

# C# Essential

Анонимные и динамические типы. LINQ.

# C# Essential

После урока обязательно



Повторите этот урок в видео формате на [ITVDN.com](http://itvdn.com)



Проверьте как Вы усвоили данный материал  
на [TestProvider.com](http://TestProvider.com)

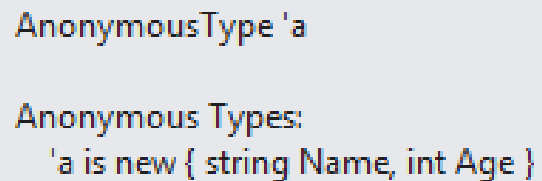
# Анонимные и динамические типы. LINQ.

# АНОНИМНЫЕ ТИПЫ

## Anonymous type

Анонимные типы предлагают удобный способ инкапсуляции набора свойств в один объект без необходимости предварительного явного определения типа.

```
var instance = new { Name = "Alex", Age = 27 };
```



AnonymousType 'a

Anonymous Types:  
'a is new { string Name, int Age }

**Имя типа создается компилятором и недоступно на уровне исходного кода.**

# Анонимные типы

## Использование анонимных типов

```
var instance = new { Name = "Alex", Age = 27, Id = new { Number = 123 } };
```

**В анонимных типах могут быть вложенные анонимные типы**



Анонимные типы являются ссылочными типами, которые происходят непосредственно от класса `object`.

# LINQ

## Language Integrated Query

**Language Integrated Query (LINQ)** – проект Microsoft по добавлению синтаксиса языка запросов, напоминающего SQL, в языки программирования платформы .NET Framework.

**Все операции запроса LINQ состоят из трех различных действий:**

- Получение источника данных.
- Создание запроса.
- Выполнение запроса.



**LINQ** – представляет стандартные, легко изучаемые шаблоны для создания запросов и обновления данных; технология может быть расширена для поддержки потенциально любого типа хранилища данных.

## Выражения запроса

**from** – задает источник данных.

**group** – используется для получения последовательности групп, организованной на основе указанного ключа.

**select** – для получения всех других типов последовательностей.

**into** – можно использовать в предложении **select** или **group** для создания временного идентификатора, в котором хранится запрос.

**orderby** – сортирует результаты в порядке возрастания или убывания.

**where** – используется для фильтрации элементов из источника данных по одному или нескольким выражениям предиката.



Выражение запроса должно начинаться предложением **from** и оканчиваться предложением **select** или **group**.

# АНОНИМНЫЕ ТИПЫ

## LINQ

Анонимные типы обычно используются в предложении **select** выражения запроса для возврата подмножества свойств из каждого объекта в исходной последовательности.

```
var query = from x in numbers
            select new { Input = x, Output = x * 2 };
```



**LINQ (Language-Integrated Query)** – является революционной инновацией в **.NET Framework** версии **3.5**, которая является мостом между миром объектов и миром данных.



# LINQ

## Запросы LINQ

```
var query =  
    employees  
    .Where(emp => emp.Salary > 100000)  
    .OrderBy(emp => emp.LastName)  
    .OrderBy(emp => emp.FirstName)  
    .Select(emp => new  
    {  
        LastName = emp.LastName,  
        FirstName = emp.FirstName  
    });
```

Любой LINQ-запрос, трансформируется в последовательность вызовов расширяющих методов.

# LINQ

## Запросы LINQ

**let** – представляет новый локальный идентификатор, на который можно ссылаться в остальной части запроса.

Его можно представить, как локальную переменную видимую только внутри выражения запроса.

```
var query = from emp in employees
             let fullName = emp.FirstName + " " + emp.LastName
             orderby fullName descending
             select fullName;

foreach (var person in query)
    Console.WriteLine(person);
```



Конструкция **from** похожа на оператор **foreach**. LINQ-запрос выполнится при обращении к нему.

# Динамический тип

## dynamic

```
static void Main()
{
    dynamic variable = 1;

    variable = "Hello world!";

    variable = DateTime.Now;
}
```

"События" – не могут  
быть типом **dynamic**

В C# 4.0 появился новый тип – **dynamic**. Тип является статическим типом, но объект типа **dynamic** обходит проверку статического типа.

В большинстве случаев он функционирует, как тип **object**.

Во время компиляции предполагается, что элементы с типом **dynamic** поддерживают любые операции.

Разработчику не нужно следить за тем, откуда объект получает свое значение.



В отличие от ключевого слова **var**, объект, объявленный как **dynamic**, может менять тип во время выполнения.

# C# Essential

Q&A

# Информационный видеосервис для разработчиков программного обеспечения

