




Chapter

Object and Class

- 
- **Object-oriented program must first classify the features and behaviors of an object.**
 - **The features of an object are described by Properties.**
 - **The behaviors of an object are described by Methods, and are simulated in terms of code.**
 - **Objects refer to data with properties and methods**



1. What is class?

- **Class is the collection of objects of the same attributes.**
- **Class itself is not a real object.**
 - ⇒ **Class is used to define the structure of an object, and describe the properties and methods of these similar objects.**
 - ⇒ **Objects refer to the classes that can be used to perform as individuals.**



1.2 Access restrictions for class members

- There are three common modifiers for accessing class members.

1. public

Access to public members is not restricted. Public members are allowed to be used in class, subclass, or declared objects. They are at public level.

2. private

Private members can be only accessed within their own classes.

They are at private level and can not be used by the outside world.

3. protected

Protected members can not only be accessed by their base classes, but also by their sub classes. They are at protection level.



1.3 Create objects and classes

How to create a class

- C# uses `class{...}` to define a class
- The definition of classes and events:
 - ⇒ it's cannot be placed in a method.
 - ⇒ it's cannot be placed outside the namespace `{...}`.
- Class definition must be a domain-wide announcement.

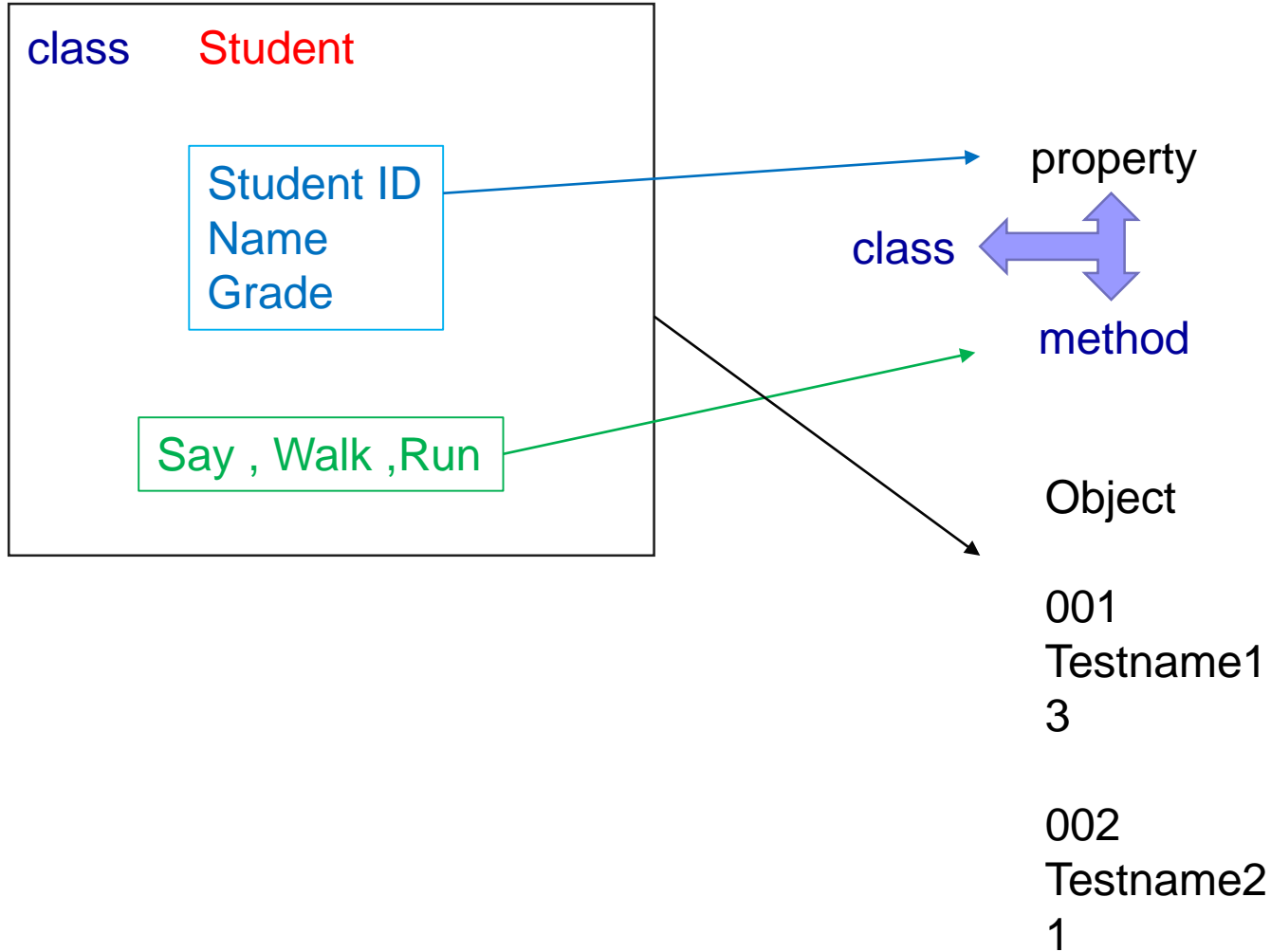


3 How to create properties

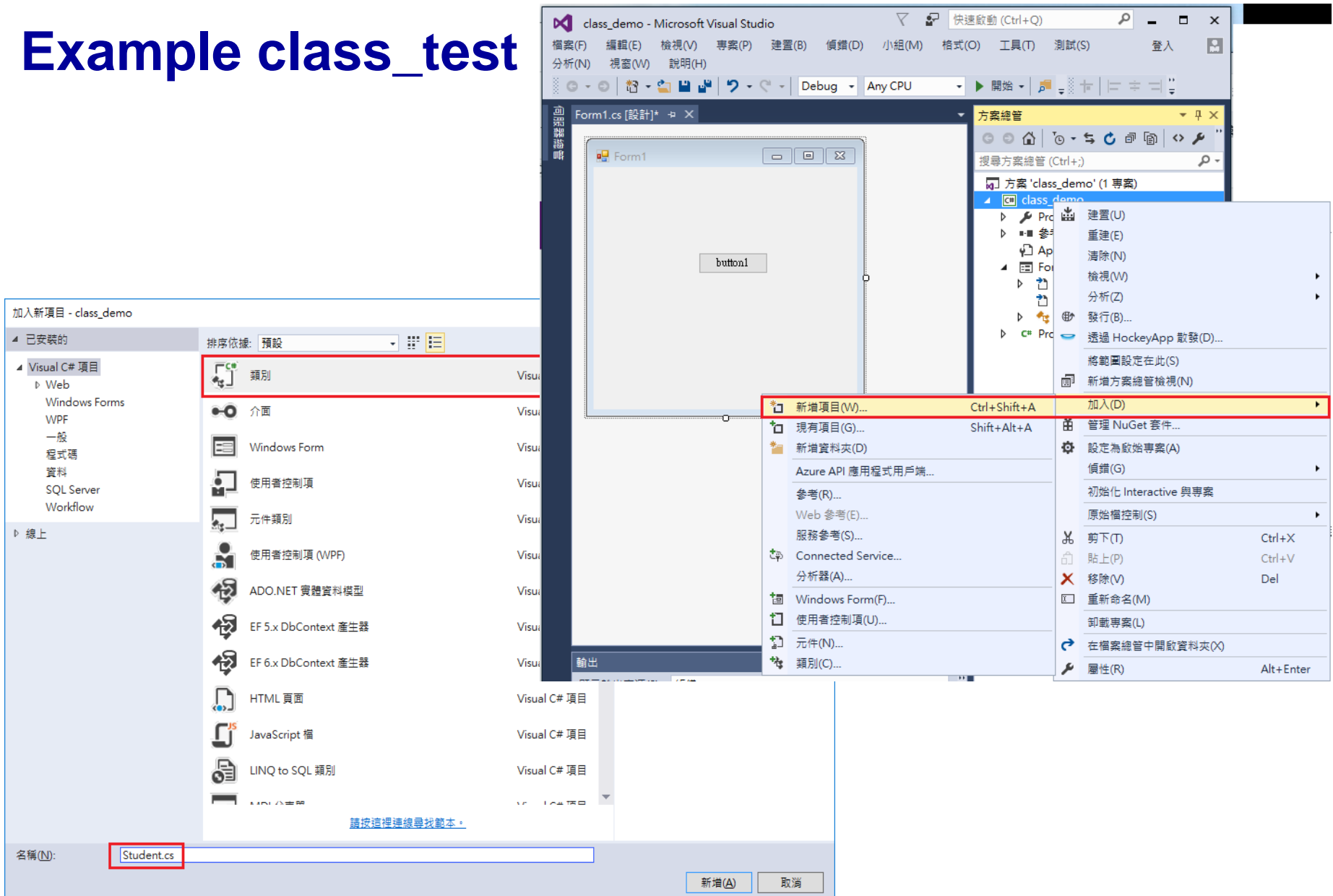
There are two ways to create properties:

1. Directly declare public variables in class
2. Use get and set accessors

Example class_test



Example class_test



Student.cs - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace class_test
{
    class Student
    {
        //Property==> The property of class is that Property uses variables to record the content.
        public int StudentID;
        public string Name;
        public int Grade;

        // Method==>>class can do , output type, string method name (input type and name)
        public string Say()
        {
            return "My name is" + Name + " , I am a" + Grade + "grade student";
        }
    }
}
```



class_test - Microsoft Visual Studio

檔案(F) 編輯(E) 檢視(V) 專案(P) 建置(B) 偵錯(D) 小組(M) 工具(T) 測試(S) 分析(N) 視窗(W) 說明(H)

Debug Any CPU 開始

回退
前進
取消
確定

Student.cs Form1.cs Form1.cs [設計]

class_test

class_test.Student

StudentID

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace class_test
8  {
9      class Student
10     {
11         //Property==> The property of class is that Property uses variables to record the content.
12         public int StudentID;
13         public string Name;
14         public int Grade;
15
16         // Method==>class can do , output type, string method name (input type and name)
17         public string Say()
18         {
19             return "My name is " + Name + " , I am a " + Grade + " grade student";
20         }
21     }
22 }
23
```

2. Constructor functions

- Each class has a constructing function (a constructor).
 - ⇒ It has a method with the same name as the class.
- This constructor is called when an object is created based on a class definition.
- Constructor is usually used to set the initial value of the variable defined in the class.
- ① The initial value of the numeric data type is set to zero.
- ② The initial value of Boolean data type shall be set to false.
- ③ The reference type's initial value is set to null.
- Constructors may not set the initial value.
 - ⇒ All these data types are automatically initialized.
- Constructor without any parameters is called Default Constructor.



When constructing and destructing an object, the follows shall be noted:

- 1. If no constructor is defined in the category, a default constructor is automatically provided that does nothing (Default constructor).**
 - **If a constructor is not written when defining a class, C# automatically generates a Default Constructor.**
 - **When a construct is defined, the default construct automatically disappears.**
- 2. The name of the constructor must have the same name as the class name.**

```

Form1.cs  X  Form1.cs [設計]  Student.cs
C# Constructor_test  Constructor_test.Form1
7      using System.Text;
8      using System.Threading.Tasks;
9      using System.Windows.Forms;
10
11      namespace Constructor_test
12      {
13          public partial class Form1 : Form
14          {
15              public Form1()
16              {
17                  InitializeComponent();
18              }
19
20              private void button1_Click(object sender, EventArgs e)
21              {
22                  Student s1 = new Student(); // Question 1. Write a
23
24                  s1.Name = "testname1";
25                  s1.StudentID = 10602001;
26
27                  Student s2 = new Student();
28                  s1.Name = "testname2";
29                  s1.StudentID = 10602002;
30
31                  MessageBox.Show(s2.Say());
32              }
33          }
34      }
35

```

[Reminder]
Create object: new class name();

```

Student.cs  Form1.cs [設計]  Form1.cs X
C# Constructor_test  Constructor_test.Form1
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace Constructor_test
12 {
13     public partial class Form1 : Form
14     {
15         public Form1()
16         {
17             InitializeComponent();
18         }
19
20         private void button1_Click(object sender, EventArgs e)
21         {
22             Student s1 = new Student(); // Question1. 4 rows of
23
24             s1.Name = "testname1";
25             s1.Grade = 1;
26             s1.StudentID = 10602001;
27
28             Student s2 = new Student();
29             s1.Name = "testname2";
30             s1.Grade = 1;
31             s1.StudentID = 10602002;
32
33             MessageBox.Show(s2.Say());
34         }
35     }
36 }

```

[Question]

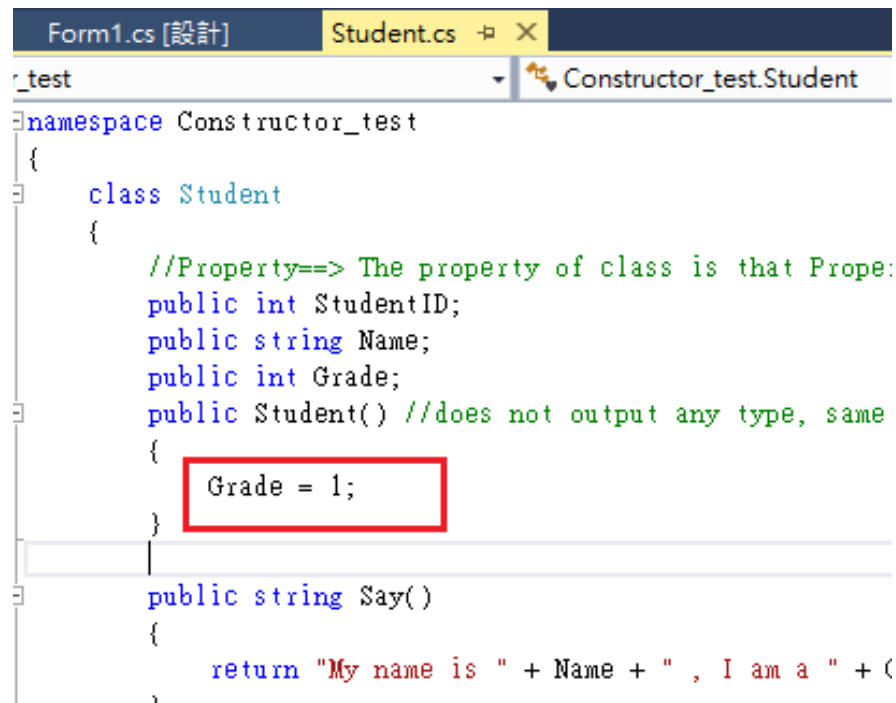
1. Four rows of code are needed when creating one object
2. Repeat setting the same value

```
Form1.cs  Form1.cs [設計]  Student.cs  X
# Constructor_test  Constructor_test.Student  Grade

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Windows.Forms; //使用 messagebox
6  using System.Threading.Tasks;
7
8  namespace Constructor_test
9  {
10     class Student
11     {
12         //Property==> The property of class is that Property uses variables to record the
13         public int StudentID;
14         public string Name;
15         public int Grade;
16
17         public Student() // does not output any type, same name as the class
18         {
19             MessageBox.Show("This is a test!!");
20         }
21
22         public string Say()
23         {
24             return "My name is " + Name +
25
26         public string Talk(Student s) //
27
28 namespace Constructor_test
29 {
30     public partial class Form1 : Form
31     {
32         public Form1()
33         {
34             InitializeComponent();
35
36         private void button1_Click(object sender, EventArgs e)
37         {
38             new Student();
39         }
40     }
41 }
```

Create a sub Constructor

1. One kind of Method
2. Implement when creating objects



```
Form1.cs [設計] Student.cs [X]
Constructor_test.Student
namespace Constructor_test
{
    class Student
    {
        //Property==> The property of class is that Prope:
        public int StudentID;
        public string Name;
        public int Grade;
        public Student() //does not output any type, same
        {
            Grade = 1;
        }
        public string Say()
        {
            return "My name is " + Name + " , I am a " + (
```

If the default of attribute is known at the beginning, it can be applied to the sub constructor to set objects beforehand



The End