

C# - Ultimate Guide - Beginner to Advanced | Master class

Section 23 – Anonymous Types

Anonymous Types

When you create an object with a set of properties along with values; automatically C# compiler creates a class (with a random name) with specified properties. It is called as 'anonymous type' or 'anonymous classes'.

Useful when you want to quickly create a class that contains a specific set of properties.

Creating Anonymous Object (based on anonymous type)

```
var referenceVariable = new { Property1 = value, Property2 = value, ... };
```

anonymous type [automatically generated]

```
class RandomClassName
{
    public type Property1 { get; set; }
    public type Property2 { get; set; }
}
```

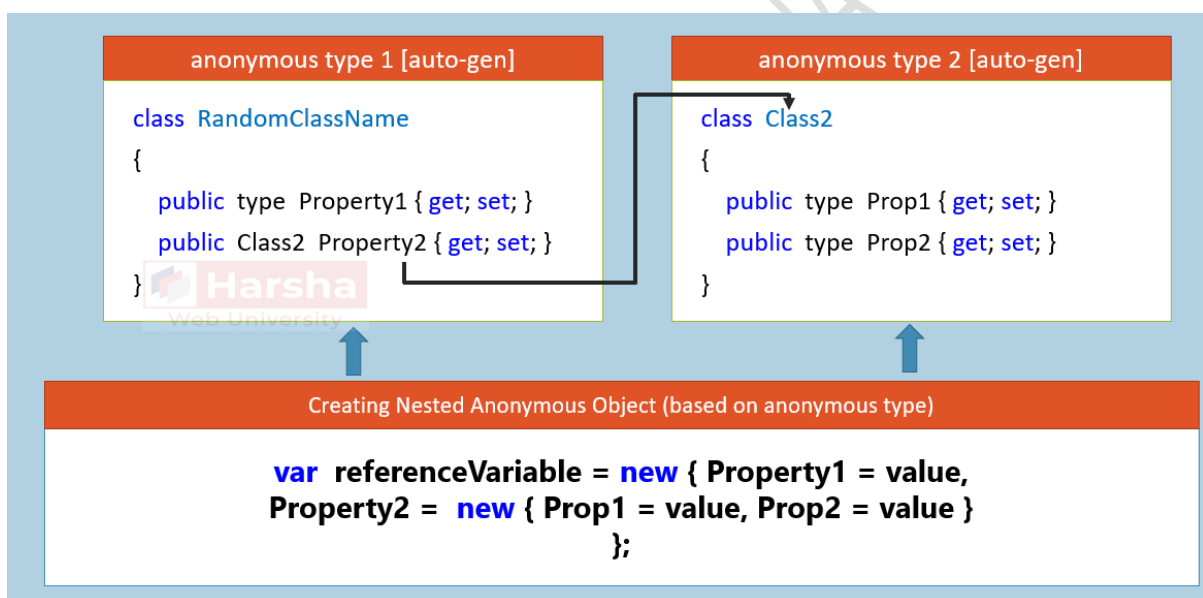
- Anonymous types are created by the C# compiler automatically at compilation time.
- The data types of properties of anonymous types will be automatically taken based on the value assigned into the property.
- Anonymous types are derived from System.Object class directly.
- Anonymous types are by default sealed class.
- Properties of anonymous types are by default readonly properties.
- Anonymous types can't contain other members such as normal properties, events, methods, fields etc.
- Properties of anonymous types will be always 'public' and 'readonly'.
- You can't add additional members of anonymous types once after compiler creates it automatically.
- 'null' can't be assigned into property of anonymous type.
- The data type of anonymous objects are always given as "var".

- Anonymous types can't be casted to any other type, except into System.Object type.
- You can't create a field, property, event or return type of a method, parameter type as of anonymous type.
- It is recommended to use the anonymous objects within the same method, in which they are created. You can pass anonymous type object to method as parameter as 'System.Object' type; but it's not recommended.

Nested Anonymous Types

You can nest an anonymous object into another.

Then two anonymous types will be created.



Anonymous Arrays

You can create 'array of anonymous objects' or 'implicitly typed array' with group of anonymous objects.

All objects must contain same set of properties.

Creating anonymous array / Implicitly typed array:

```
var referenceVariable = new []  
{  
    new { Property1 = value, Property2 = value2, ... },  
    new { Property1 = value, Property2 = value2, ... },  
    ...  
};
```

anonymous type [automatically generated]

```
class RandomClassName  
{  
    public type Property1 { get; set; }  
    public type Property2 { get; set; }  
}
```