COMP1022Q
Introduction to Computing with Excel VBA

Objects

Gibson Lam, David Rossiter and Eddie Chan

Overview

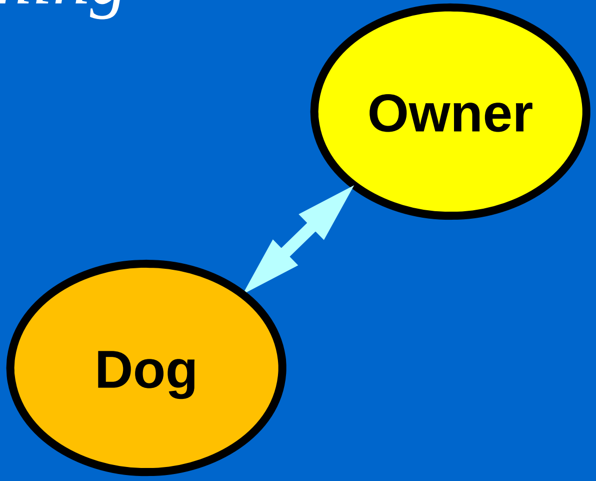
- *Object-Oriented Programming* is an advanced topic in computer programming
- In this presentation, we will look at these:
  - Introduction to Object-Oriented Programming
  - What is a Class?
  - An Example Class – a Dog Class
  - Another Example Class – a Person Class

Introduction to Objects

- There are many 'objects' around us in the real world, e.g. a dog and a car are both objects
- We can say that each object has two kinds of characteristics: *attributes* and *behaviors*
- For example, a dog has:
  - *attributes* such as name, colour and weight
  - *behaviors* such as eating, barking and running

Object-Oriented Programming

- We are dealing with 'objects' every day
- It would be great if we can ask a program to 'think' using objects too
- This way of programming, thinking using objects, is called *object-oriented programming*
- To do that we first design the objects and then use the objects to interact with each other



What is a Class?

- In Computer Science we usually call the definition of an object a *Class*
- A class is only a definition
- When you want to create an object you need to make an *instance* of the class
- In a program you can create as many instances of the class as you want

An Example of Using a Class 1/2

- Let’s say we have created a Dog class
- In order to make Snoopy and Odie we need to create an instance of the Dog class for each of them, like this:

A Dog Class

Make an instance →

Name: Snoopy
Colour: White
Weight: 30kg

Make an instance →

Name: Odie
Colour: Yellow
Weight: 25kg

Two instances of the Dog class with different attributes

An Example of Using a Class 2/2

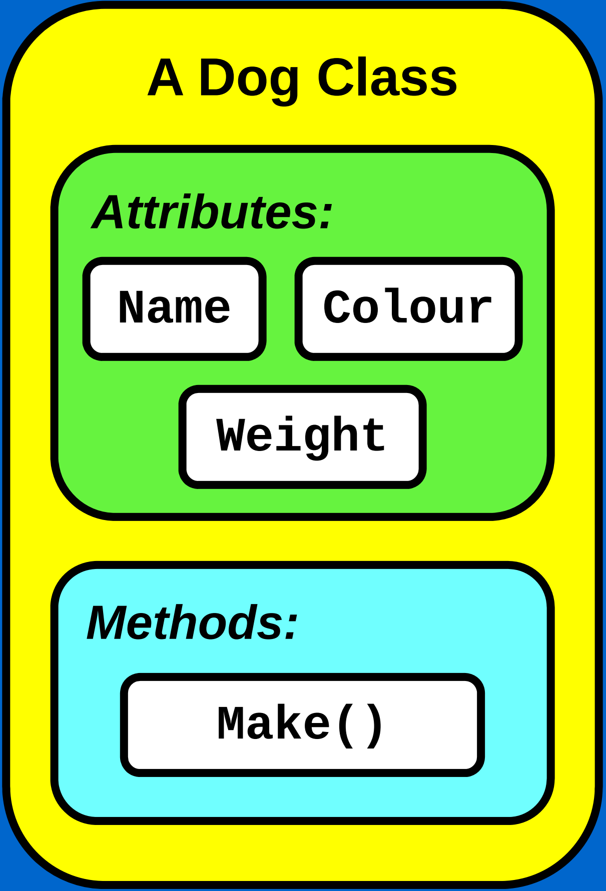
- Both the Snoopy instance and the Odie instance are created using the same class, the Dog class
- They are different to each other because they have different attribute values, such as their name, colour and weight

Name: Snoopy
Colour: White
Weight: 30kg

Name: Odie
Colour: Yellow
Weight: 25kg

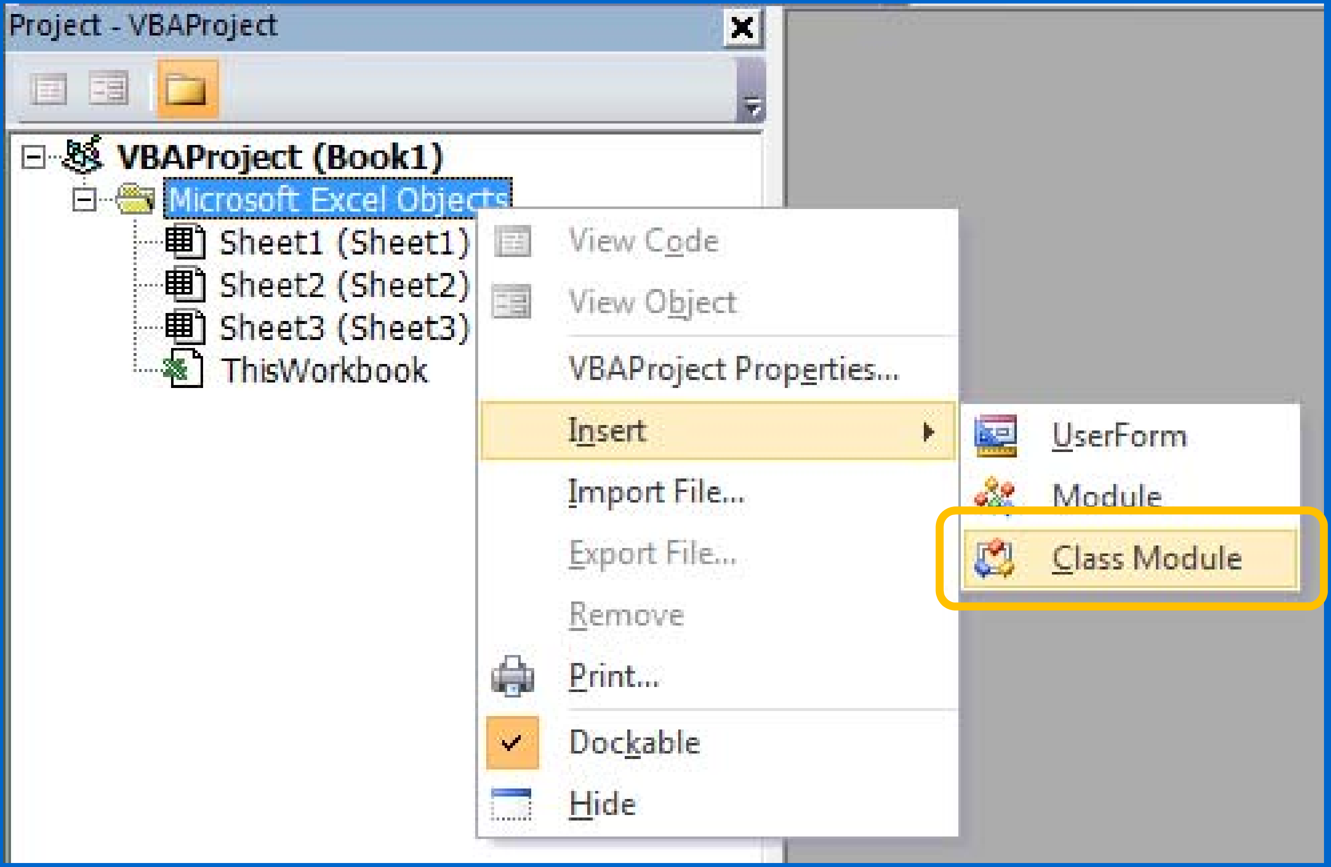
Creating the Dog Class in VBA

- Let's create the Dog class in VBA
- The Dog class has the following attributes:
  - `Name`
  - `Colour`
  - `Weight`
- And the class has this behaviour:
  - `Make()`
- In computer programming, we call a behaviour a *method*



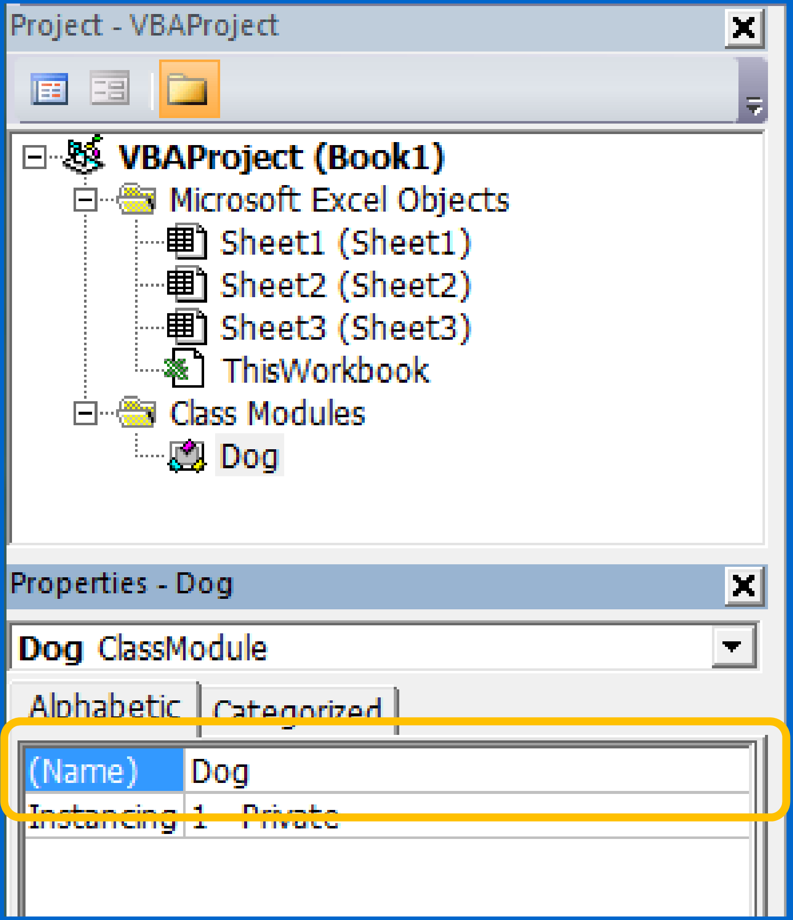
Creating a Class Module 1/2

- To create a class in VBA we need to first create a *Class Module* in the VBAProject, like this:



Creating a Class Module 2/2

- After we have the class module we need to change its name to the class name we want to use
- For example, here we change the name of the class module to ‘Dog’ because we are making a class called dog



(Name)	Dog
Instancing	1 - Private

Making the Class Attributes

- Next, we can add the attributes to the class
- To do that, double-click on the class module and type the code at the top of the file i.e.:

```
Public Name As String
Public Colour As Long
Public Weight As Double
```

- These are the attributes of the class
- As we discussed previously in the course, the word ‘Public’ means any code in the same Excel file can read the content of these attributes

Making a Class Method

- We have to use long integers, i.e. `Long`, to store colours, but understanding that is outside the scope of this course

- Now we add the method in the class module:

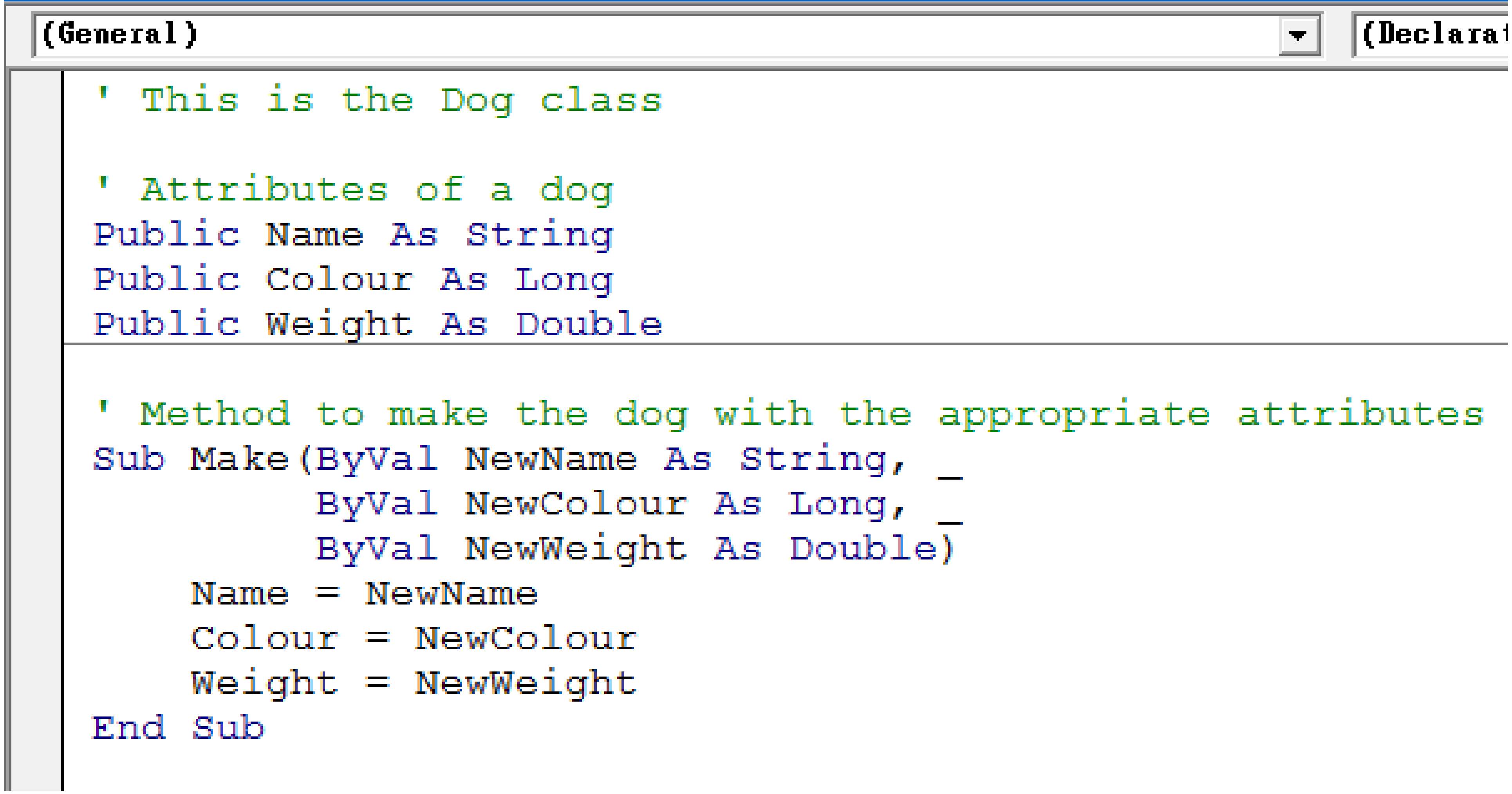
```
Sub Make(ByVal NewName As String, _
         ByVal NewColour As Long, _
         ByVal NewWeight As Double)
    Name = NewName
    Colour = NewColour
    Weight = NewWeight
End Sub
```

Initialize the attribute values

- The purpose of this method is to initialize the Dog instance with the appropriate attribute values

The Dog Class Code

- In summary, this is the code of the Dog class module:



```
' This is the Dog class

' Attributes of a dog
Public Name As String
Public Colour As Long
Public Weight As Double

' Method to make the dog with the appropriate attributes
Sub Make(ByVal NewName As String, _
         ByVal NewColour As Long, _
         ByVal NewWeight As Double)
    Name = NewName
    Colour = NewColour
    Weight = NewWeight
End Sub
```

Using the Dog Class 1/2

- After defining the Dog class, let’s use the class to create two dogs, Snoopy and Odie
- To create an instance of the Dog class, first, we need to create a variable to store the instance, like this:

$$\texttt{Dim }\underbrace{\texttt{Snoopy}}_{\textit{Name of this instance}}\texttt{ As }\underbrace{\texttt{Dog}}_{\textit{‘Dog’ is the data type (the class)}}$$

- Second, we create an instance using the *New* keyword:

$$\texttt{Set Snoopy = }\underbrace{\texttt{New Dog}}_{\textit{Create a new instance of the ‘Dog’ class}}$$

Using the Dog Class 2/2

- Now the `Snoopy` variable stores an instance of the Dog class
- However, the attributes of `Snoopy` are not set
- Let’s use appropriate attributes for Snoopy by calling the `Make()` method we made, like this:

$$\texttt{Snoopy.Make }\underbrace{\texttt{"Snoopy"}}_{\textit{Name}}\texttt{, }\underbrace{\texttt{vbWhite}}_{\textit{Colour}}\texttt{, }\underbrace{\texttt{30}}_{\textit{Weight}}$$

Name: Snoopy
Colour: White
Weight: 30kg

Creating a Second Instance of the Dog Class

- Similarly, we can create another instance of the Dog class using the following code:

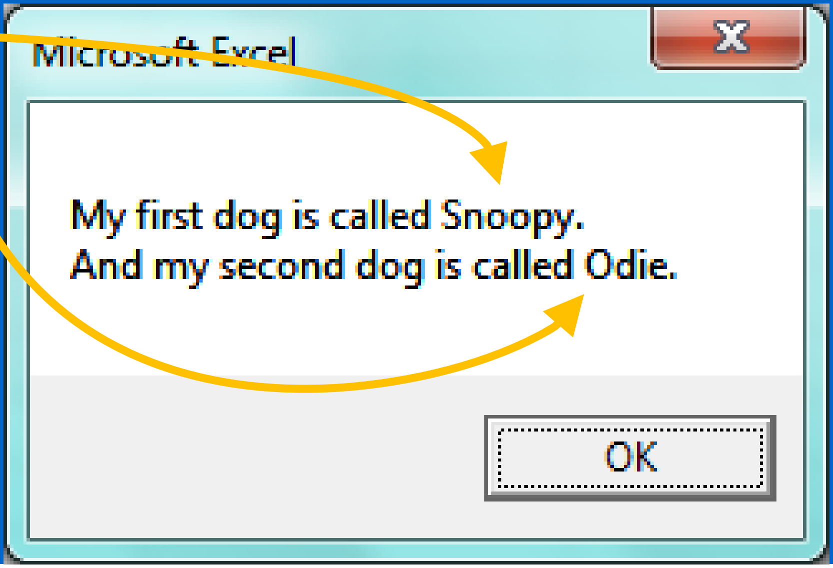
```
Dim Odie As Dog

Set Odie = New Dog
Odie.Make "Odie", vbYellow, 25
```

Name: Odie
Colour: Yellow
Weight: 25kg

Using Both Dog Instances

- We have two instances of the Dog class now
- Let’s show their names using the following code:

```
Msgbox "My first dog is called " & _
       Snoopy.Name & "." & vbNewLine & _
       "And my second dog is called " & _
       Odie.Name & "."
```

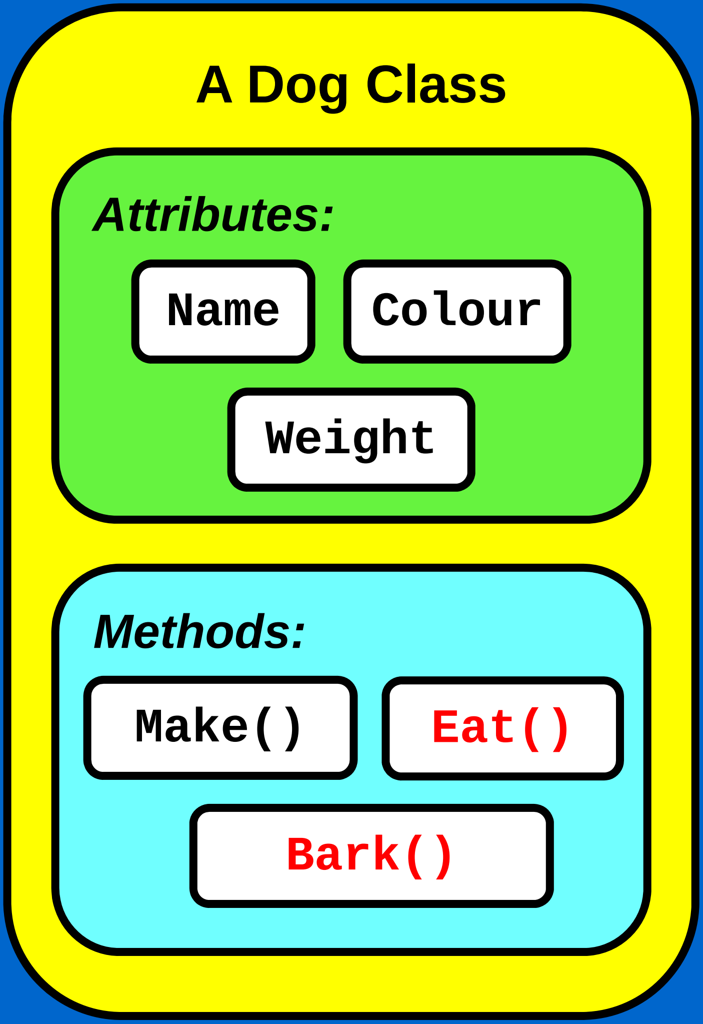


```
My first dog is called Snoopy.
And my second dog is called Odie.
```

vbNewLine simply moves the text to the next line

Extending the Dog Class

- So far, the Dog class does not do anything
- Let’s make it more interesting by adding two more methods to the class:
  - `Bark()`
  - `Eat()`

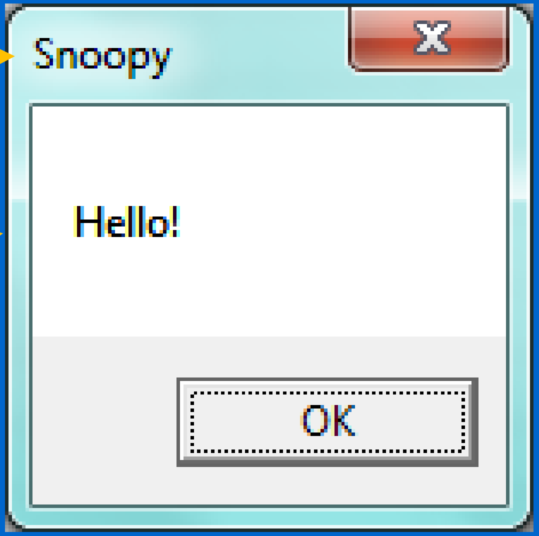


Creating a Bark Method

- Let's first create the `Bark()` method
- The `Bark()` method gets an input parameter and then 'barks' using a message box, as shown below:

```
Sub Bark(ByVal Woof As String)
    MsgBox Woof, __, Name
End Sub
```

We don't need the message box icon here so the parameter is omitted



- For example, Snoopy can bark 'Hello' using this code:

```
Snoopy.Bark "Hello!"
```

Creating a Eat Method

- Let's add another method, `Eat()`, to the Dog class
- The idea of the `Eat()` method is:
  1. The dog's weight increases after eating
  2. The dog barks when the dog is full

```
Sub Eat()
    Weight = Weight + 1

    If Weight >= 35 Then
        Bark "Oh dear! I am full!"
    End If
End Sub
```

Here the Eat() method uses another method, Bark(), from within the same class

Using the Eat Method 1/2

- Using the `Eat()` method you can then keep on feeding the dogs

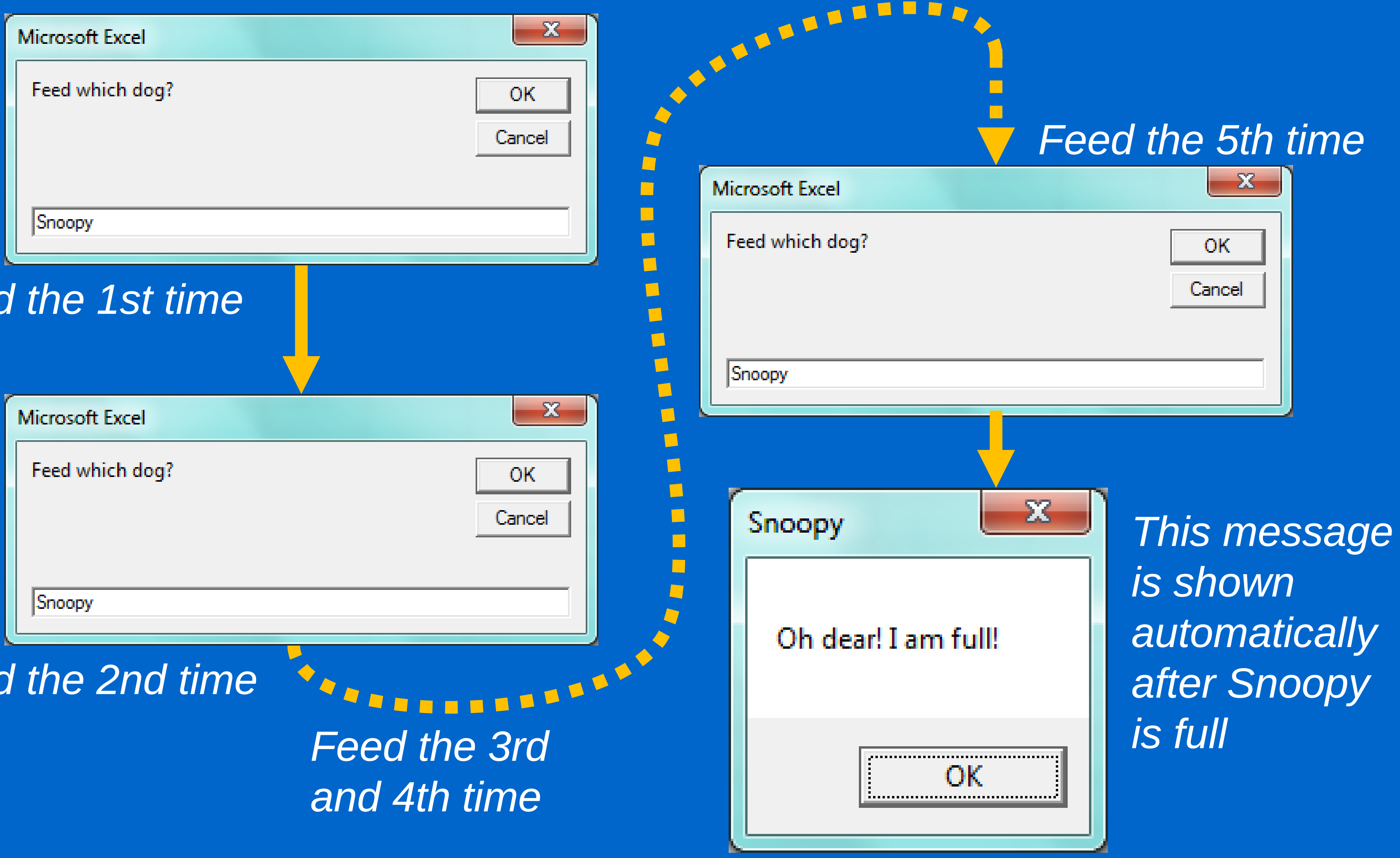
```
Dim DogToFeed As String
Do
    DogToFeed = InputBox("Feed which dog?")

    If DogToFeed = "Snoopy" Then
        Snoopy.Eat
    ElseIf DogToFeed = "Odie" Then
        Odie.Eat
    End If
Loop Until DogToFeed = ""
```

Feed the dog whose name was typed in

Stop feeding the dogs if the user didn't type anything

Using the Eat Method 2/2

- Here is an example of feeding Snoopy five times:



Microsoft Excel

Feed which dog?

Snoopy

Feed the 1st time

Microsoft Excel

Feed which dog?

Snoopy

Feed the 2nd time

Feed the 3rd and 4th time

Feed the 5th time

Microsoft Excel

Feed which dog?

Snoopy

Snoopy

Oh dear! I am full!

This message is shown automatically after Snoopy is full

The Person Class

- Let’s look at another example
- In this example, we define a Person class
- The Person class contains five attributes and four methods, which are shown in the next two slides

Attributes in the Person Class

- Here are the attributes of the Person class:

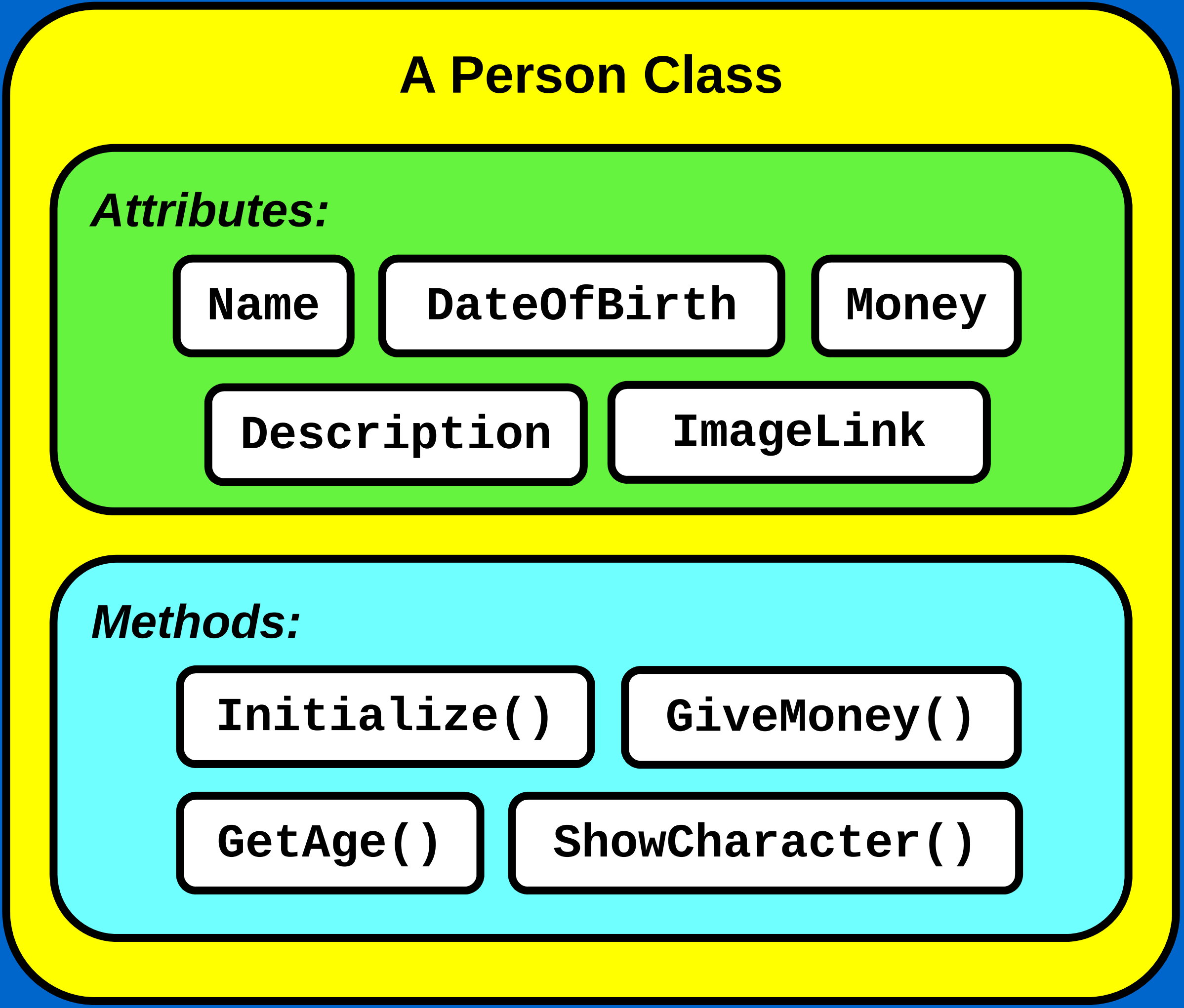
`Name`	Name of the person
`DateOfBirth`	Date of birth
`Money`	How much money the person has
`Description`	A simple text description
`ImageLink`	A link to an image

Methods in the Person Class

- Here are the methods of the Person class:

`Initialize()`	Set up the initial values of the attributes
`GiveMoney()`	Take money from this person and give it to another person
`GetAge()`	Calculate the person’s age
`ShowCharacter()`	Show an image of the person

A Summary of the Person Class



- The code for this class is shown in the next few slides

Attributes of the Person Class

- The attributes of the Person class are created like this:

```
Public Name As String
Public DateOfBirth As Date
Public Money As Double
Public Description As String
Public ImageLink As String
```

The Initialize Method

```
Sub Initialize(ByVal NewName As String, _
               ByVal NewDateOfBirth As Date, _
               ByVal NewMoney As Double, _
               ByVal NewDescription As String, _
               ByVal NewImageLink As String)
    Name = NewName
    DateOfBirth = NewDateOfBirth
    Money = NewMoney
    Description = NewDescription
    ImageLink = NewImageLink
End Sub
```

Here the five attributes of the class are given their initial values

The GiveMoney Method

- This method handles what happens when this person gives someone else some money
- To achieve that, the method deducts some money from this instance of the object, and increases it in another instance

```
Sub GiveMoney(ByVal Amount As Double, _
              TargetPerson As Person)
    ' Decrease the attribute Money of this object
    Money = Money - Amount

    ' Increase the money of the target instance
    TargetPerson.Money = TargetPerson.Money + _
                         Amount
End Sub
```

The GetAge Method

- This method does a very simple assessment of how many years old this person is (it is not very accurate)

```
Function GetAge() As Integer
    Dim CurrentYear As Integer

    ' Get the current year
    CurrentYear = DateTime.Year(DateTime.Now)

    GetAge = CurrentYear – _
             DateTime.Year(DateOfBirth)
End Function
```

Return the current date and time

This is one of the attributes of the object

The ShowCharacter Method

- This method shows the image of the person, i.e. Hello Kitty, in a Web browser

```
Sub ShowCharacter()
    ' Start a web browser and go to
    ' the web page which shows the
    ' image of the person
    ThisWorkbook.FollowHyperlink ImageLink
End Sub
```

Using the Class

- We can now use the Person class as many times as we like, e.g. we can create these four Person objects:

Name: Hello Kitty
DateOfBirth: 13/12/1984
Money: $10,000
Description: Very energetic and loves to play outdoors

Name: Mary White
DateOfBirth: 28/11/1960
Money: $50,000
Description: Very kind and loving, she loves cooking and taking care of the house

Name: George White
DateOfBirth: 15/10/1957
Money: $100,000
Description: Hardworking and dependable, but has a good sense of humor

Name: Mimi
DateOfBirth: 13/12/1984
Money: $5,000
Description: Kitty's twin sister. She wears a ribbon on her right ear so people can tell her and Kitty apart

Creating One Person

- Let's create Hello Kitty with these attributes, plus one more:

Name:	Hello Kitty
DateOfBirth:	13/12/1984
Money:	$10,000
Description:	Very energetic and loves to play outdoors

- The VBA code to create this person is:

```
Dim Kitty As Person
Set Kitty = New Person

Kitty.Initialize "Hello Kitty", "1984/12/13", _
  10000, "Very energetic and loves to play " & _
  "outdoors", "http://www.hellokittyfan.com/" & _
  "hello-kitty-pics/hello-kitty-01.gif"
```

Creating Three More People 1/2

- Similarly, we can create the three other people in the family using the code shown here:

```
Dim Mary As Person, George As Person, Mimi As Person

Set Mary = New Person
Mary.Initialize "Mary White", "1960/11/28", 50000, _
  "Very kind and loving, she loves cooking and " & _
  "taking care of the house", _
  "http://www.hellokittyfan.com/" & _
  "hello-kitty-pics/hello-kitty-mom-01.gif"
```

Continued on the next slide

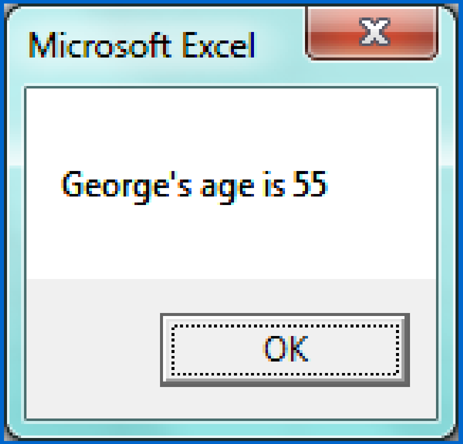
Creating Three More People 2/2

```
Set George = New Person
George.Initialize "George White", "1957/10/15", _
  100000, "Hardworking and dependable, but " & _
  "has a good sense of humor", _
  "http://www.hellokittyfan.com/" & _
  "hello-kitty-pics/hello-kitty-dad-01.gif"

Set Mimi = New Person
Mimi.Initialize "Mini White", "1984/12/13", _
  5000, "Kitty's twin sister. She wears a " & _
  "ribbon on her right ear so people can  " & _
  "tell her and Kitty apart", _
  "http://www.hellokittyfan.com/" & _
  "hello-kitty-pics/mimmy-01.gif"
```

Showing Their Ages

- Now, we have created some instances of the Person class, let's show some examples of how to use them
- For example, if you want to see the age of George, you can use the following line of code:

```
MsgBox "George's age is " _
  & George.GetAge()
```


Microsoft Excel

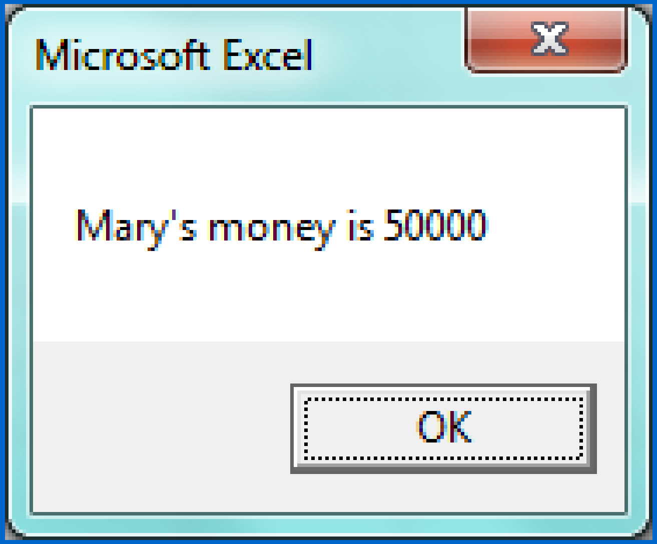
George's age is 55

- It makes sense to use a method to calculate the age because it changes every year so we calculate it every time

Showing Their Money

- If you want to see how much money Mary has you can use the following code:

```
MsgBox "Mary's money is " _
  & Mary.Money
```


Microsoft Excel

Mary's money is 50000

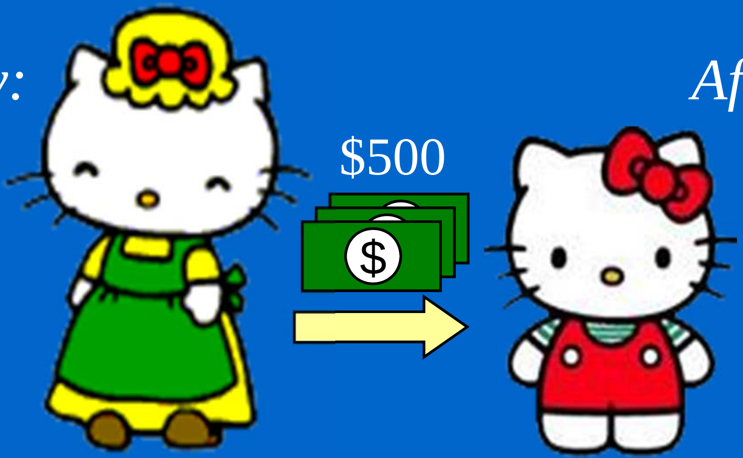
Chinese New Year

- In Chinese New Year, Chinese people usually give red packets to the younger generation
- Let's say Mary gives a red packet of $500 to Kitty in Chinese New Year

[Starts with $50,000]

After giving the money:

Name: Mary White
Money: $49,500



[Starts with $10,000]

After receiving the money:

Name: Hello Kitty
Money: $10,500

Using the Class Method

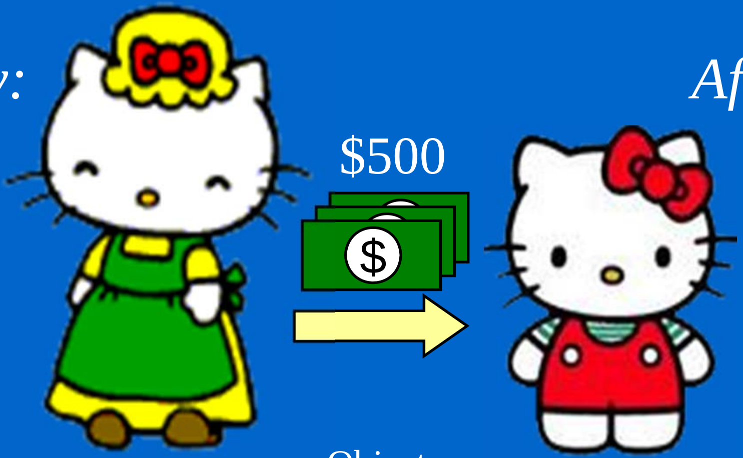
- In VBA code, this means Mary performs `GiveMoney()` to Kitty so that Mary's money is decreased whereas Kitty's money is increased, i.e.:

```
Mary.GiveMoney 500, Kitty
```

[Starts with $50,000]

After giving the money:

Name: Mary White
Money: $49,500



[Starts with $10,000]

After receiving the money:

Name: Hello Kitty
Money: $10,500

Cleverer Code

- If we want to we can write cleverer code which automatically works out whether it is the first day of Chinese New Year, like this:

```
Dim FirstDayChineseNewYear As Date

FirstDayChineseNewYear = "2013/02/10"

If DateTime.Date = FirstDayChineseNewYear Then
    Mary.GiveMoney 500, Kitty
End If
```

First day of Chinese New Year in 2013

Return the current date

The First Day of Every Month

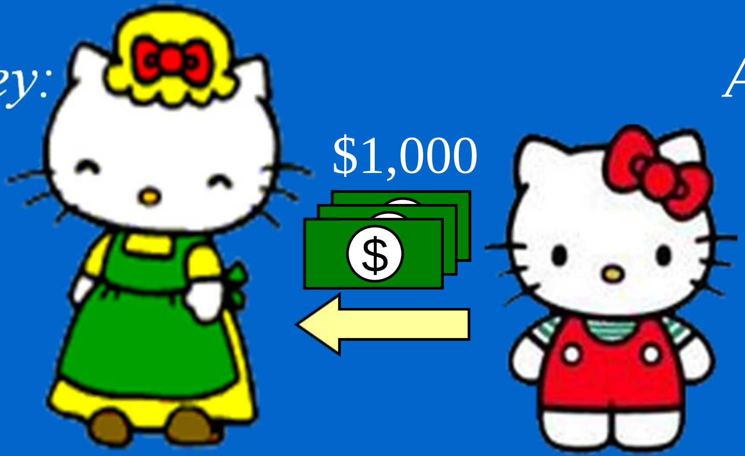
- Kitty gives $1,000 to her mother Mary on the first day of the month (every month)

[Starts with $50,000]

After receiving the money:

Name: Mary White

Money: $51,000



[Starts with $10,000]

After giving the money:

Name: Hello Kitty

Money: $9,000

Using the Class Method

- In VBA code, this means Kitty performs `GiveMoney()` to Mary so that Kitty's money is decreased whereas Mary's money is increased, i.e.:

```
Kitty.GiveMoney 1000, Mary
```

[Starts with $50,000]

After receiving the money:

Name: Mary White

Money: $51,000

[Starts with $10,000]

After giving the money:

Name: Hello Kitty

Money: $9,000

Cleverer Code

- If we want to we can write cleverer code which automatically works out whether it is the first day of the month, like this:

```
' Check if today is the first day of a month
If DateTime.Day(DateTime.Date) = 1 Then
    Kitty.GiveMoney 1000, Mary
End If
```

Who Has More Money?

- The following code shows how to evaluate who has more money:

```
If Kitty.Money > Mary.Money Then
    MsgBox Kitty.Name & " is richer."
ElseIf Mary.Money > Kitty.Money Then
    MsgBox Mary.Name & " is richer."
Else
    MsgBox "They have the same amount of money."
End If
```

Who Is Older?

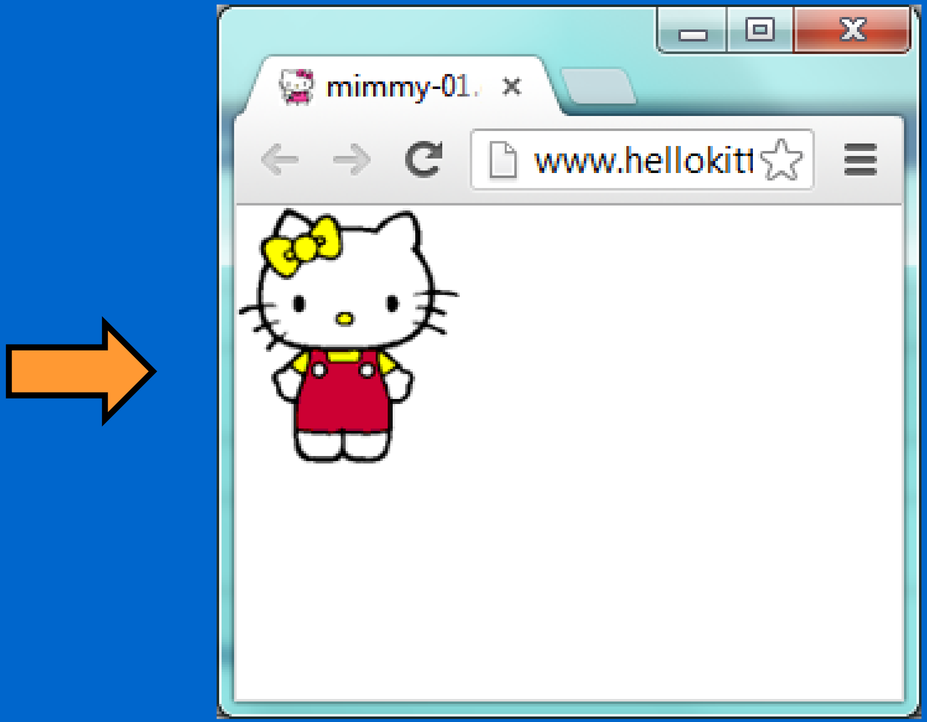
- The following code shows how to evaluate who is older:

```
If Kitty.GetAge() > Mary.GetAge() Then
    MsgBox Kitty.Name & " is older."
ElseIf Mary.GetAge() > Kitty.GetAge() Then
    MsgBox Mary.Name & " is older."
Else
    MsgBox "They have the same age."
End If
```

What Do They Look Like?

- All Person objects contain web links to their image
- To see what they look like, we can use their `ShowCharacter()` method
- The method starts a web browser and shows the image there, like this:

```
Mimi.ShowCharacter
```


mimmy-01.
www.hellokitt