note that anonymous objects are read-only and cannot be modified after they are created. They are also limited in their scope, since they can only be used within the method or code block in which they are created.

## Anonymous Functions

Lambda expressions



```
//Lambda Expression  
var result = list.Select(x => x).ToList();
```

Delegate inference

```
Func<int, int, int> add = delegate(int x, int y)  
{  
    return x + y;  
};
```

Action

Action is a delegate type that represents a method with no return value and zero or more input parameters.

```
Action<string> printMessage = message => Console.WriteLine(message);
```

Predicate

Predicate is a delegate type that represents a method that takes an input parameter and returns a boolean value.

```
Predicate<int> isEven = number => number % 2 == 0;
```

Func

Func is a delegate type that represents a method with a return value and zero or more input parameters.

In-memory objects: data/memory that is stored in Collection

/List, Dictionary, Arrays can use Linq because the output of Linq is IEnumerable

out-memory: sources outside scope of code, can be from text, Json, databases...

Linq can be still used on these; however, the output will be IQueryable.

LINQ Providers:

LINQ to Objects: In memory Sources

LINQ to Entities: processing data stored in External sources/database

LINQ to XML: old and not common practice

ORMs: Object Relational Mapping: connect the Asp.net application to SOL Server without any "Impedance Mismatch"

Map means the conversion of object types .

ORMs need provider: they are the connection that will connect your LINO to SOL server

```
class Movies{
    int id{get; set;}
    List<int> GenreIds {get;set;}
}
class Genres{
    int id {get;set;}
    List<int> MovieIds {get;set;}
}
```

```

0 references
public class EmployeeRepository
{
    //var DbContext = new DbContext();
    0 references
    public List<Employee> GetEmployees()
    {
        return new List<Employee>()
        {
            new Employee() {Id = 1, Name = "Fred", Age= 50,
                Department= "It"},
            new Employee() {Id = 2, Name = "Jacinda", Age= 51,
                Department= "Sales"},
            new Employee() {Id = 3, Name = "Anderson", Age= 52,
                Department= "HR"},
            new Employee() {Id = 4, Name = "Jack", Age= 53,
                Department= "Marketing"},
            new Employee() {Id = 5, Name = "Quan", Age= 54,
                Department= "It"}
        };
    }
}

```

SRP: Single Responsibility of Principal:

one class should have one responsibility over something

Data -> Repositories -> Service (Business Logic)

Readonly vs const: readonly doesn't have to be instantiated after declaration and can be done later

```

//Select Id, Name from Employee
var resultQSelect1 = from emp in list
                    select new
                    {
                        Id = emp.Id,
                        Name = emp.Name
                    };

```