

Chapter Seven: Sprites

In this chapter we will discuss one of the most important building blocks of any game: sprites. We will also begin building a new game to demonstrate how to use sprites!

Lesson One: Sprite Concepts

What is a sprite? In game programming, a sprite is a graphical object on the screen that we can move and manipulate with our code. Sprites can be simple, inanimate, immovable objects or complicated, animated, movable, objects that can interact and collide with one another. A single game can have any number of sprites. Sprites can represent your good guys, bad guys, ships, missiles, walls, etc.



In game programming, a Sprite is not a refreshing drink! A Sprite can be any graphical object on the screen. A spaceship, ping-pong ball, and a rock are all possible examples of game Sprites.

Although your sprites may be visually very different from one another, you will find that implementing them in code requires the same basic set of tasks each time. Sprites all typically need to move and react in response to user input or to each other. You may want your sprite to speed up or slow down or bounce off a wall.

Instead of writing lots of repetitive code to handle each sprite individually, most game programs use a sprite library as part of their game logic. For this course we provide a sprite library for you! The sprite library defines a new Sprite data type you can declare as a variable. The Sprite contains properties to represent the object's size, position, speed, and other parameters. The Sprite also contains methods that will make the sprite move, collide, and other useful features. By using our provided Sprite library you will be able to write programs more quickly and easily instead of re-inventing the wheel within each program!

In the next two chapters we will describe several important sprite concepts and then specify how those concepts are supported by the Sprite library. Along the way you will be developing a fully functional Bubble Blaster game with the techniques you have learned!

The Sprite Library

Our Sprite library is contained in a code file called ‘SpriteLibrary.vb’. This file will be present in each of your Activity Starter projects. During the next few chapters we will show you how this library works. Once we are done you will have a fully-featured Sprite library that can be used in any game program!

To use the Sprite library, first you need to add the file ‘SpriteLibrary.vb’ to your project. This is done by clicking on the Project menu and then choosing ‘Add Existing Item’. Then you just need to navigate to the directory that contains the ‘SpriteLibrary.vb’ file and add that file to the project. Since we have provided Activity Starter projects for all of the activities, this step has already been done for you.

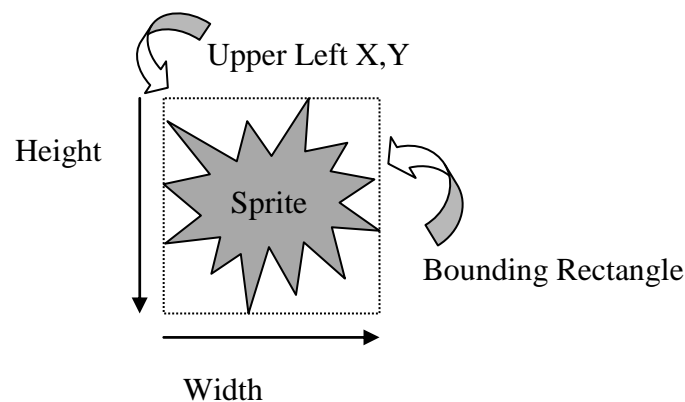
Once you have the Sprite Library file added to the project, you can then create as many variables as you like where the data type is ‘Sprite’:

```
Dim mySprite As Sprite = New Sprite()
```

Notice that you have to use the **New** keyword to create a new copy of the sprite to store in the variable!

Sprite Positioning and Size

Sprites are visual objects you see on the screen. Sprites therefore must have some properties describing their position and size. The position is represented by the same X and Y coordinates we have worked with in previous chapters. Since we are representing the position of a sprite with a single pair of coordinates, we need to know which part of the sprite those coordinates represent. It is customary to identify sprite position using the coordinates of the upper-left corner of the sprite. If the sprite is not a rectangle, you can mentally draw a rectangle around the outer edges of the sprite shape. This rectangle is called a bounding rectangle. The sprite is then located using the upper-left corner of the bounding rectangle.



The Sprite library contains a function called **GetBoundingRectangle()**, which will return the height, width and size of the Sprite’s bounding rectangle. This information is returned as a Rectangle that specifies the location and size of the bounding rectangle for the sprite. Since most graphics functions (**DrawEllipse()**,

DrawRectangle(), etc.) can take one **Rectangle** variable instead of usual four individual values that are required for location and size, the **GetBoundingRectangle()** function comes in real handy!

So far you have learned about two properties of a sprite: *position* and *size*. These properties are part of the **Sprite** data type as follows:

```
Public UpperLeft As Point
Public Size As Point
```

Both properties are **Points**, which means they have an X and Y component. The **UpperLeft** **Point** represents the screen coordinates of the upper left corner. The **Size** **Point** represents the width (X) and height (Y) of the sprite.

When you want to get or set the sprite size or position just read or assign the **UpperLeft** and **Size** properties accordingly.

```
' set the sprite position to (50,100)
mySprite.UpperLeft.X = 50
mySprite.UpperLeft.Y = 100

' set the sprite size to width = 20, height = 30
mySprite.Size.X = 20
mySprite.Size.Y = 30
```

Sometimes you may find it more convenient to get or set the sprite position in terms of the sprite's center instead of the upper-left corner. Our **Sprite** library has functions that will do that too!

```
Public Sub SetCenter(ByVal center As Point)
Public Function GetCenter() As Point
```

You can call those methods on your sprite objects as follows:

```
' get the sprite's current center location
Dim myCenter = mySprite.GetCenter()

' set the sprite's new location using the center point at (200,150)
myCenter.X = 200
myCenter.Y = 150
mySprite.SetCenter(myCenter)
```