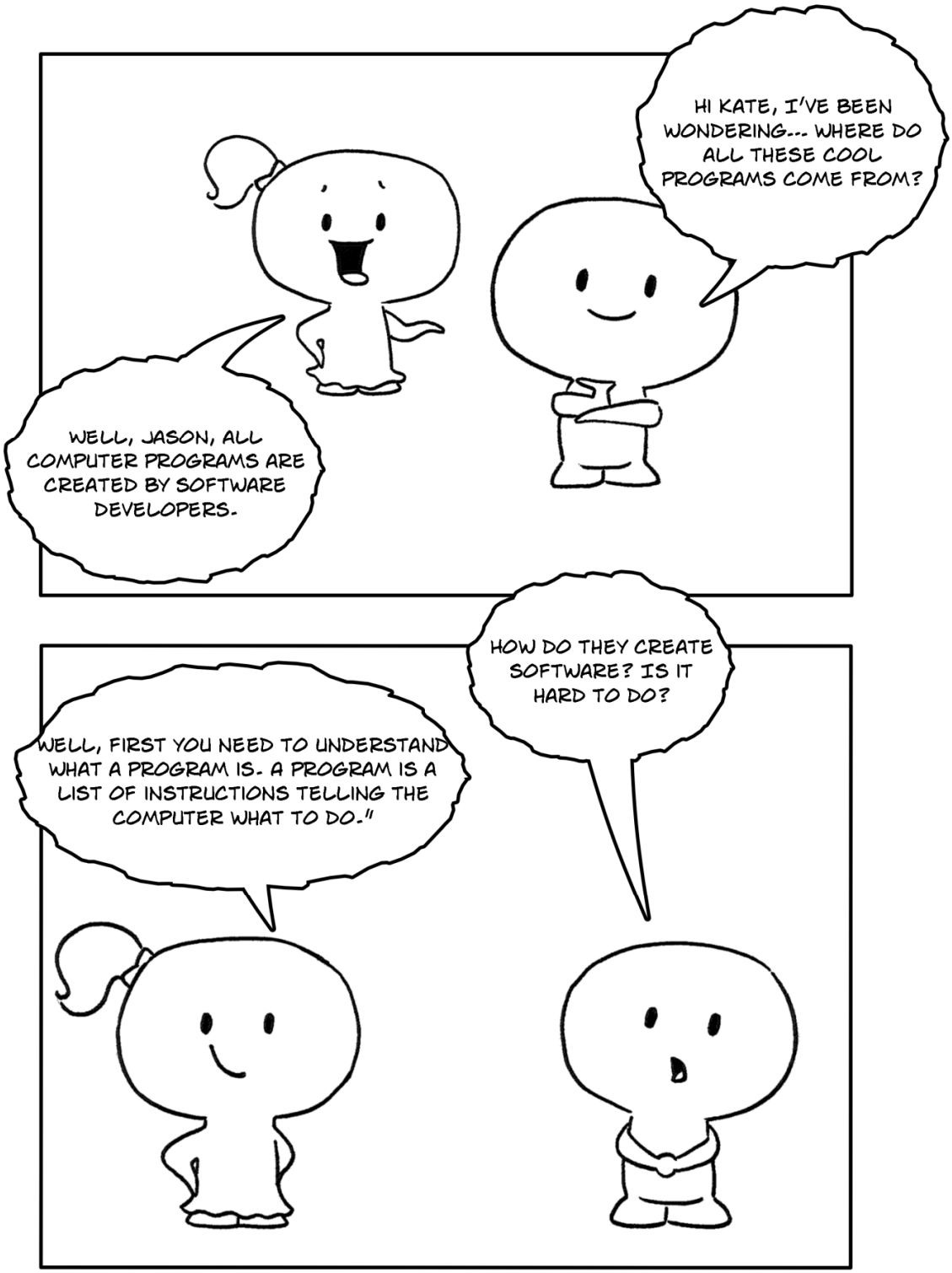
Chapter 1

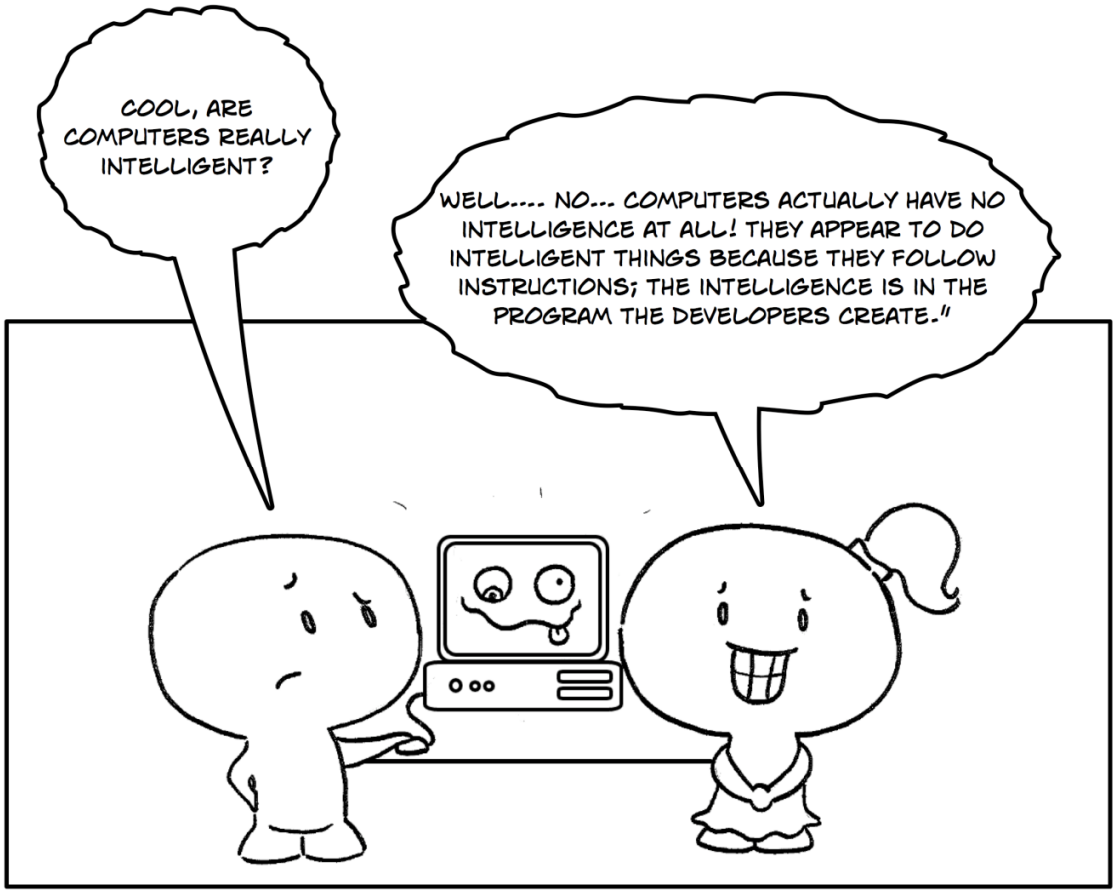
"Hello World"

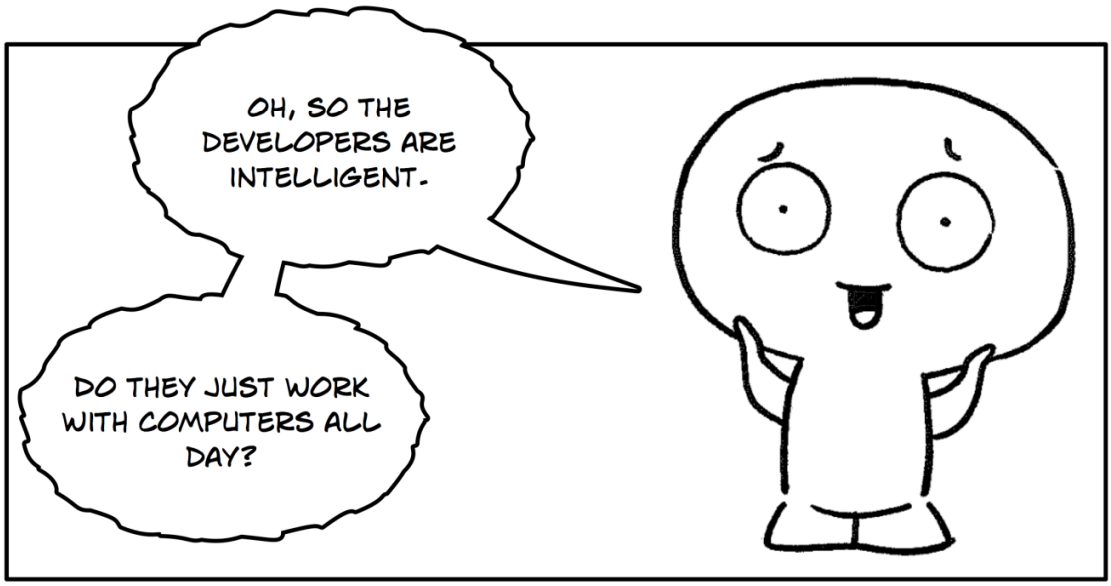
Summary:

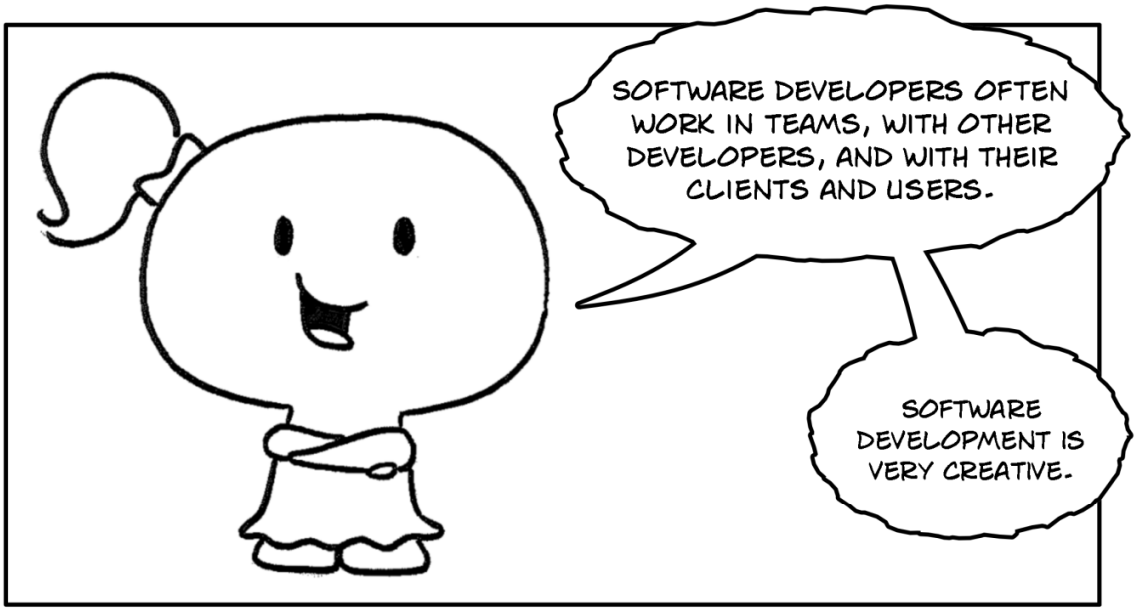
In this chapter you will perform basic actions with “Hello World” program such as changing color and position of elements on the screen, and drawing a smiley face.

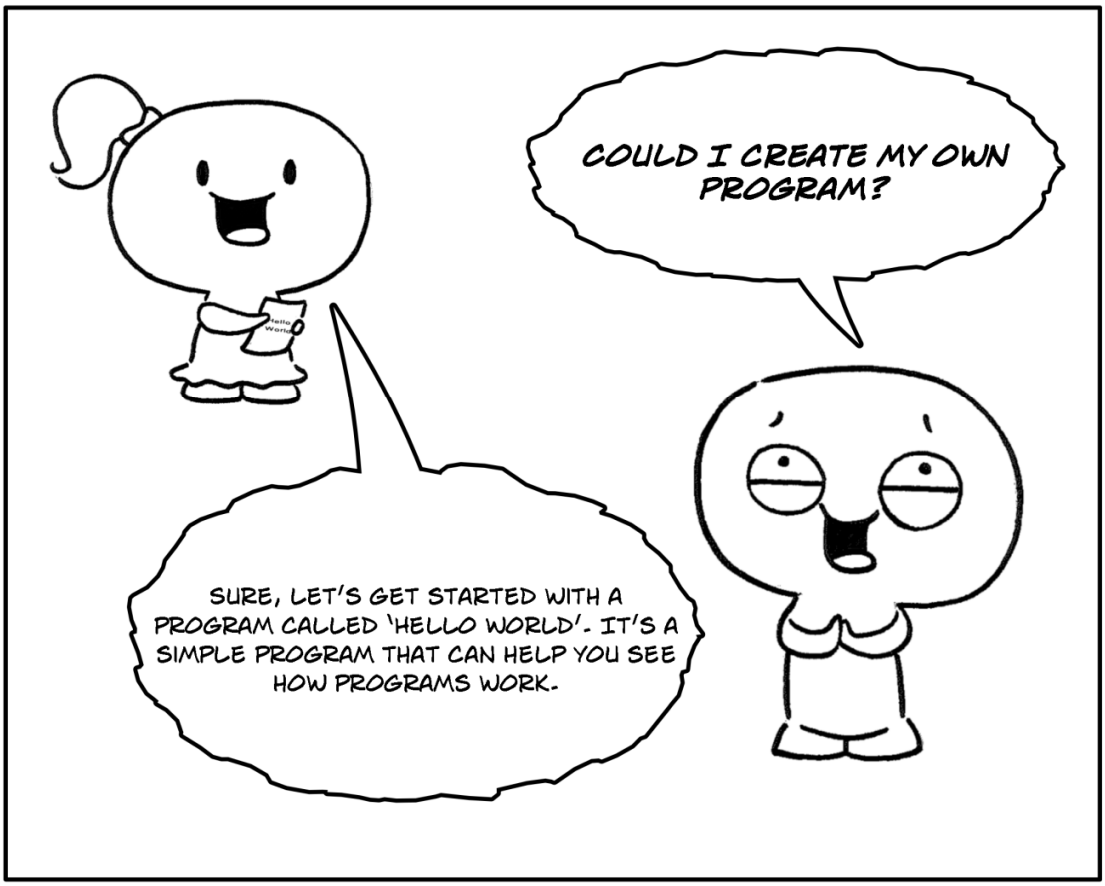
To start you need to have Visual Studio, which has to be set up for Visual Basic .NET, a template that has been provided for this project and it can also be downloaded from www.swingame.com . The template should be copied into Documents -> Visual Studio 2005/2008 -> Templates -> Project Templates.











Part 1

"Hello World" program is the program to start learning any programming languages from. This is the simple program which prints the "Hello World!" message on the screen. It is used to explain the computer principles for a particular programming language. It also helps to check whether the Swingame project set up correctly. Follow these steps to get a "Hello World" program working with SwinGame.

1. Open Visual Studio, and click File > New > Project. As shown in Figure 1.

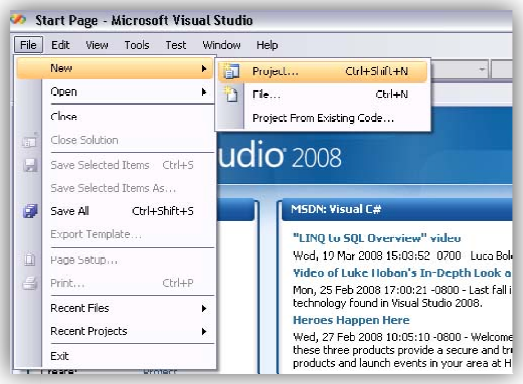


Figure 1

2. Select the Visual Basic/Basic Language and click the SwinGame VB.NET Project Template. Name new project as "HelloWorld". Click OK. The process is shown in Figure 2.

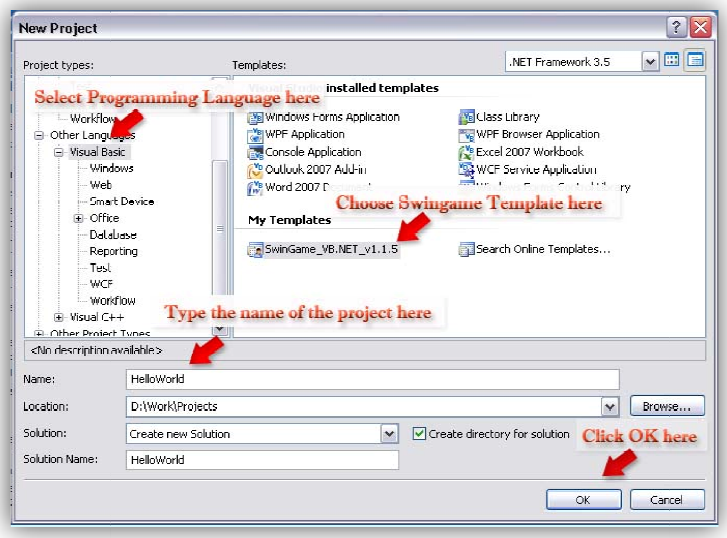
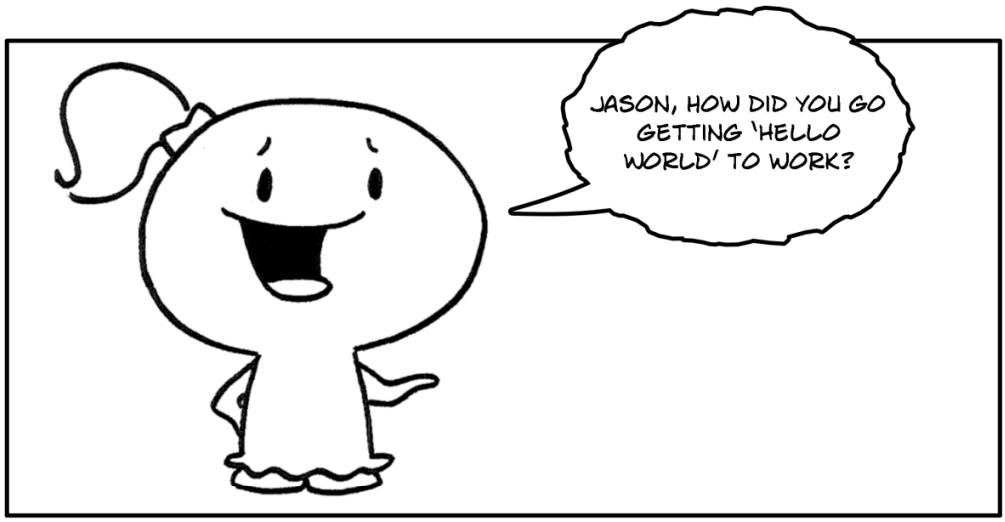
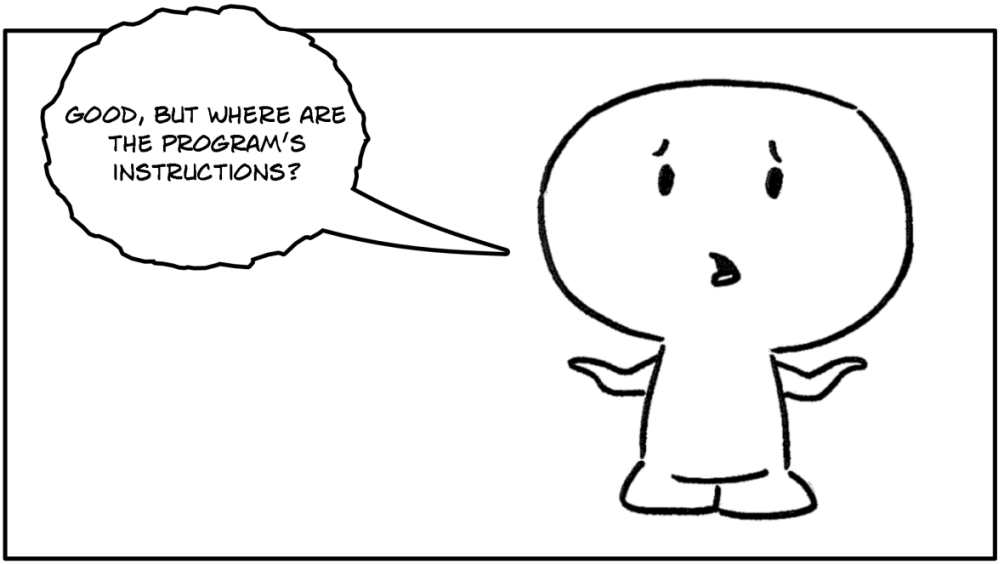


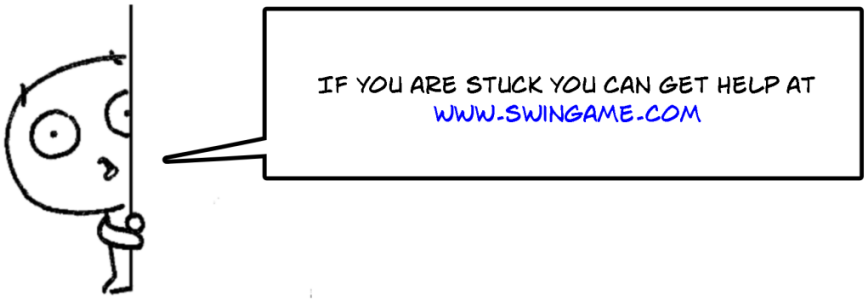
Figure 2

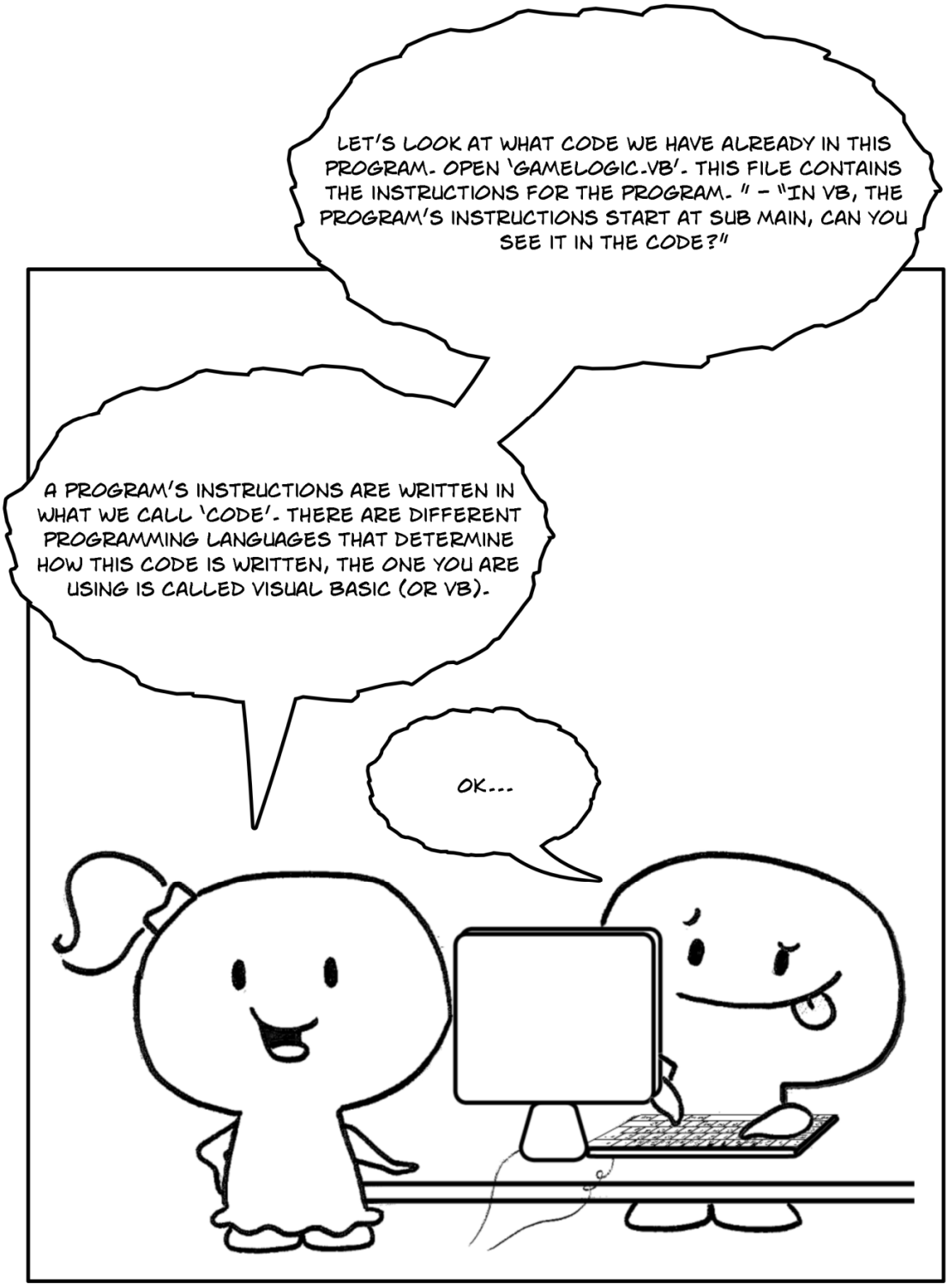
3. Press the "StartDebugging" button at the top of the screen (looks like a green arrow , F5 works too) to see what it does.

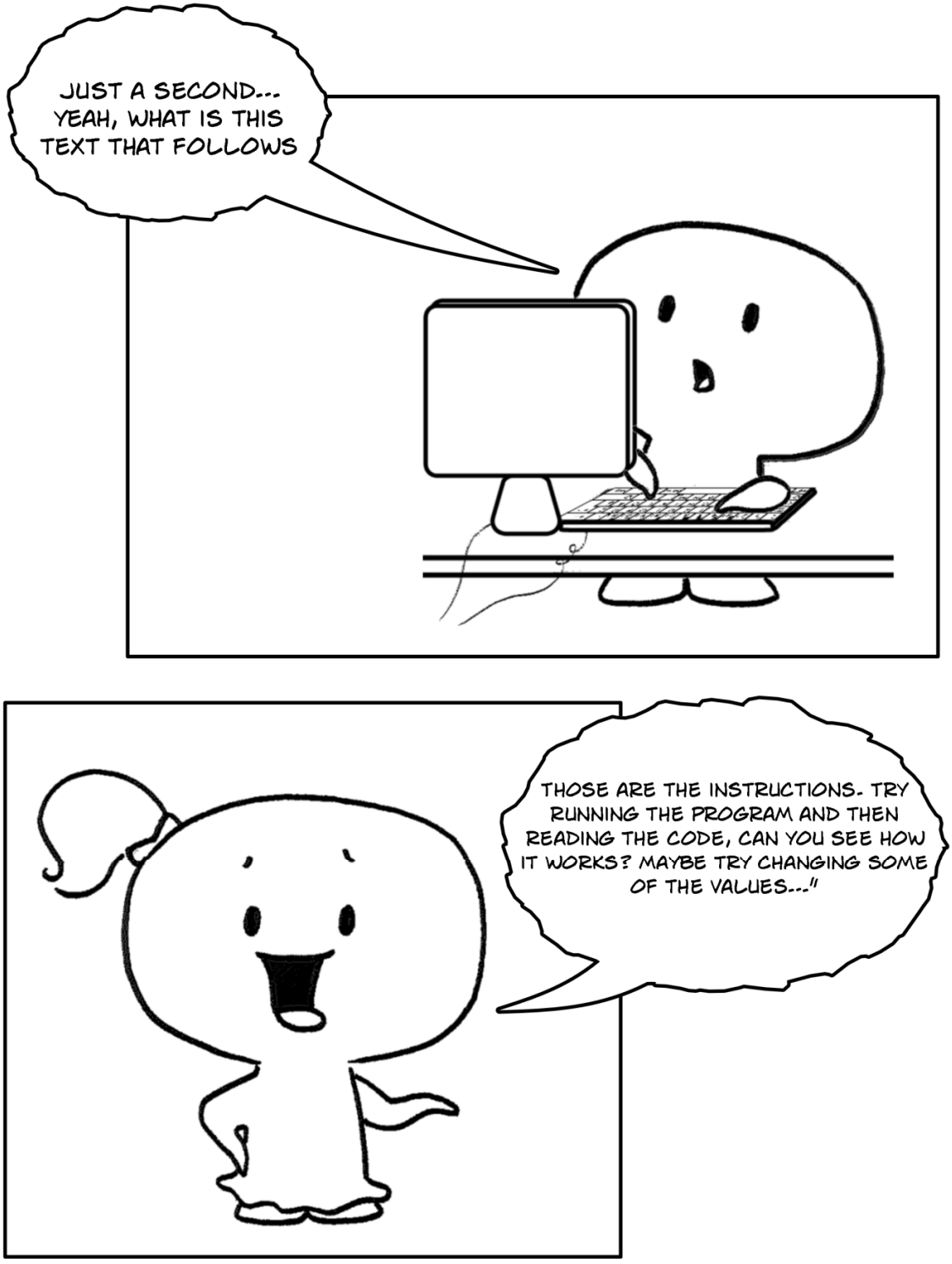
What is happening on your screen? Answer on the worksheet.











Part 2

In the Solution Explorer on the right hand side of the screen double click on "GameLogic", as shown in Figure 3. You should be presented with the following code:

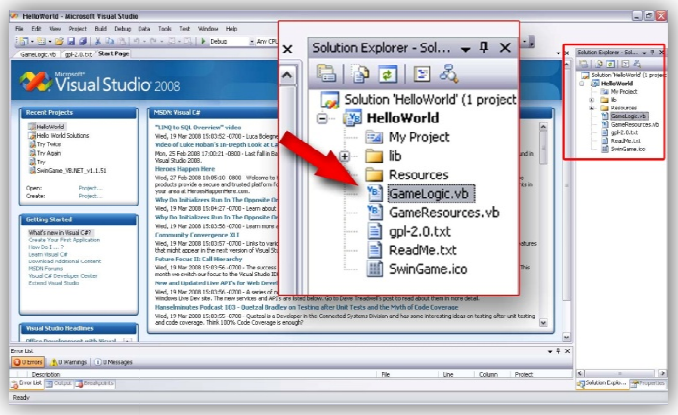


Figure 3

Code to be shown:

```
Module GameLogic
    Public Sub Main()
        'Opens a new Graphics Window
        Core.OpenGraphicsWindow("Game", 800, 600)

        'Open Audio Device
        Audio.OpenAudio()

        'Load Resources
        LoadResources()

        'Game Loop
        Do
            'Clears the Screen to Black
            SwinGame.Graphics.ClearScreen()

            'Draws rectangle on the screen
            Graphics.FillRectangle(Color.Red, 20, 150, 500, 50)

            'Draws text on the screen
            Text.DrawText("Hello World!", Color.Aqua, GameFont("ArialLarge"), 50, 50)

            'Refreshes the Screen and Processes Input Events
            Core.RefreshScreen()
            Core.ProcessEvents()

        Loop Until SwinGame.Core.WindowCloseRequested() = True

        'Free Resources and Close Audio, to end the program.
        FreeResources()
        Audio.CloseAudio()

    End Sub

End Module
```

The purpose of this code is to create a starting point for your project. If you press the "StartDebugging" button at the top of the screen (looks like a green arrow  , F5 works too) you will see what it does. Have a look, and then close the window.

You can basically read what it is doing `'comments are in green with a single parenthesis at the start like this sentence`. Every line in the program is doing something, but the most important are subs (i.e. `Public Sub Main()... End Sub`) and sub calls (i.e. `Graphics.FillRectangle(rectangle color, Xpos, Ypos, Width, Height)` or `Text.DrawText("Text To Draw", Color, GameFont("FontName"), Xpos, Ypos)`).

The Game Loop is a piece of code that is repeating to execute until the game window is closed. It is the actual "working" part of your game - it draws elements on the screen and provides the main functionality of the game. In our case, the game loop starts from `Do` keyword followed by some code to execute continuously and ends `Loop Until SwinGame.Core.WindowCloseRequested() = True`, that tells the program to stop executing of the code when the window is closed.
Do loop in VB.NET keeps executing statements while or until the condition is true. It has to be executed at least once, because the test occurs at the end of the loop. Here is Nassi-Shneiderman (N-S) Diagram for the game loop:

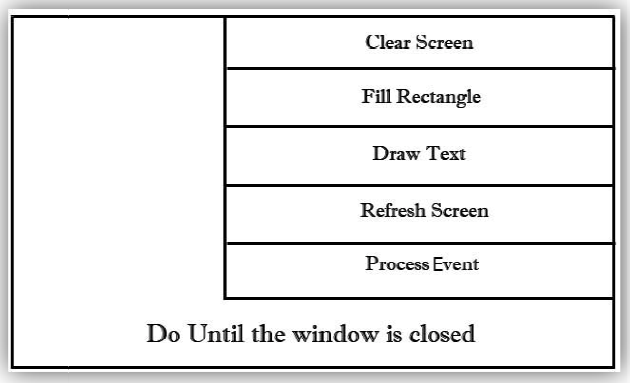


Figure 4. Game Loop

Did you know:

A Nassi-Shneiderman diagram (or NSD) is a graphical design representation for structured programming. Developed in 1972 by Isaac Nassi and Ben Shneiderman, these diagrams are also called structograms, as they show a program's structures.

Exercise1: *Reading the code*

Answer the following questions on the worksheet:

a) Which line number contains the code that tells the computer to draw text on the screen?
b) Which line number contains the code that tells the computer to draw a rectangle on the screen?
c) Where (line number) does the game loop start and end?

NOTE: If there are no line numbers on the left side of the window, add them manually:

- Go to tools -> Options (Figure 5)

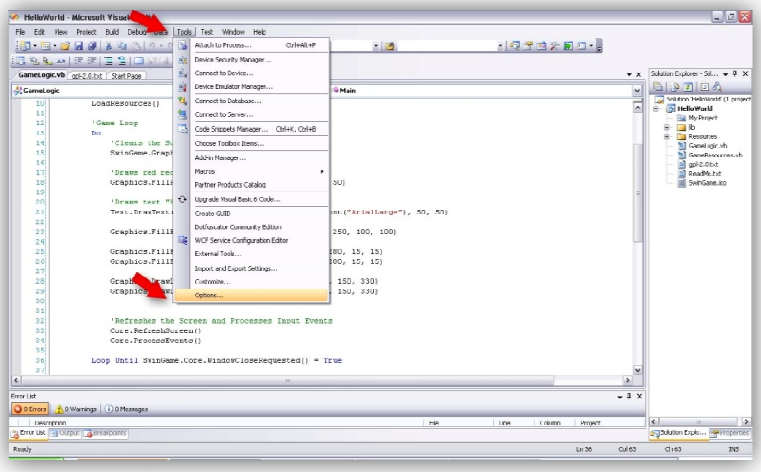


Figure 5

- Choose Text Editor -> All Languages -> General
- Check Display Section -> Line Numbers (Figure 6)

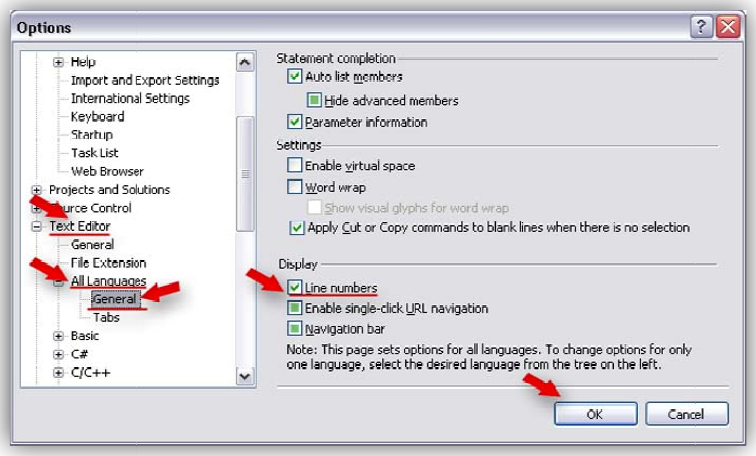


Figure 6

Exercise 2: *Changing the text*

Make the following changes in your *Hello World* program and write your solutions onto the worksheet:

a) Change the text "Hello World!" to "Hello *Your Name*!"

Hints: When mouse is over the sub call (i.e. LoadResources()), the pop up window will appear which contains the list of parameters that this function takes and their order along with function description (Figure 7).

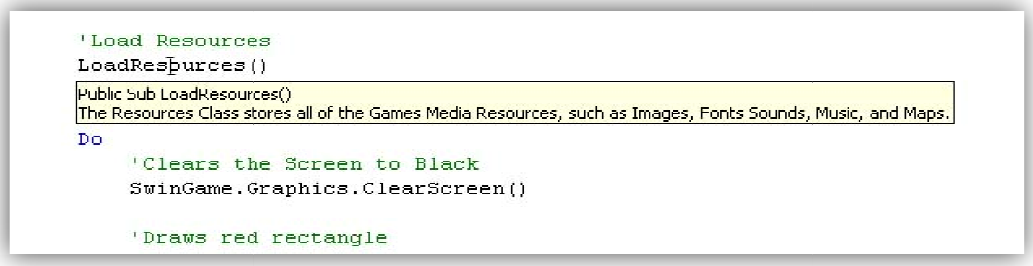


Figure 7

Exercise 3: *Changing the color*

Make the following changes in your *Hello World* program and write your solutions onto the worksheet:

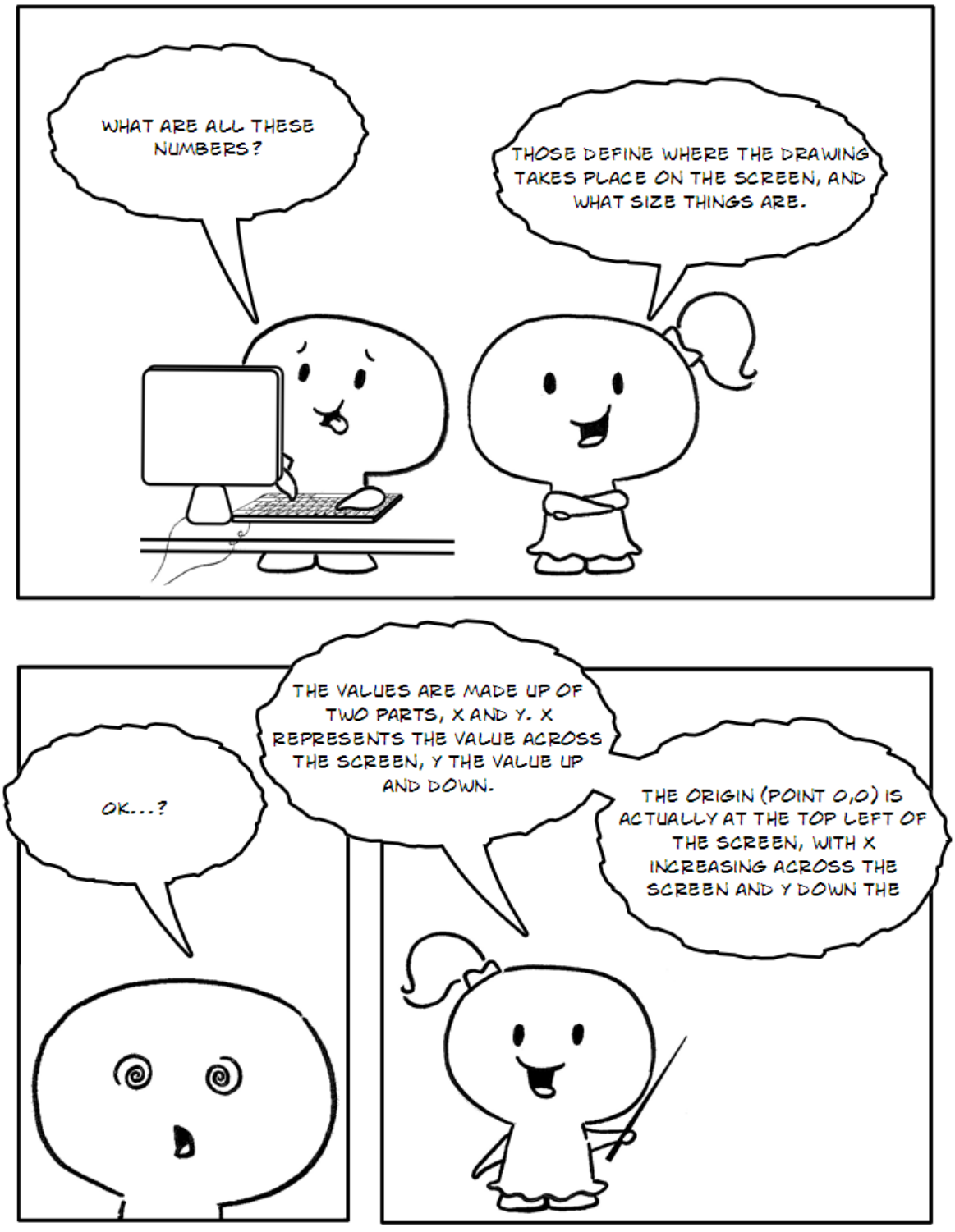
NOTE: Spelling is important. In this case, you MUST use American spelling - COLOR, not Colour. Otherwise, it will produce an error message and prevent your program from executing.

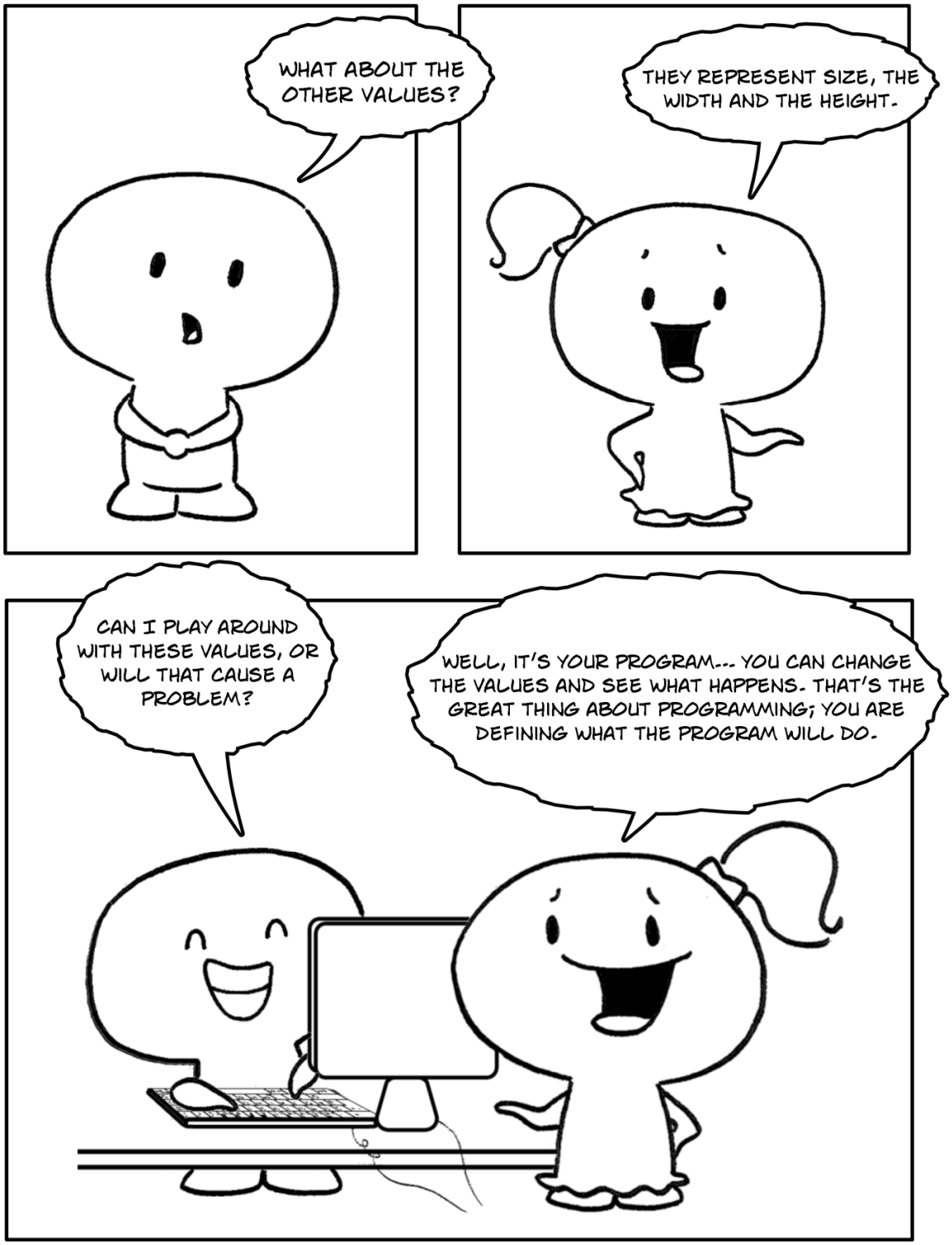
a) Change the color of the text to `Color.GreenYellow`.

Hints: When you type `Color.` the list of possible colors will appear.

b) Change the color of the rectangle to `Color.Blue`.
c) Change the background color to `Color.White`.

NOTE: The `ClearScreen()` sub clears the screen to black color by default. To customize the background color, simply type "`Color.`" followed by color name (`Swingame.Graphics.ClearScreen(Color.White)`.





Part 3

Exercise 1: *Locations on the screen*

Draw the following shapes and text onto your worksheet.

a) On your worksheet, draw a small rectangle with coordinates X = 15 and Y = 5 by hand.
b) Draw the text "Hello *Your Name*" at X = 5, Y = 20.

Hints: The location of each element is determined by its X and Y coordinates. X and Y, in this case, are coordinates of the top left corner of each element as shown in Figure 8.

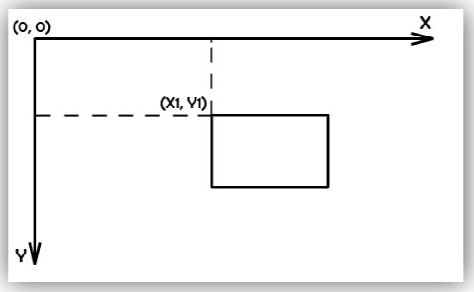


Figure 8. Coordinates

Exercise 2: *Changing drawing locations*

Make the following changes to your *Hello World* program and write your solutions onto the worksheet:

a) Change the location of the text; put it in the middle of the screen.
b) Change the location of the rectangle; put it under the text.

NOTE: The size of SwinGame screen is 800 pixels wide (X Axis) and 600 pixels tall (Y Axis) (Figure 9).

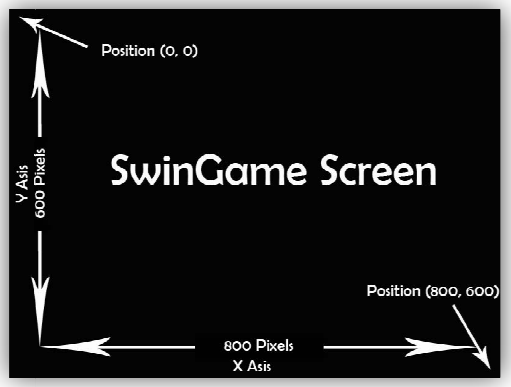


Figure 9

Exercise 3: *Size of an element*

Draw the following shapes and text onto your worksheet.

a) Draw a rectangle with width = 5, height = 10 at the position X = 0, Y = 5 by hand.
b) Draw a rectangle with width = 10, height = 5 at the position X = 5, Y = 10 by hand.

Exercise 4: *Draw an element with the new size.*

Make the following changes to your *Hello World* program and write your solutions onto the worksheet:

a) Change the size of the rectangle to width = 630 and height = 20.

Hints: The size of any element, such as a rectangle or a circle, is declared by using width and height parameters (Figure 10).

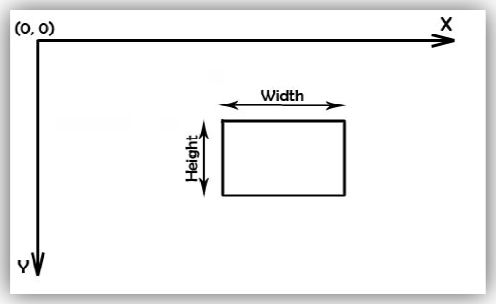
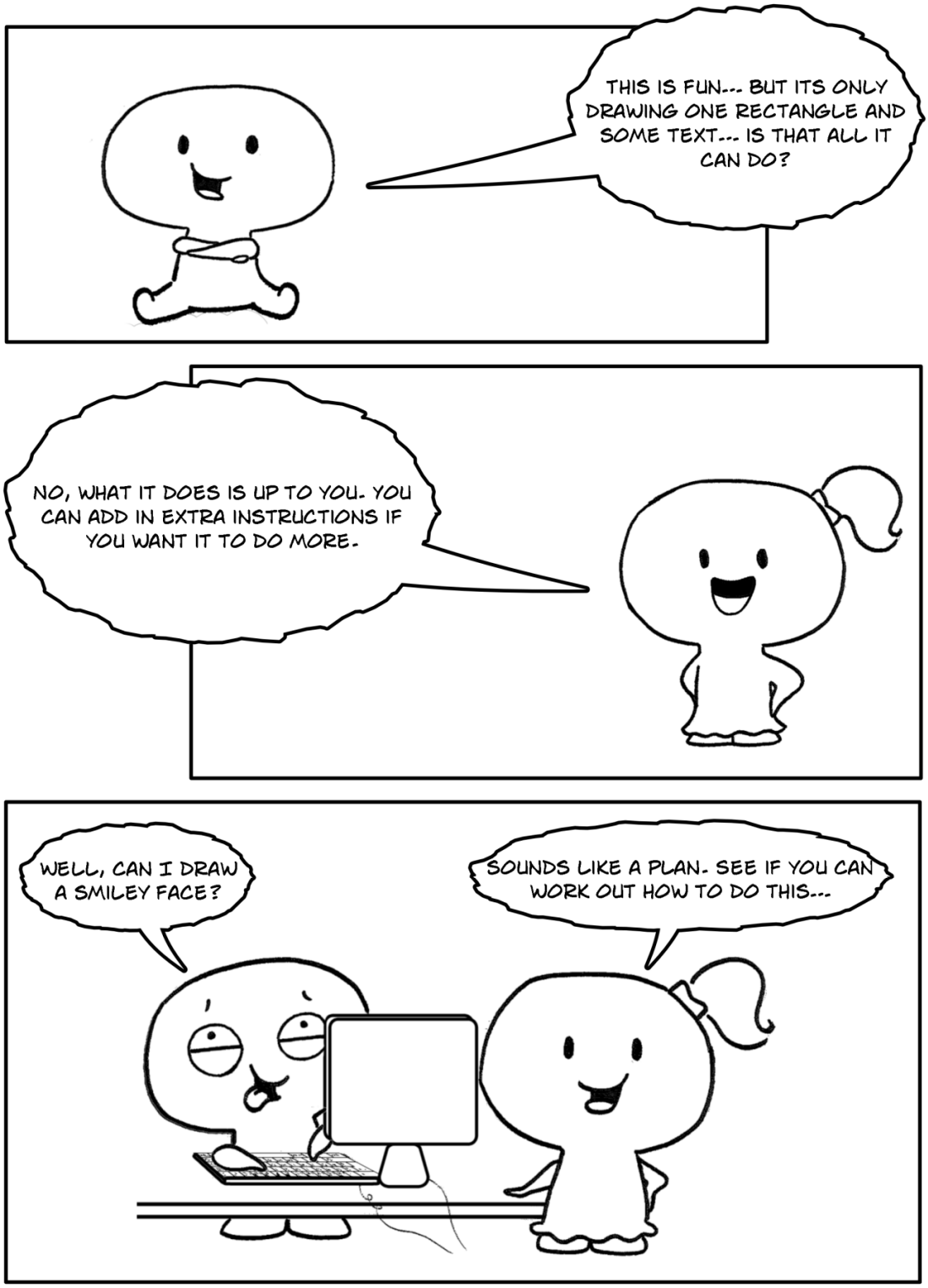


Figure 10. Size



Extra Exercise:

Draw "Smiley" face on the screen then put the code you entered to the worksheet:

1) Draw a yellow circle on the screen – "face". Use
   `Graphics.FillEllipseOnScreen(Color, Xpos, Ypos, Width, Heigh).`
   For example, `Graphics.FillEllipseOnScreen(Color.Yellow, 100, 250, 100, 100)`

2) Draw the eyes – two black circles inside the yellow circle. Use the same sub call to draw, i.e.:
   ```
   Graphics.FillEllipseOnScreen(Color.Black, 125, 280, 15, 15)
   Graphics.FillEllipseOnScreen(Color.Black, 160, 280, 15, 15)
   ```
3) Draw the "smile" on the screen.
   Tell the computer to draw two lines which are connected at the bottom, inside the yellow circle. To do so, use `Graphics.DrawLineOnScreen(Color, XPosStart, YPosStart, XPosEnd, YPosEnd,` i.e.:
   ```
   Graphics.DrawLineOnScreen(Color.Black, 130, 310, 150, 330)
   Graphics.DrawLineOnScreen(Color.Black, 170, 310, 150, 330)
   ```

   NOTE: There is no arc line in SwinGame, "smile" will be made by two lines. The smile ` will look as in Figure 11.

Figure 11

IMPORTANT!!!

Please, save exercises for each chapter in a folder named as "Chapter X", where X is the number of the chapter. Put "Chapter X" folder into My Documents\Your Full Name.