الأعضاء الساكنة وفضاءات الأسماء
Static Members and Namespaces

الكلمات المفتاحية

الحقول الساكنة، الطرائق الساكنة، الصف الساكن Math، فضاء الأسماء.

ملخص الفصل

خُصّص هذا الفصل لتوضيح مفهوم الأعضاء الساكنة وفائدتها وكيفية التصريح عنها واستخدامها، واحتوى الفصل على شرح لمفهوم فضاءات الأسماء والفائدة منها وكيفية إنشاء فضاءات أسماء متداخلة.

الأهداف التعليمية

يتعرّف الطالب في هذا الفصل على:

- الحقل الساكن وكيفية استخدامه
- الطريقة الساكنة وكيفية استدعائها
- الكلمة المفتاحية this
- الباني الساكن والفائدة منه
- الصف الساكن Math ودوره في العمليّات الرياضية
- فضاءات الأسماء والفائدة من استخدامها

1. الحقول الساكنة والطرائق الساكنة

في لغة C#، يمكن التصريح عن الحقول والخصائص والطرائق والبناة بأنّها ساكنة، ويمكن للصفوف أيضاً أن تكون ساكنة.

يكون العنصر ساكناً إذا تمّ التصريح عنه بذلك من خلال استخدام الكلمة المفتاحية static. ومعنى ذلك أنّ العنصر يتبع للصف المعرّف ضمنه ولا يخصّ غرضاً محدّداً من أغراض الصف. وعندها يمكن استخدام العنصر مباشرة من دون الحاجة إلى إنشاء غرض من الصف الذي ينتمي إليه، ويتمّ استخدام العنصر مرفقاً باسم الصف متبوعاً بالمحرف (.).

1.1. الحقول الساكنة

عادة ما تُستخدَم الحقول الساكنة لتعريف ثوابت، حيث يمكن استخدامها مباشرة مع اسم الصف من دون الحاجة إلى إنشاء غرضٍ من الصف لاستدعاء قيمة الثابت، ولكن يمكن أن يُستخدم الحقل الساكن أيضاً كمتغيّر خاصّ بالصف. وعند إنشاء أغراضٍ من هذا الصف، يمكن لأي غرض قراءة نفس قيمة الحقل الساكن (أو الخاصية الساكنة)، أي لا توجد سوى نسخة واحدة من الحقل الساكن (أو الخاصية الساكنة). وعند إجراء أي تعديل على الحقل الساكن (أو الخاصية الساكنة) ستتمّ ملاحظة هذا التغيير من قبل جميع أغراض الصف، أي أنّ قيمة الحقل الساكن (أو الخاصية الساكنة) ستتمّ مشاركتها مع جميع أغراض الصف.

مثال:

في الرمّاز التالي، تمّ تعريف الصف Person لتوصيف مجموعة من الأشخاص حيث يضمّ:

- الحقل الساكن instances الذي تسند إليه القيمة (0) والمعبّر عن عدد الأغراض (المنتسخات) التي تمّ إنشاؤها من الصف
- الحقل name المعبِّر عن اسم الشخص والذي تُرك من دون تهيئة.
- الباني Person الذي يقوم بإسناد قيمة مدخلة للحقل name ويزيد قيمة الحقل instances بمقدار (1) عند إنشاء غرض جديد من الصف. أي أنّ الحقل instances يشير إلى عدد الأغراض التي يتمّ إنشاؤها من الصف.

كما تم تعريف Tester عن طريق إنشاء الطريقة Main كما يلي:

- إنشاء ثلاثة أغراض من Person تحمل الأسماء "Ahmad" و"Ziad" و"Reem" على الترتيب.
- إسناد القيمة (Univ.) إلى الخاصية الساكنة Organisation مسبوقة باسم الصف.
- طلب طباعة عدد الأغراض باستدعاء قيمة المتحوّل الساكن instances مسبوقة باسم الصف.
- طلب طباعة عدد الأغراض مرّة أخرى باستدعاء الطريقة الساكنة WritePersonNumber مسبوقة باسم الصف.
- طلب طباعة قيمة الخاصية الساكنة Organisation مسبوقة باسم الصف.

- طلب طباعة عدد الأغراض من خلال استدعاء الطريقة غير الساكنة GetInstances مع كل غرض من الأغراض الثلاثة.

```
using System;
namespace StaticMembers{
    class Person{
        public static int instances = 0; // static field, it is public for testing
purpose only,
        private string name;
        public static string Organisation { get; set; }  // static property
        public int GetInstances(){
            return instances;
        }// end GetInstances

        public static void WritePersonNumber(){
            Console.WriteLine("Using WritePersonNumber, Person Counter is : {0} ",
instances);
        }// end WritePersonNumber
         // constructor
        public Person(string name){
            instances++;
            this.name = name;
        }//end constructor

        public string Name{
            get { return name; }
            set { name = value; }
        }// end Name
    }// end class Person
    class PersonTest{
        static void Main(){
            Person person1 = new Person("Ahmad");
            Person person2 = new Person("Ziad");
            Person person3 = new Person("Reem");
            Person.Organisation = "Univ.";
            // using the class name (Person1) to call the value of instances
            Console.WriteLine("Using instances, Person Counter is : {0} ",
Person.instances);
            // if you use the first object (Person1) to call the value of
```

```
            // instances like this:
            //  Console.WriteLine("Person Counter is : {0} ", person.instances);
            // you will get the error message:
            // "Member 'Person.instances' cannot be accessed with an instance
            //  reference qualify it with a type name instead
            Person.WritePersonNumber();
            //   person1.WritePersonNumber();// Not accepted
            Console.WriteLine("Person Organisation is : {0} ",
Person.Organisation);
            Console.WriteLine("Person1 Counter is : {0} ",
person1.GetInstances());
            Console.WriteLine("Person2 Counter is : {0} ",
person2.GetInstances());
            Console.WriteLine("Person3 Counter is : {0} ",
person3.GetInstances());
            //  Console.WriteLine("Person Counter is : {0} ",
Person.GetInstances());
            // Not accepted: you will get the error message: “an object reference
is
            // required for the non-static field, method
            //or property 'Person.GetInstances()”
            Console.ReadKey();
        }// end Main
    }// end class PersonTest
} // end namespace StaticMembers
```

- عند التنفيذ نحصل على الخرج التالي:

```
Using instances, Person Counter is : 3
Using WritePersonNumber, Person Counter is : 3
Person Organisation is : Univ.
Person1 Counter is : 3
Person2 Counter is : 3
Person3 Counter is : 3
```

نلاحظ أنّ قيمة الحقل الساكن instances المعبّر عن عدد الأغراض هي (3)، وجميع الأغراض الثلاثة التي تمّ إنشاؤها لديها نفس القيمة، وهي نفس القيمة المتحصل عليها عند استدعائها مباشرة من الصف باستخدام أعضائه الساكنة. أي أنّه تمّ التعامل مع الحقل الساكن، في هذه الحالة، وكأنّه متغيّر عام لجميع الأغراض.

نلاحظ أيضاً أنّه إذا استخدمنا أحد الأغراض (مثلاً person1) مباشرة لاستدعاء قيمة الحقل الساكن instances، سنحصل على رسالة الخطأ التالية:

```
Member ‘Person.instances’cannot be accessed with an instance
reference; qualify it with a type name instead
```

والتي تفيد بأنّه يجب استخدام اسم نمط (أي اسم الصف) لاستدعاء الحقل الساكن.
ويمكن استخدام أحد الأغراض لاستدعاء الطريقة غير الساكنة GetInstances التي تعيد قيمة الحقل الساكن instances، أي أنّه يمكن لطريقة غير ساكنة استخدام عنصر ساكن.

2.1. الطرائق الساكنة

يتمّ التصريح عن طريقة ما بأنها ساكنة باستخدام الكلمة المفتاحية static، ويجب على الطريقة الساكنة استخدام أعضاءٍ ساكنةٍ فقط، ولا يمكنها استخدام الكلمة المفتاحية this لأنها تُستخدَم من قبل الغرض كمرجع لنفسه. فعند تصميم الصف، تُستخدم الكلمة المفتاحية this في جسم التابع كمرجع له لاستخدام أعضائه، ويتمّ استخدامها لتجنّب الغموض الحاصل عن استخدام أسماء وسائط دخل للطرائق مطابقة لأسماء حقول أو خصائص الصف.

مثال:

في الرمّاز الآتي، الاسم name هو اسم حقل للصف وهو أيضاً اسم وسيط دخل لباني الصف الذي سيقوم بإسناد قيمة الوسيط إلى الحقل، وكذلك الأمر بالنسبة للحقل salary. ويمكن الاستغناء عن استخدام this بتغيير أسماء وسطاء الدخل إن أمكن ذلك.

```
public class Employee {
                private int salary;
                private string name;
                public Employee(string name, int salary)
                {
                  // Use this to qualify the members of the class
                  // instead of the constructor parameters.
                this.name = name;
                this.salary = salary;
                }
      }
```

وفي الصف السابق Person، الطريقة WritePersonNumber ساكنة وتطبع على الشاشة قيمة الحقل الساكن instances، وعند محاولة استخدام الحقل name ضمنها نحصل على رسالة الخطأ:

```
An object reference is required for the non-static field, method,
or property ‘Person.name’
```

والتي تفيد بضرورة إنشاء غرض من أجل استخدام عنصر غير ساكن.
أما عند محاولة استخدام الكلمة المفتاحية this، فنحصل على رسالة الخطأ:

```
Keyword ‘this’ is not valid in a static property, static method,
or static field initializer
```

والتي تفيد بأنّه لا يُسمح باستخدام هذه الكلمة مع الخصائص الساكنة ولا مع الطرائق الساكنة ولا حتى عند تهيئة الحقول الساكنة.

3.1. الباني الساكن

يمكن أن يكون للصف بانٍ ساكنٍ Static Constructor، ويتمّ التصريح عن الباني الساكن باستخدام الكلمة static أيضاً. ويُستخدم الباني الساكن من أجل تهيئة الحقول الساكنة أو للقيام بعمل ما يجب تنفيذه مرّة واحدة فقط. ويتمّ استدعاؤه تلقائياً قبل إنشاء أول غرض منه أو قبل أول استخدام لأحد أعضائه الساكنة، ويُستخدم أحياناً عند الكتابة في ملفّات التسجيل Log Files. وليس للباني الساكن محدّد وصول، وليس له وسائط دخل، ولا يمكن القيام بتحميل زائد له ولا يورَّث ولا يتمّ استدعاؤه مباشرة ولا يمكن أن يمتلك الصف أكثر من بانٍ ساكنٍ واحد.

مثال:

نوضّح كيفية استخدام الباني الساكن في الرمّاز التالي:

```
using System;
namespace StaticConstructor {
     class Car {
        // * A static constructor is executed only once,
        // when a class is first accessed.
        // * A static constructor cannot have any access modifiers
        // * A static constructor cannot have any parameters
```

```
        static Car() {
            Console.WriteLine("Car initialized");
        }// end static constructor

        //Instance constructor, this is executed every
        //time the class is created
        public Car()
        {
            Console.WriteLine("Car created");
        }// end Instance constructor

        static void Main( ) {
            Car mazda = new Car();
            Car toyota = new Car();
            Console.ReadKey();
        }// end Main
    }// end  class Car
}// end namespace StaticConstructor
```

وبعد التنفيذ، نحصل على الخرج الآتي:

```
Car initialized
Car created
Car created
```

4.1. الصف الساكن

يمكن للصف أن يكون ساكناً، ويتمّ التصريح عن صف بأنّه ساكن من خلال الكلمة المفتاحية static كما هو موضّح:

```
[<access modifier>]
static class <class_name>
{
    // … Class body goes here
}
```

والصف الساكن هو صف جميع عناصره ساكنة، وبالتالي لا يمكن إنشاء أي غرض منه. ويتمّ استخدام الصفوف الساكنة لتجميع مجموعة من الثوابت والطرائق ذات الاستخدام العام ضمن بنية واحدة.

1.4.1. الصف الساكن Math

يحوي الصف الساكن System.Math على مجموعة من الطرائق والثوابت التي تُستخدم لكتابة التعابير الرياضية المختلفة، ويوضّح الجدول الآتي بعضها:

الطريقة	توصيف
Abs(x)	القيمة المطلقة لـ x
Ceiling(x)	التقريب لأصغر عدد طبيعي ليس أصغر من x
Cos(x)	قيمة تجيب x (x مقاسة بالراديان)
Exp(x)	الرفع لقوة قاعدتها العدد النبري e
Floor(x)	التقريب لأكبر عدد طبيعي ليس أكبر من x
Log(x)	اللوغاريتم الطبيعي لـ x (الأساس e)
Max(x, y)	أكبر قيمة
Min(x, y)	أصغر قيمة
Pow(x, y)	x مرفوع للقوة y
Sin(x)	جيب x (x بالراديان)
Sqrt(x)	الجذر التربيعي لـ x

ومن بين الثوابت الرياضية المعرّفة ضمن الصف Math:

العدد PI = 3.14159… والعدد النبري E = 2.71828….

مثال:

في الرمّاز الآتي، نوضّح كيفية استدعاء طرائق وثوابت الصف Math:

```
using System;
namespace MathTest
{
    class MathTesting
    {
        static void Main()
        {
            Console.WriteLine(" Abs(-2.5) = {0}", Math.Abs(-2.5));
            Console.WriteLine(" Abs(3.3) = {0}", Math.Abs(3.3));
            Console.WriteLine(" Abs(0) = {0}", Math.Abs(0));
```

```
            Console.WriteLine("************************************");
            Console.WriteLine(" Ceiling(8.3)= {0}", Math.Ceiling(8.3));
            Console.WriteLine(" Ceiling(-8.8)= {0}", Math.Ceiling(-8.8));
            Console.WriteLine("************************************");
            Console.WriteLine(" Cos( 0 ) = {0}", Math.Cos(0));
            Console.WriteLine("************************************");
            Console.WriteLine(" Exp( 1 ) = {0}", Math.Exp(1));
            Console.WriteLine(" Exp( 2 ) = {0}", Math.Exp(2));
            Console.WriteLine("************************************");
            Console.WriteLine(" Floor(9.5) = {0}", Math.Floor(9.5));
            Console.WriteLine(" Floor(-9.5) = {0}", Math.Floor(-9.5));
            Console.WriteLine("************************************");
            Console.WriteLine(" Log(7.38905609893065) = {0}",
               Math.Log(7.38905609893065));
            Console.WriteLine("************************************");
            Console.WriteLine("Max( 2.2,5.1) = {0}", Math.Max(2.2, 5.1));
            Console.WriteLine("Min(2.2 ,5.1) = {0}", Math.Min(2.2, 5.1));
            Console.WriteLine("************************************");
            Console.WriteLine(" Pow( 2.0, 5) = {0}", Math.Pow(2.0, 5));
            Console.WriteLine("************************************");
            Console.WriteLine(" Sin( 0 ) = {0}", Math.Sin(0));
            Console.WriteLine("************************************");
            Console.WriteLine(" Sqrt( 9.0) = {0}", Math.Sqrt(9.0));
            Console.WriteLine("************************************");
            Console.WriteLine(" E= {0}", Math.E);
            Console.WriteLine(" PI = {0}", Math.PI);
            Console.WriteLine("************************************");
            Console.ReadKey();
        }// end Main
    }//end class MathTesting
}// end namespace MathTest
```

وبعد التنفيذ، نحصل عل الخرج الآتي:

```
Abs(-2.5) = 2.5
 Abs(3.3) = 3.3
 Abs(0) = 0
**********************************
 Ceiling(8.3)= 9
 Ceiling(-8.8)= -8
**********************************
 Cos( 0 ) = 1
**********************************
 Exp( 1 ) = 2.718281828459045
 Exp( 2 ) = 7.38905609893065
**********************************
 Floor(9.5) = 9
 Floor(-9.5) = -10
**********************************
 Log(7.38905609893065) = 2
**********************************
Max( 2.2,5.1) = 5.1
Min(2.2 ,5.1) = 2.2
**********************************
 Pow( 2.0, 5) = 32
**********************************
 Sin( 0 ) = 0
**********************************
 Sqrt( 9.0) = 3
**********************************
 E= 2.718281828459045
 PI = 3.141592653589793
**********************************
```

2. فضاءات الأسماء

عند العمل كفريق، قد يحدث أحياناً أن يقوم مبرمجان، من غير قصد، باستخدام نفس الاسم للدلالة على صفين مختلفين. فعند الحاجة لاستخدام هذين الصفين، سيحصل تضارب بالتسمية Naming Collision. ولحلّ هذه المشكلة، يتمّ تنظيم الصفوف ضمن فضاءات أسماء Namespaces، بحيث يتمّ استخدام اسم الصف مرّة واحدة ضمن فضاء الأسماء الخاص به. وبذلك، يمكن التمييز بين صفين لهما نفس الاسم بالاعتماد على فضاء أسماء كلٍّ منهما لأنّه لا يمكن أن يوجدا معاً ضمن فضاء الأسماء نفسه. وللوصول إلى الصف ClassName ضمن فضاء الأسماء MyNamespace يجب استخدام العبارة MyNamespace.ClassName. وفي حال كتابة العبارة using MyNamespace; في مقدّمة البرنامج، فإنّ using تعمل على استدعاء فضاء الأسماء MyNamespace، وعندها يمكن استخدام اسم الصف ClassName مباشرة من دون أن يُسبَق باسم فضاء الأسماء.

فضاء الأسماء هو تجمّع لأسماء عدّة صفوف مع بعضها البعض من دون الأخذ بعين الاعتبار كيفية تخزين الصفوف ضمن نظام الملفّات. أي أنّه يمكن لفضاء أسماء أن يوجد ضمن ملفّ وحيد أو أن يتمّ توزيع صفوفه على عدّة ملفّات. كما يمكن أن يحتوي الملفّ الواحد على أكثر من فضاء أسماء، وعادة ما تمتلك الصفوف المنتمية لنفس فضاء الأسماء صفات مشتركة وتُستَخدم ضمن نفس السياق. ويمكن لفضاء الأسماء أن يحوي فضاءات أسماء أخرى فنحصل على ما يُسمّى بفضاءات الأسماء المتداخلة Nested Namespaces.

مثال1:

```
using System;
namespace NS1
{
    namespace Ns2
    {
        namespace CSharp
        {
            public class Program
            {
                public static void Main()
                {
                    Console.WriteLine(typeof(Program).Namespace);
                    Console.ReadKey();
                }//end Main
            }//end Program
        }//end namespace CSharp
```

```
    }// end namespace N2
}// end namespace N1
```

يحتوي فضاء الأسماء NS1 على فضاء الأسماء NS2، والذي بدوره يحوي فضاء الأسماء CSharp الحاوي على الصف Program. وفي الطريقة Main، تمّ استخدام العبارة typeof(Program).Namespace التي تسمح بالحصول على اسم فضاء الأسماء الحاوي على الصف Program. وبعد تنفيذ البرنامج، نحصل على الخرج الآتي:

```
NS1.Ns2.CSharp
```

مثال2:

يمكن الاستعاضة عن الرمّاز في المثال1 بالرمّاز الآتي الذي يعطي نفس الخرج بعد تنفيذه:

```
using System;
namespace NS1.Ns2.CSharp
{
    public class Program
    {
        public static void Main()
        {
            Console.WriteLine(typeof(Program).Namespace);
            Console.ReadKey();
        }//end Main
    }//end Program
}// end namespace NS1.Ns2.CSharp
```

ويتضّح ممّا سبق أن فضاءات الأسماء تسمح بتنظيم الرمّاز المصدري ويتمّ الفصل بينها باستخدام المحرف (.).

في مكتبة صفوف إطار العمل FCL الخاصّ بـ (.NET)، يتمّ تنظيم الصفوف مسبقة التعريف ضمن عدّة فضاءات أسماء نشرح بعضها في الجدول الآتي:

فضاء الأسماء	توصيف
System	يحتوي على الصفوف الأساسية لمعظم البرامج، فيضمّ الأنماط الأساسية البسيطة والمركّبة مثل String وDateTime وغيرها، بالإضافة إلى الصفوف الأساسية الخاصّة مثل Math والاستثناءات والمصفوفات كما أنّه يضمّ الصف Console
System.Collections	يحتوي على صفوف وواجهات تُستخدم في تعريف القوائم والأرتال والمكدِّسات وجداول التقطيع والقواميس
System.IO	يضمّ صفوفاً تسمح بالتعامل مع الملفّات وسيول المعطيات، ويحتوي على أنماط تفيد بالتعامل مع نظم الملفّات. ومن أهمّ صفوفه: File وDirectory وStreamReader وStreamWriter وFileStream وBinaryReader وBinaryReader
System.Net	يضمّ صفوفاً تدعم العديد من البروتوكولات المستخدمة في الشبكات، مثل: HTTP وFTP وSMTP
System.Security	يضمّ الصفوف والواجهات المسؤولة عن توفير أمن المعطيات والتحكّم بالوصول إلى مختلف المكوّنات البرمجية
System.Text	يحتوي على صفوف للتعامل مع ترميز ASCII وUnicode، ويضمّ صفوفاً للتحويل بين كتل المحارف وكتل البايتات وصفوفاً لمعالجة وتنسيق السلاسل المحرفية
System.Threading	يضمّ صفوفاً وواجهات للاستخدام في البرمجة متعدّدة الخيوط
System.Xml	يضمّ صفوفاً وواجهات للتعامل مع XML

ويمكن أن يضم الصف عدة طرائق تحمل جميعها نفس الاسم ولكل منها سلوك مختلف عن الأخرى، وهذا ما نسميه بالتحميل الزائد للطرائق والذي سيكون موضوع الفصل القادم.

الأنشطة المرافقة

التمرين الأول: صف حساب التوفير SavingAccount

1. قم بالتصريح عن SavingAccount الذي يضمّ الحقل الساكن annualInterestRate لتخزين الفائدة السنوية لجميع الحسابات من هذا الصف.
2. أضف إلى هذا الصف الحقل الخاصّ savingsBalance لتخزين قيمة رصيد الحساب.
3. وأضف الطريقة CalculateMonthlyInterest التي تُستخدم لحساب الفائدة الشهرية، وذلك بضرب الرصيد savingsBalance بمعدّل الفائدة السنوية annualInterestRate مقسوماً على 12، ويجب إضافة هذه الفائدة إلى الرصيد.
4. يجب أن يكون للصف الطريقة الساكنة ModifyInterestRate لإسناد قيمة جديدة إلى الحقل الساكن annualInterestRate.
5. قم باختبار الصف السابق عن طريق إنشاء غرضين saver1 وsaver2 مع رصيد ابتدائي 2000 و 3000. وقم بإسناد 4% إلى الفائدة السنوية annualInterestRate، وبحساب الفائدة الشهرية. ثم قم بطباعة الأرصدة الجديدة.

التمرين الثاني: صف الموظف Employee

1. قم بالتصريح عن الصف Employee الذي يضمّ الحقل الساكن count الذي يعبّر عن عدد الموظفين.
2. أضف إلى هذا الصف الخاصّيتين التلقائيتين FirstName وLastName اللتين لا تسمحان بتغيير قيمهما من خارج الصف.
3. قم بالتصريح عن بانٍ للصف يأخذ وسيطي دخل ويُسند قيمتيهما إلى الخاصّيتين السابقتين، ويزيد قيمة الحقل count بمقدار واحد، ويطبع قيم العناصر الثلاثة.
4. قم بالتصريح عن هادم للصف، يقوم بإنقاص قيمة الحقل Count بمقدار واحد.
5. قم باختبار الصف السابق عن طريق إنشاء ثلاثة أغراض e1 وe2 وe2، وطباعة قيمة الحقل count.

المراجع

1. https://docs.microsoft.com/en-us/dotnet/csharp/
2. Dan Clark: “Beginning C# Object-Oriented Programming”, Berkeley, CA, Apress, 2013.
3. "التصميم والبرمجة غرضية التوجّه"، الدكتور سامي خيمي، الإجازة في تقانة المعلومات، من منشورات الجامعة الافتراضية السورية، الجمهورية العربية السورية، 2018.