## Release Notes (3.19, 2019?)

### Screen.GX and Screen.GY

You can now get the graphic size of the terminal screen with the Screen.GX and Screen.GW values. These are useful to make a graphics display that’s neatly laid out on the terminal screen.

In the example, text is placed on the left side of the screen and a graphical area is on the right. The graphics position includes the border, but the size does not.

REM Make a small text area on the left

REM and a graphics area on the right

CLS WHITE BLACK

PRINT AT 1,1 "\*---5----\*----5----\*----5----\*----5"

PRINT AT 2,1 "\*--------1---------2---------3-----"

PRINT AT 4,1 "This text area is on the left"

PRINT AT 5,1 "Screen.W", Screen.W

PRINT AT 6,1 "Screen.H", Screen.H

PRINT AT 8,1 "The graphics area is on the right"

PRINT AT 9,1 "Screen.GW", Screen.GW

PRINT AT 10,1 "Screen.GH", Screen.GH

PRINT AT 16, 1"Line 16"

REM The graphics screen is placed based on the

REM outer position of the graphics (including the

REM border area). The screen size is based on

REM the inner area.

REM Hence the need for account for the padding.

PADDING = 1

X1 = Screen.GW/2 - PADDING

Y1 = 0

W = Screen.GW - X1 - PADDING\*2

H = Screen.GH - Y1 - PADDING\*2

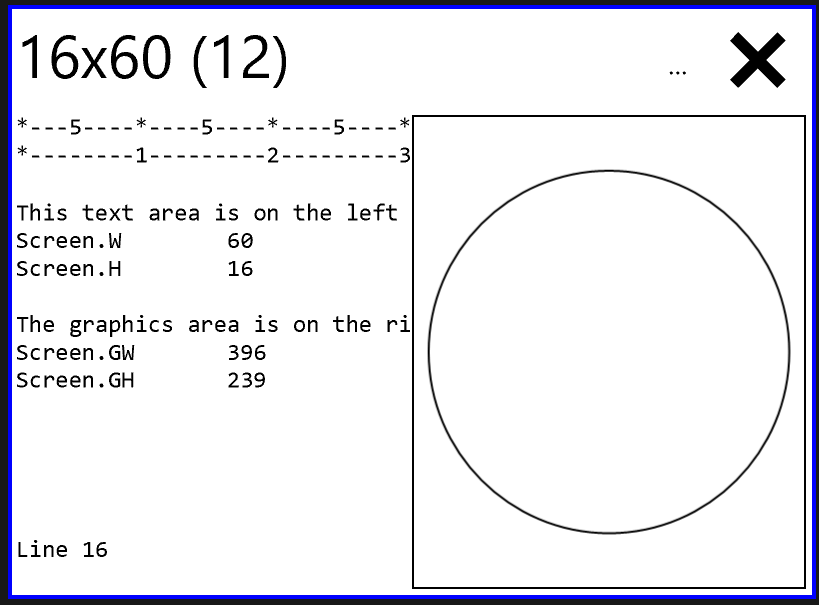
g= Screen.Graphics (X1, Y1, H, W)

g.Border = BLACK

g.Fill = WHITE

g.Circle (W/2, H/2, Math.Min (W/2, H/2)-5)

The output looks like this:



The output will always have the same proportions regardless of the characters size of the screen or the size of the font.

### Skoobot is now supported!

The Skoobot is a tiny Bluetooth-controllable robot from <https://www.william-weiler-engineering.com/> . Sample programs include the simplest possible program to set the robot into Rover mode, a keyboard-driven robot program, and a GUI program.

Use device = Bluetooth.PickDevicesName("Skoobot\*") to let the user select one particular Skoobot. Then to get a Skoobot specialization, use the device.As method, passing in “Skoobot” like this: skoobot = device.As ("Skoobot")

Available Skoobot methods are

|  |  |
| --- | --- |
| **Category** | **Methods** |
| Raw Motion Commands | Left30()  Right30()  Forward()  Backward() |
| Stop | Stop() |
| Sounds | PlayBuzzer() |
| AI modes | RoverMode()  RoverModeRev()  FotovoreMode() |

For example, to have the Skoobot move forward, once you have a skoobot specialization, just call skoobot.Forward()

These correspond directly to the underlying Bluetooth commands. The Skoobot primary commands are using service 00001523-1212-efde-1523-785feabcd123 and characteristic 00001525-1212-efde-1523-785feabcd123. You might be familiar with these as they are the standard GUIDs for the Nordic Semiconductor nRF Blinky app. The meaning of the command byte has been extended so to include the Skoobot commands

|  |  |
| --- | --- |
| **Skoobot Command** | **Command Value** |
| Right30 | 0x08 |
| Left30 | 0x09 |
| Right | 0x10 |
| Left | 0x11 |
| Forward | 0x12 |
| Backward | 0x13 |
| Stop | 0x14 |
| StopTurning | 0x15 |
| MotorsSleep | 0x16 |
| PlayBuzzer | 0x17 |
| RoverMode | 0x40 |
| FotovoreMode | 0x41 |
| RoverModeRev | 0x42 |

The Skoobot specialization understands all these commands.

There are complete Skoobot sample programs including

**“A first control program**” which is the simplest possible real Skoobot program. It demonstrates how to connect to a Skoobot and to get a Skoobot specialization.

**“Control program for Skoobot”** is a GUI control program

**“Keboard-driven Skoobot program”** is a command-line program for the Skoobot.

The first control program is:

CLS

PRINT "SKOOBOT CONTROL PROGRAM"

PRINT "Sets the Skoobot into Rover mode"

PRINT "Will automatically stop after 5 seconds"

device = Bluetooth.PickDevicesName ("Skoobot\*")

IF (device.IsError) THEN

PRINT "No device selected"

STOP

END IF

REM get the specialization of the device

skoobot = device.As ("Skoobot")

skoobot.RoverMode()

REM Run for about 5 seconds

PAUSE 50\*5

skoobot.Stop()

STOP