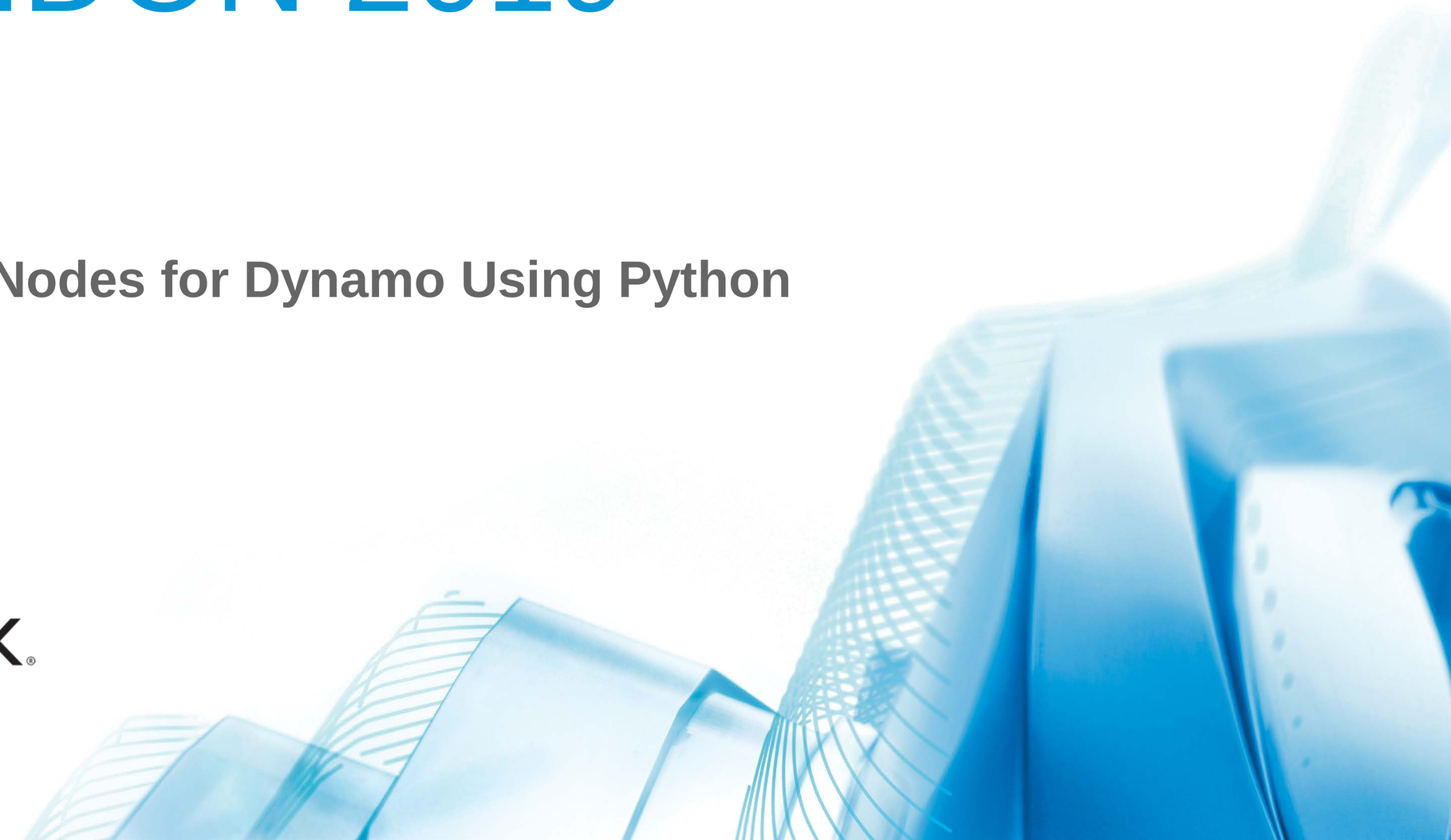
AU LONDON 2019

Ben Robinson

How to Build GUI Nodes for Dynamo Using Python

HANDOUT



AUTODESK. UNIVERSITY

Key Resources

ZETCODE.COM

Step by step instructions on building some WinForms in Iron Python

http://zetcode.com/tutorials/ironpythontutorial/introduction/

VOIDSPACE.ORG

Similar to Zetcode, with a few more methods and controls covered

http://www.voidspace.org.uk/ironpython/winforms/index.shtml

MICROSOFT WINFORMS DOCUMENTATION

The full documentation of WinForms. Examples of all available methods etc. Written in C#

https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.form?view=netframework-4.8

DYNAMO FORUM

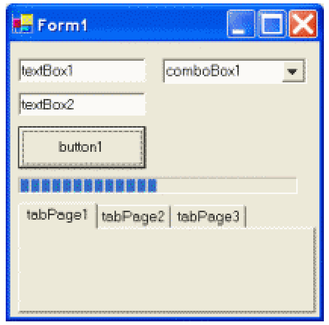
Its not a popular topic, but there are a lot of people on the forum including myself and the Data-Shapes team that can help

https://forum.dynamobim.com/search?q=winform

Key Resources

Dock & Anchor

IronPython & Windows Forms, Part VII



Note

This is part of a series of tutorials on using IronPython with Windows Forms.

```
import clr
clr.AddReference('System.Windows.Forms')

from System.Windows.Forms import Application, Form, Button, DockStyle

class MainForm(Form):
    def __init__(self):
        for i in range(1, 6):
            btn = Button()
            btn.Text = "Button %s" % i
            btn.Dock = DockStyle.Top
            self.Controls.Add(btn)


Application.EnableVisualStyles()
form = MainForm()
Application.Run(form)
```

Key Resources

ZETCODE.COM

Step by step instructions on building some WinForms in Iron Python

http://zetcode.com/tutorials/ironpythontutorial/introduction/

VOIDSPACE.ORG

Similar to Zetcode, with a few more methods and controls covered

http://www.voidspace.org.uk/ironpython/winforms/index.shtml

MICROSOFT WINFORMS DOCUMENTATION

The full documentation of WinForms. Examples of all available methods etc. Written in C#

https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.form?view=netframework-4.8

DYNAMO FORUM

Its not a popular topic, but there are a lot of people on the forum including myself and the Data-Shapes team that can help

https://forum.dynamobim.com/search?q=winform

Key Terminology

CLASS & OBJECT

A blueprint created by a programmer for an object. This defines a set of attributes that will characterize any object that is instantiated from this class.

```
#In WinForms, any window or a dialog is a Form.
class DropDownForm(Form):

    def __init__(self):     #the __init__ method inside a class is its constructor

        self.Text = "AU London"      #text that appears in the GUI titlebar
        self.Icon = Icon.FromHandle(icon.GetHicon()) #takes a bitmap image and conve
```

An Object is simple an instance of a class

```
ddForm = DropDownForm()
```

```
#combox drop down
cBox = ComboBox()   #dropdown control form
```

Key Terminology

FUNCTION & DEFINITION

A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for your application and a high degree of code reusing

A definition Is simple a user defined function

```
def okButtonPressed(self, sender, args):
    self.Close()    #trigger to close the GUI when button is pressed
    self.runNextOutput = True #if the ok button is pressed set runNextOutput as True
```

```
btnOk.Anchor = (AnchorStyles.Bottom | AnchorStyles.Right)
btnOk.Click += self.okButtonPressed  #Register the event on the button bress to trigger the def
```

Key Terminology

METHOD & PROPERTY

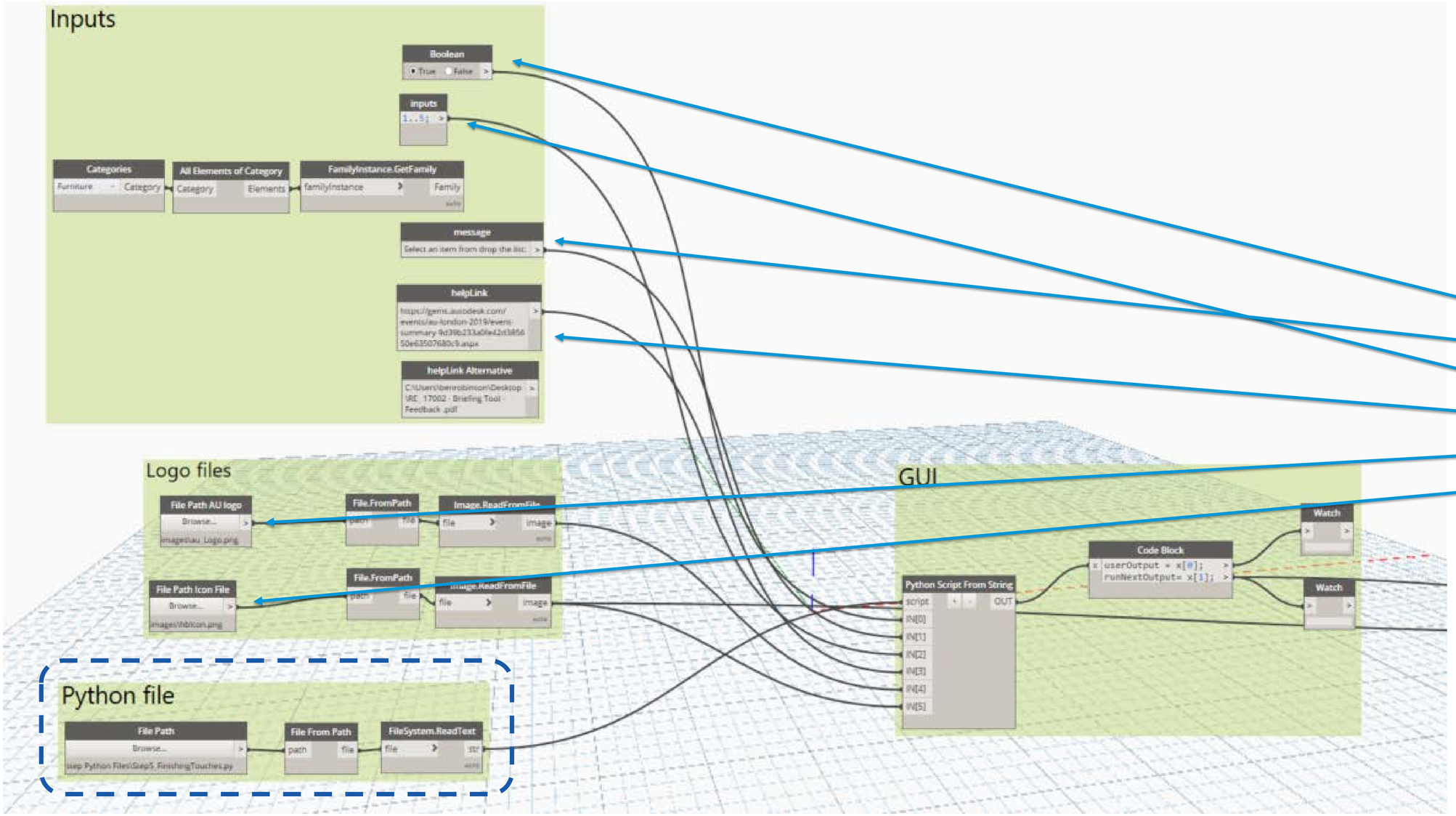
Similar to a function, it called by its name, but it is implicitly associated with an object/class.

https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.control.controlcollection.addrange?view=netframework-

```
cBox.Width = uiWidth -(spacing*2)
cBox.Items.AddRange(listInput)    # Adds an array of items to the list o
cBox.DropDownStyle = ComboBoxStyle.DropDownList    #setting to dropdown
```

Properties : controllable attributes of an object

Dynamo Setup




```
##############
#INPUTS HERE:

run = IN[0]
message = IN[1]
listInput =tuple(IN[2])
linkaddress = IN[3]
logoFile = IN[4]
icon = IN[5]
```

1: The Boiler Plate & Inputs

A Boiler plate is essentially code that can be reused without context in new applications

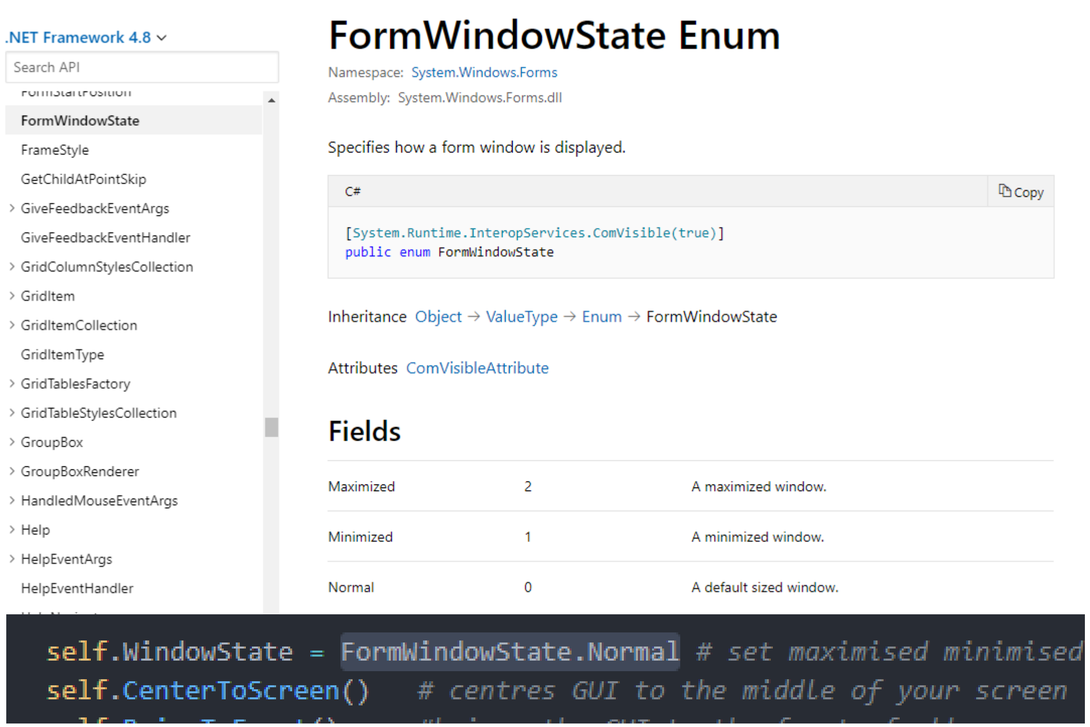
- Not all references are required to build a standalone GUI
  - I keep them incase I will need them further down the line.
  - Then delete unused when imports when it is finished

```
############-------------\-------------############
#My Default Boiler Plate
import clr
clr.AddReference('ProtoGeometry')
from Autodesk.DesignScript.Geometry import *

clr.AddReference('RevitAPIUI')
from  Autodesk.Revit.UI import Selection

clr.AddReference("RevitAPI")
import Autodesk
from Autodesk.Revit.DB import *

clr.AddReference('RevitNodes')
import Revit
clr.ImportExtensions(Revit.Elements)
clr.ImportExtensions(Revit.GeometryConversion)

clr.AddReference('RevitServices')
import RevitServices
from RevitServices.Persistence import DocumentManager
from RevitServices.Transactions import TransactionManager
from System.Collections.Generic import *

import sys
pyt_path = r'C:\Program Files (x86)\IronPython 2.7\Lib'
sys.path.append(pyt_path)

import math
doc = DocumentManager.Instance.CurrentDBDocument
uiapp = DocumentManager.Instance.CurrentUIApplication
app = uiapp.Application

############-------------\-------------############
#INPUTS HERE:


TransactionManager.Instance.EnsureInTransaction(doc)

# "End" the transaction
TransactionManager.Instance.TransactionTaskDone()
```

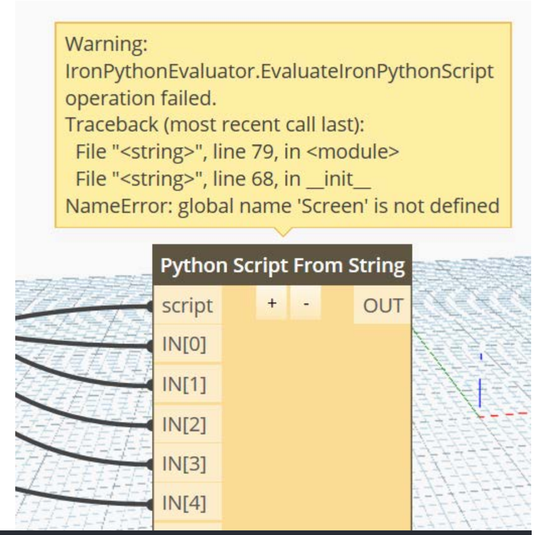
2: Hello World GUI

- 1: Add the WinForms and drawing assemblies
- 2: import the individual references we need now
- 3: For the main form create a class called “DropDownForm” this will use the Form class to create our GUI
- 4: __init__“ is a constructor in Python classes. Its called to create the GUI object and initiate the forms attributes
- 5: self represents the instance of the class. By using the "*self*" keyword we can access the *attributes* and *methods* of the class such as .Text which sets the title of the GUI
- 6: Create an instance of our DropDownForm() class
- 7: Run the application

```
import sys
pyt_path = r'C:\Program Files (x86)\IronPython 2.7\Lib'
sys.path.append(pyt_path)

import math
doc = DocumentManager.Instance.CurrentDBDocument
uiapp = DocumentManager.Instance.CurrentUIApplication
app = uiapp.Application

##########-----------\-----------##########
#UI additional references
clr.AddReference("System.Windows.Forms")  1
clr.AddReference("System.Drawing")

from System.Windows.Forms import Application, Form  2

##########-----------\-----------##########
#INPUTS HERE:

run = IN[0]
message = IN[1]
listInput =tuple(IN[2])     #Combo box requires tuple not list input
url = IN[3]
logoFile = IN[4]
icon = IN[5]


# create a instance of the form class called DropDownform.
#In Winforms, any window or a dialog is a Form.
class DropDownForm(Form):  3
    def __init__(self):  4 #the __init__ method inside a class is its constructor
        self.Text = "AU London"  5 #text that appears in the GUI titlebar


ddForm = DropDownForm()  6

if run:     #if input is true run the application.
    Application.Run(ddForm)  7
```




```
self.WindowState = FormWindowState.Normal # set maximised minimised
self.CenterToScreen()    # centres GUI to the middle of your screen
```




```
clr.AddReference("System.Drawing")

from System.Windows.Forms import Application, Form, FormWindowState#, Screen
from System.Drawing import Icon, Color
```

Tip 1

To find a list of all properties and methods of the Form Class check the below documentation

https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.form?view=netframework-4.8

Tip 2

If you use a control and get the above error it will be because you have not imported the control from Windows.Forms

3: Basic Form Controls

- 1: GUI require widgets/controls: ways of receiving inputs (buttons etc.) When we use any control it must be imported.
- 2: Get a bitmap image from at the input path and use it as the GUI icon
- 3: These methods ensure the GUI comes to the front your screen and is scaled normally
- 4: Get the size of the user's screen and make the GUI ¼ dimension of the screen.
- 5: *FormBorderStyle = FormBorderStyle.FixedDialog stops the form from being scaled in size by a user. Disable this for testing incase controls disappear*
- *6: Create 2 output values to store the values of the item selected and run control*

```python
from System.Windows.Forms import Application, Form, FormWindowState, Screen
from System.Drawing import Icon, Color

############-------------\-------------############
#INPUTS HERE:

run = IN[0]
message = IN[1]
listInput =tuple(IN[2])     #Combo box requires tuple not list input
url = IN[3]
logoFile = IN[4]
icon = IN[5]

userOutputDefaultStr = "No selection made, Re-run, and select an item from the dropdown menu" #set

# create a instance of the form class called DropDownform.
#In Winforms, any window or a dialog is a Form.
class DropDownForm(Form):

    def __init__(self):     #the __init__ method inside a class is its constructor

        self.Text = "AU London"      #text that appears in the GUI titlebar
        self.Icon = Icon.FromHandle(icon.GetHicon()) #takes a bitmap image and converts to a file t
        self.BackColor = Color.FromArgb(255, 255, 255)

        self.WindowState = FormWindowState.Normal # set maximised minimised or normal size GUI
        self.CenterToScreen()   # centres GUI to the middle of your screen
        self.BringToFront()     #brings the GUI to the front of all opens windows.
        self.Topmost = True    # true to display the GUI infront of any other active forms

        screenSize = Screen.GetWorkingArea(self)  #get the size of the computers main screen, as th
        self.Width = screenSize.Width / 4  #set the size of the form based on the size of the users
        self.Height = screenSize.Height / 4
        uiWidth = self.DisplayRectangle.Width    #get the size of the form to use to scale form ele
        uiHeight = self.DisplayRectangle.Height

        #self.FormBorderStyle = FormBorderStyle.FixedDialog      # fixed dialog stops the user from

        self.userOutput = userOutputDefaultStr  #create a container to store the output from the fo
        self.runNextOutput =  False  #set these default values

############-------------\-------------############




ddForm = DropDownForm()

if run:     #if input is true run the application.
    Application.Run(ddForm)
```

Setting out the GUI Controls

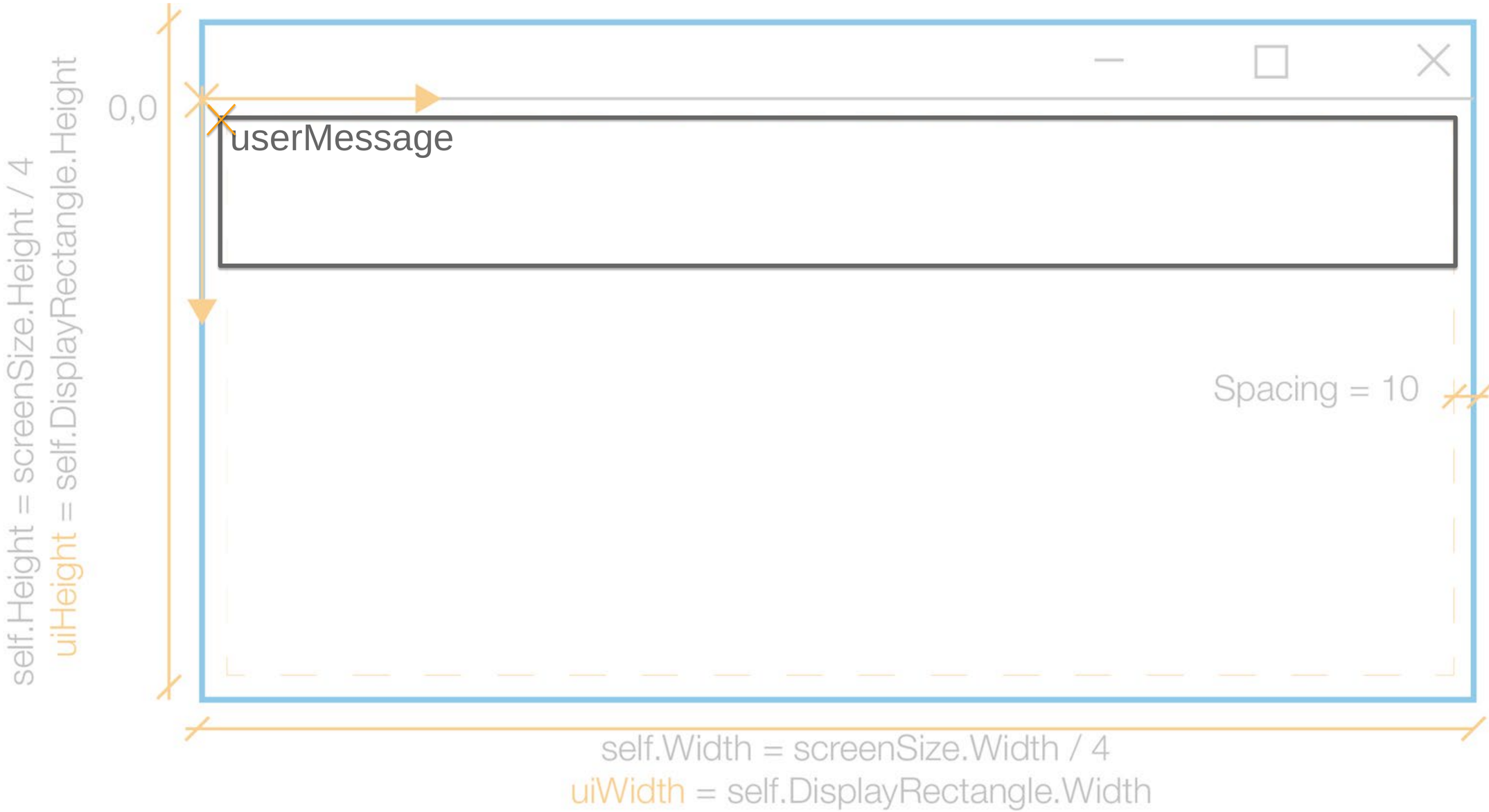
4: Adding Simple Controls

- 1: The label control allows text to be added to the GUI .
- 2: Point object is required in order to location a control (X,Y)
- 3: Size the label using uiWidth + uiHeight. This ensures is scales will the GUI for different type screens.
- 4: self.Controls.Add() Adds the control to the GUI
- 5: Get the ratio of logo height to width. Value must be a float as an int will round to the nearest 1.
- 6: SizeMode scale the image to fit the input size.
- 7: AnchorStyles lock the control to a given corner(s) this is needed if the user can change the size of the GUI

```
from System.Windows.Forms import Application, Form, FormWindowState, Screen, Label, PictureBox,
from System.Drawing import Icon, Color, Font, Point, Size
```

```
44
45  ##############------------\------------##############
46          spacing = 10    #spacing size for GUI elements to form a consistent
47
48          # creates the text box for a info message
49      1   userMessage = Label()   #label displays texts
50          font = Font("Helvetica ", 10)
51          userMessage.Text = message
52          userMessage.Font = font
53      2   userMessage.Location = Point(spacing, spacing)
54      3   userMessage.Size = Size(uiWidth-(spacing*2),(uiHeight/4))
55          self.Controls.Add(userMessage)
56      4
57
58  #////////////////////////////////////////////////////////////////////////////////
59          #logo file
60          logo =PictureBox()
61          logo.Image = logoFile
62      5   ratio =  float(logo.Height)/ float(logo.Width)  #needs to be a floa
63          logo.Size = Size(uiWidth/4, (uiHeight/4)*ratio) #scale the image by
64          logo.Location = Point(spacing, (uiHeight- logo.Height)-spacing)
65      6   logo.SizeMode = PictureBoxSizeMode.Zoom     # zooms the image to f
66      7   logo.Anchor = (AnchorStyles.Bottom | AnchorStyles.Left)      #ancho
67          self.Controls.Add(logo)
68          #logo.BorderStyle = BorderStyle.Fixed3D    #gives a border to the p
69
70
71
72
73  ddForm = DropDownForm()
74
75  if run:     #if input is true run the application.
76      Application.Run(ddForm)
77
78
```

Setting out the GUI Controls



5: ComboBox Control

- 1: ComboBox is a drop down and text input control
- 2: Add a list of items for the drop down using .AddRange()
- 3: ComboBoxStyle.DropDownList remove the ability for a user to input a text value into the control
- 4: SelectedIndexChanged+= registers the event handler of the user selecting an item from the control to then run the def dropDownOutput
- 5: self = instance of the form class (dropDownOutput)
  sender = the control object sending that raised the event
  Args = the argument/event from the sender
  assign the select item to userOutput variable
- 6: assign the output of ddForm.userOutput to results and output it

```
98          logo.Image = logoFile
99          ratio =  float(logo.Height)/ float(logo.Width)  #needs to be a floa
100         logo.Size = Size(uiWidth/4, (uiHeight/4)*ratio) #scale the image by
101         logo.Location = Point(spacing, (uiHeight- logo.Height)-spacing)
102         logo.SizeMode = PictureBoxSizeMode.Zoom     # zooms the image to fi
103         logo.Anchor = (AnchorStyles.Bottom | AnchorStyles.Left)      #ancho
104         self.Controls.Add(logo)
105         #logo.BorderStyle = BorderStyle.Fixed3D    #gives a border to the p
106
107 #############------------\------------#############
108
109         #combox drop down
110 (1)     cBox = ComboBox()   #dropdown control form
111         cBox.Location = Point(spacing,uiHeight/2)
112         cBox.Width = uiWidth -(spacing*4)
113 (2)     cBox.Items.AddRange(listInput)    # Adds an array of items to the l
114 (3)     cBox.DropDownStyle = ComboBoxStyle.DropDownList     #setting to dro
115 (4)     cBox.SelectedIndexChanged += self.dropDownOutput  #.Click+= registe
116         self.Controls.Add(cBox)
117
118
119 #############------------\-----------------------------------------------
120         #when a user selects a item in the drop down the dropDownOutput met
121 (5)     def dropDownOutput(self, sender, args):     #self is the instance of th
122             self.userOutput = sender.SelectedItem   #output the selected item.
123
124
125 ddForm = DropDownForm()
126
127 if run:     #if input is true run the application.
128     Application.Run(ddForm)
129
130 (6) results = ddForm.userOutput
131
132     OUT = results
```

6: Button Controls

- 1: Create a button control called btnOk
- 2: when the button is clicked register the event to run the def okButtonPressed
- 3: when the button is pressed close the form and set the runNextOutput variable btnOk = True, btnCancel = False
- 4: default values created act as output control. So if a user selects an item from the dropdown but does not press the next button the runNextOutput variable will return False.
- 5: User must select an item and click the btnOk to output the select item and True.

```
78
79      self.userOutput = userOutputDefaultStr #create a container to store the output from the
80      self.runNextOutput =  False  #set these default values
81
82
```

```
120     #Create ok button
121     btnOk = Button()    #create a button control
122     btnOk.Text = "Next"
123     btnOk.Location = Point(uiWidth - ((btnOk.Width * 2) + spacing ), uiHeight - (btnOk.Height
124     btnOk.Anchor = (AnchorStyles.Bottom | AnchorStyles.Right)
125     btnOk.Click += self.okButtonPressed    #Register the event on the button bress to trigger
126     self.Controls.Add(btnOk)
127
128     #Create Cancel Button
129     btnCancel = Button()
130     #btnCancel.Parent = self
131     btnCancel.Text = "Cancel"
132     btnCancel.Location = Point(uiWidth - (btnOk.Width + spacing), uiHeight - (btnOk.Height +
133     btnCancel.Anchor = (AnchorStyles.Bottom | AnchorStyles.Right)
134     btnCancel.Click += self.CnlButtonPressed
135     self.Controls.Add(btnCancel)
136
137
138 #############------------\-------------------------------------------------------------------
139     #when a user selects a item in the drop down the dropDownOutput method is called.
140 def dropDownOutput(self, sender, args):     #self is the instance of the GUI form. Sender is
141     self.userOutput = sender.SelectedItem   #output the selected item.
142
143 def okButtonPressed(self, sender, args):
144     self.Close()    #trigger to close the GUI when button is pressed
145     self.runNextOutput = True #if the ok button is pressed set runNextOutput as True
146
147 def CnlButtonPressed(self, sender, args):
148     self.Close()
149     self.runNextOutput = False #if the ok button is pressed set runNextOutput as False
150
151
152 ddForm = DropDownForm()
153
154 if run:     #if input is true run the application.
155     Application.Run(ddForm)
156
157     if ddForm.userOutput == userOutputDefaultStr:       #if the user does not select a item So th
158         results = ddForm.userOutput, ddForm.runNextOutput       #output the default string and ru
159     else:
160         results = ddForm.userOutput, ddForm.runNextOutput       #else if someone has selected a i
161
162     OUT = results
```

6: Finishing Touches

- 1: Create a linkLabel control: A button which activates a link
- 2: link is assigned to helpLink.Tag
- 3: when the link is pressed the event is logged to run the def openLink
- 4: Create a Panel. It's meant for grouping but can be used for colour
- 5: Sets its colour using the FromArgb method and set the border style to fixed 3D
- 6: If the link is a weblink you need webbrowser.Open()

  if it's a pdf etc you need System.Diagnostics.Process……

  Don't forget the additional import references!

```
import webbrowser   #needed to open a url
import System.IO    #needed to be able to open a pdf

#from System.Windows.Forms import *
from System.Windows.Forms import Application, Form, FormWindowState, Screen, Label, PictureBox, PictureBoxSizeM
from System.Windows.Forms import Button, LinkLabel, Panel
from System.Drawing import Icon, Color, Font, Point, Size
```

```
        #Create a weblink
1       helplink = LinkLabel()
        helplink.Text = "User Guide"
2       helplink.Tag = linkaddress      #tag is the web address
3       helplink.Click += self.openLink  #register click event with event handler
        helplink.Location = Point(uiWidth - ((btnOk.Width *3)+ spacing), uiHeight - (btnOk.Heigh
        self.Controls.Add(helplink)
        helplink.Anchor = (AnchorStyles.Bottom | AnchorStyles.Left)


4       colourPanel = Panel()
        colourPanel.Height = cBox.Height + spacing #Locate the panel behind the combo box.
        colourPanel.Width = uiWidth
        colourPanel.Location = Point(0,(uiHeight/3)-5)
5       colourPanel.BorderStyle = BorderStyle.Fixed3D # graphical style to pronounce the edge of
        colourPanel.BackColor = Color.FromArgb(1, 125, 199)  #set background colour by RGB value
        self.Controls.Add(colourPanel)


##############------------\-------------------------------------------------------------------
        #when a user selects a item in the drop down the dropDownOutput method is called.
    def dropDownOutput(self, sender, args):     #self is the instance of the GUI form. Sender is
        self.userOutput = sender.SelectedItem   #output the selected item.

    def okButtonPressed(self, sender, args):
        self.Close()    #trigger to close the GUI when button is pressed
        self.runNextOutput = True #if the ok button is pressed set runNextOutput as True

    def CnlButtonPressed(self, sender, args):
        self.Close()
        self.runNextOutput = False #if the ok button is pressed set runNextOutput as False

    def openLink(self, sender, event):
        webbrowser.open(sender.Tag)     #open a weblink
6       #System.Diagnostics.Process.Start(sender.Tag);  #to open a PDF
        self.Close()
        self.runNextOutput = False
```