A TrafficLights Widget

In this Supplement:

- ❑ Designing a TrafficLights Widget
- ❑ Coding a TrafficLights Widget
- ❑ Testing a TrafficLights Widget

Creating a TrafficLights Widget

Introduction

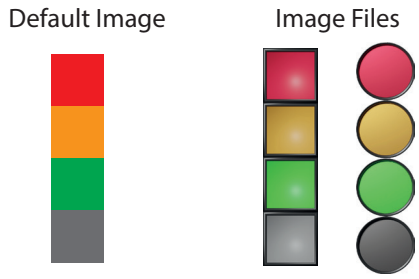
A *TrafficLights* widget cycles through the colours red, amber and green. In this implementation the lights change when clicked on or when a call to *SetGUITrafficLightsColor()* is made.

Unlike most of the other widgets in **Hands On AppGameKit Studio** Volume 2, this one is more likely to be used as a component of a game's main screen layout and so can be affected by zooming and view-offsets. If this is not desirable *FixGUITrafficLightsToScreen()* can be called which will make the widget immune to those effects.

When creating a *Trafficlights* widget we have the option to use a default 4-frame image or to use our own image file. In FIG-1.1 we can see the default image and two suitable image files (showing all four frames).

FIG-1.1

TrafficLight Images



The fourth frame (grey) is displayed when the widget is disabled.

Identifying the Required Operations

Like the *ColorPicker* widget, *TrafficLights* is a fairly simple control and requires very few operations. Those operations (given here in alphabetic order) are described briefly below:

CreateTrafficLights()
Creates a *TrafficLights* widget at a specified point, using a named image (or the default image) of a given size. The function returns the ID assigned to the widget.

DeleteGUITrafficLights()
Deletes the specified widget.

FixGUITrafficLightsToScreen()
Fixes the widget's screen size and position so that it is not affected by zoom and view-offset operations.

GetGUITrafficLightsEnabled()
Returns the widget's enabled/disabled state.

GetGUITrafficLightsExists()
Returns 1 if a *TrafficLights* widget of the specified ID exists, otherwise 0 is returned.

GetGUITrafficLightsState()
Returns a value representing the frame being displayed by an active widget (1 to 3). Zero is returned if the widget is inactive (when it displays frame 4).

HandleGUITrafficLights()
Detects user clicks on the sprite and in response displays the next colour in the sequence (returning to red after green).

SetGUITrafficLightsColor()
Sets the colour displayed by the widget.

SetGUITrafficLightsDepth()
Sets the layer on which the widget's sprite is drawn.

SetGUITrafficLightsEnabled()
Enables/disables the widget.

SetGUITrafficLightsPosition()
Moves the widget to a new position.

SetGUITrafficLightsSize()
Resizes the widget.

In addition, there is one helper function:

BuildDefaultGUITrafficLights()
This helper function creates a sprite using a default image if no image filename is specified in the call to *CreateGUITrafficLights()*.

Implementing the *TrafficLights* Widget

Data

We need to store the ID of the *TrafficLights*' sprite and the current state of the widget. This gives us the following data structure:

```
type GUITrafficLightsType
   spr      as integer  //Sprite ID
   state    as integer  //Widget state -1: deleted, 0:disabled,
                        ↳1:red, 2: amber, 3: green
endtype
```

Since a program may use several *TrafficLights* widgets, we'll need a global array to store the information:

```
global GUITrafficLights as GUITrafficLightsType[0]
```

Functions

Since *CreateGUITrafficLights()* calls *BuildDefaultGUITrafficLights()*, we'll start by defining that helper function.

As in the previous supplement, we'll produce a mini-spec for each function followed by its code and any points of interest.

BuildDefaultGUITrafficLights()

Mini-spec:

FUNCTION NAME	:	BuildDefaultGUITrafficLights
PARAMETERS		
In	:	None
Out	:	result : integer
PRE-CONDITION	:	None
DESCRIPTION	:	Creates a four-framed animated sprite. Each frame is a block of single colour. Frame 1: red, 2: amber, 3: green, 4: grey. *result* is set to the ID of sprite created.

Function code:

```
//*** Creates a trafficlights sprite using default images ***
//*** Returns ID of button sprite created                ***
function BuildDefaultGUITrafficLights(w as float)
   //***Create sprite ***
   tempspr = CreateSprite(0)
   //** Set size of sprite **
   h as float : h = w*GetDisplayAspect()
   SetSpriteSize(tempspr,w,h)
   //*** First frame (red) ***
   SetSpriteColor(tempspr,255,0,0,255)
   //***Grab first frame ***
   ClearScreen()
   DrawSprite(tempspr)
   img1 = GetImage(0,0,w,h)
   ClearScreen()
   //*** Second frame (amber) ***
   SetSpriteColor(tempspr,0xFF,0xBF,0,255)
   ClearScreen()
   DrawSprite(tempspr)
   img2 = GetImage(0,0,w,h)
   ClearScreen()
   //*** Third frame (green) ***
   SetSpriteColor(tempspr,0,255,0,255)
   ClearScreen()
   DrawSprite(tempspr)
   img3 = GetImage(0,0,w,h)
   ClearScreen()
   //*** Fourth frame (grey) ***
   SetSpriteColor(tempspr,100,100,100,255)
   ClearScreen()
   DrawSprite(tempspr)
   img4 = GetImage(0,0,w,h)
   result = CreateSprite(img1)
   SetSpriteAnimation(result,GetImageWidth(img1),
   ↳GetImageHeight(img1),1)
   AddSpriteAnimationFrame(result,img2)
   AddSpriteAnimationFrame(result,img3)
   AddSpriteAnimationFrame(result,img4)
   //*** Delete elements used ***
   DeleteSprite(tempspr)
endfunction result
```

CreateGUITrafficLights()

Mini-spec:

FUNCTION NAME	:	CreateGUITrafficLights
PARAMETERS		
In	:	x : real
		y : real
		w : real
		img : string
Out	:	id : integer
PRE-CONDITION	:	None
DESCRIPTION	:	Creates a sprite containing image *img* positioned at (*x,y*) and with a width of *w* (height is calculated automatically to create a square-shaped sprite). If *img* is an empty string, the default image is used. Sets the initially selected colour to red. *id* is set to the ID of the new widget.

Function code:

```
//*** Creates a four-frame traffic light (r,a,g,grey)
function CreateGUITrafficLights(x as float, y as float, w as float,
↳img as string)
   //*** Add cell to trafficlights list ***
   GUITrafficLights.length = GUITrafficLights.length+1
   //*** Last position in array is new trafficlight's ID ***
   id = GUITrafficLights.length
   //*** Create 4-frame sprite using default/specified image ***
   if img = ""
      GUITrafficLights[id].spr = BuildDefaultGUITrafficLights(w)
   else
      GUITrafficLights[id].spr = CreateSprite(LoadImage(img))
      //*** Convert to an animated sprite (4 frames) ***
      imgID = GetSpriteImageID(GUITrafficLights[id].spr)
      SetSpriteAnimation(GUITrafficLights[id].spr,
      ↳GetImageWidth(imgID),GetImageHeight(imgID)/4,4)
   endif
   //*** Show first frame ***
   SetSpriteFrame(GUITrafficLights[id].spr,1)
   //*** Size sprite ***
   SetSpriteSize(GUITrafficLights[id].spr,w,-1)
   //*** Position Sprite ***
   SetSpritePosition(GUITrafficLights[id].spr,x,y)
   //*** Draw on layer 5 ***
   SetSpriteDepth(GUITrafficLights[id].spr,5)
   //*** Trafficlights state = red showing ***
   GUITrafficLights[id].state = 1
endfunction id
```

Activity 1.1

Add the code given so far to *GUILibrary.agc* in the *Function Library* folder.

Create a test program, *TestGUITrafficLights*, that creates a *TrafficLights* widget. Test it using the default image and then the image file *TrafficLights.png* (as supplied).

GetGUITrafficLightsExists()

Mini-spec:

FUNCTION NAME	:	GetGUITrafficLightsExists
PARAMETERS		
In	:	id : as integer
Out	:	result : integer
PRE-CONDITION	:	None
DESCRIPTION	:	Sets *result* to 1 if a *TrafficLights* widget with ID *id* currently exists, otherwise *result* is set to zero.

Function code:

```
//*** Returns 1 if lights exist, else zero ***
function GetGUITrafficLightsExists(id as integer)
   if id < 1 OR id > GUITrafficLights.length
      result = 0
   elseif  GUITrafficLights[id].state = -1
      result = 0
   else
      result = 1
   endif
endfunction result
```

GetGUITrafficLightsEnabled()

Most operations can only be performed on an enabled widget (*state* > 0). To test if a *TrafficLights* widget is enabled, we can use this function.

Mini-spec:

FUNCTION NAME	:	GetGUITrafficLightsEnabled
PARAMETERS		
In	:	id : as integer
Out	:	result : integer
PRE-CONDITION	:	*id* is a valid widget ID
DESCRIPTION	:	Sets *result* to 1 if the *TrafficLights* widget specified by *id* is enabled. *result* is set to zero if the widget is disabled and *result* is set to -1 if the widget does not exist.

Function code:

```
//*** Returns 1 if enabled,zero: disabled, -1: fail ***
function GetGUITrafficLightsEnabled(id as integer)
   if NOT GetGUITrafficLightsExists(id)
      result = -1
   elseif  GUITrafficLights[id].state = 0
      result = 0
   else
      result = 1
   endif
endfunction result
```

HandleGUITrafficLights()

This is the function that does the most important part: interacting with the user to change the colour shown by the widget.

Mini-spec:

FUNCTION NAME	:	HandleGUITrafficLights
PARAMETERS		
In	:	id : integer
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an enabled *TrafficLights* widget.
DESCRIPTION	:	Clicking on the widget results in it progressing to the next colour in the sequence (red>amber>green). If green when clicked, the widget returns to red and the colour sequence cycle restarts. *result* is set to 1 if the operation is successful, otherwise *result* is set to zero.

Function code:

```
//*** Changes colour on each click ***
//*** Returns 1 if okay, else zero ***
function HandleGUITrafficLights(id as integer)
   if NOT GetGUITrafficLightsExists(id)
      exitfunction 0
   elseif NOT GetGUITrafficLightsEnabled(id)
      exitfunction 0
   endif
   result = 0
   hit = GetSpriteHit(ScreenToWorldX(GetPointerX()),
   ↳ScreenToWorldY(GetPointerY()))
   if GetPointerPressed() AND hit = GUITrafficLights[id].spr
      GUITrafficLights[id].state = Mod(GUITrafficLights[id].state,3)+1
      SetSpriteFrame(GUITrafficLights[id].spr,
      ↳GUITrafficLights[id].state)
      result = 1
   endif
endfunction result
```

Activity 1.2

Add the new routines to *GUILibrary.agc* and modify *TestGUITrafficLights* so that *GetUserInput()* calls *HandleGUITrafficLights()* and check that the colour changes each time the widget is clicked.

SetGUITrafficLightsColor()

Mini-spec:

FUNCTION NAME	:	SetGUITrafficLightsColor
PARAMETERS		
In	:	id : integer
		cf : integer
Out	:	result : integer

FUNCTION NAME	:	SetGUITrafficLightsColor (continued)
PRE-CONDITION	:	*id* specifies an enabled *TrafficLights* widget AND 1 <= *cf* <= 3
DESCRIPTION	:	Sets displayed frame to *cf* (hence the colour showing is set to red (*cf*=1), amber (*cf*=2) or green (*cf*=3). *result* is set to 1 if the operation succeeds, otherwise *result* is set to zero.

Function code:

```
//*** Sets the TrafficLights' colour(1:red/2:amber/3:green) ***
//*** Returns 1 if okay, otherwise zero                     ***
function SetGUITrafficLightsColor(id as integer, cf as integer)
   if NOT GetGUITrafficLightsExists(id) OR cf < 1 OR cf > 3
      exitfunction 0
   elseif GetGUITrafficLightsEnabled(id) <> 1
      exitfunction 0
   endif
   GUITrafficLights[id].state = cf
   SetSpriteFrame(GUITrafficLights[id].spr,cf)
endfunction 1
```

GetGUITrafficLightsState()

Mini-spec:

FUNCTION NAME	:	GetGUITrafficLightsState
PARAMETERS		
In	:	id : integer
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an existing widget
DESCRIPTION	:	Sets *result* to the current state of the *TrafficLights* widget (0: disabled, 1: red, 2: amber, 3: green). *result* is set to -1 if *id* is invalid.

Function code:

```
//*** Returns state of lights 1:red, 2:amber, 3:green ***
//*** 0: disabled, -1: does not exist                 ***
function GetGUITrafficLightsState(id as integer)
   if NOT GetGUITrafficLightsExists(id)
      result = -1
   else
      result = GUITrafficLights[id].state
   endif
endfunction result
```

Activity 1.3

Add the new routines to the library file and modify *TestGUITrafficLights* by adding a text resource which displays the state of the *TrafficLights* widget (the value displayed should be 1,2, or 3). Use the function *HandleUserInput()* to update the text.

SetGUITrafficLightsEnabled()

Mini-spec:

FUNCTION NAME	:	SetGUITrafficLightsEnabled
PARAMETERS		
In	:	id : integer
		fl : integer
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an existing widget AND 0 <= *fl* <= 1
DESCRIPTION	:	Sets the state of the widget to *fl*. If *fl* is 1 (enabled), then the *TrafficLights* will show red (frame 1). If *fl* is 0 (disabled), then the *TrafficLights* will show grey (frame 4). *result* is set to 1 if the operation is successful, and zero if the operation fails.

Function code:

```
//*** Enables/Disables lights           ***
//*** Returns 1 if okay, otherwise zero ***
function SetGUITrafficLightsEnabled(id as integer, fl as integer)
   if NOT GetGUITrafficLightsExists(id) OR fl < 0 OR fl > 1
      exitfunction 0
   endif
   SetSpriteFrame(GUITrafficLights[id].spr,Mod(fl+3,4)+1)
   GUITrafficLights[id].state = fl
endfunction 1
```

Activity 1.4

Add *SetGUITrafficLightsEnabled()* to the library file.

Modify *TestGUITrafficLights* so that the *TrafficLights* widget is disabled after five seconds. Check that this causes the fourth frame (grey) to be displayed and that the widget then becomes unresponsive.

Use *HandleOther()* to disable the widget after five seconds.

SetGUITrafficLightsPosition()

Mini-spec:

FUNCTION NAME	:	SetGUITrafficLightsPosition
PARAMETERS		
In	:	id : integer
		x : real
		y : real
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an existing widget
DESCRIPTION	:	Repositions the widget's top-left corner to location (x,y). *result* is set to 1 if the operation is successful, and zero if the operation fails.

Function code:

```
//*** Positions lights at (x,y). Returns 1 if okay ***
//*** else returns zero                             ***
function SetGUITrafficLightsPosition(id as integer, x as float,
↳y as float)
   if NOT GetGUITrafficLightsExists(id)
      exitfunction 0
   endif
   SetSpritePosition(GUITrafficLights[id].spr,x,y)
endfunction 1
```

SetGUITrafficLightsDepth()

Mini-spec:

FUNCTION NAME	:	SetGUITrafficLightsDepth
PARAMETERS		
In	:	id : integer
		ly : integer
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an existing widget AND 0<= *ly* <= 10000
DESCRIPTION	:	Moves the widget to depth *ly.*
		result is set to 1 if the operation is successful, and zero if the operation fails.

Function code:

```
//*** Sets lights' depth. Returns 1 if okay ***
//*** else returns zero                     ***
function SetGUITrafficLightsDepth(id as integer, ly as integer)
   if NOT GetGUITrafficLightsExists(id) OR ly < 0 OR ly > 10000
      exitfunction 0
   endif
   SetSpriteDepth(GUITrafficLights[id].spr,ly)
endfunction 1
```

SetGUITrafficLightsSize()

Mini-spec:

FUNCTION NAME	:	SetGUITrafficLightsSize
PARAMETERS		
In	:	id : integer
		sz : real
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an existing widget AND *sz* > 0
DESCRIPTION	:	Changes the size of the *TrafficLights* sprite to *sz* width (height is calculated automatically).
		result is set to 1 if the operation is successful, and zero if the operation fails.

Function code:

```
//*** Resizes lights. Returns 1 if okay ***
//*** else returns zero                 ***
function SetGUITrafficLightsSize(id as integer, sz as float)
   if NOT GetGUITrafficLightsExists(id)
      exitfunction 0
   endif
   SetSpriteSize(GUITrafficLights[id].spr,sz,-1)
endfunction 1
```

FixGUITrafficLightsToScreen()

Unlike all other widgets in the library, *TrafficLights* will react to zoom and view-offset commands since they are more likely to be embedded in the game play visuals. However, if we need to disable this option, we can call this function.

Mini-spec:

FUNCTION NAME	:	FixGUITrafficLightsToScreen
PARAMETERS		
In	:	id : integer
		fl : integer
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an existing widget AND 0<= *fl* <= 1
DESCRIPTION	:	Fixes/unfixes the screen position and dimensions of the widget so that it is not effected by subsequent zoom or view-offset commands. *result* is set to 1 if the operation is successful, and zero if the operation fails.

Function code:

```
//*** Fixes lights to screen ***
function FixGUITrafficLightsToScreen(id as integer, fl as integer)
   if NOT GetGUITrafficLightsExists(id) OR fl < 0 OR fl > 1
      exitfunction 0
   endif
   FixSpriteToScreen(GUITrafficLights[id].spr,fl)
endfunction 1
```

DeleteGUITrafficLights()

Mini-spec:

FUNCTION NAME	:	DeleteGUITrafficLights
PARAMETERS		
In	:	id : integer
Out	:	result : integer
PRE-CONDITION	:	*id* specifies an existing widget
DESCRIPTION	:	Marks the *TrafficLights* widget as deleted and deletes the sprite and image used by the widget. *result* is set to 1 if the operation is successful, and zero if the operation fails.

Function code:

```
//*** Deletes trafficlights. Returns 1 if okay ***
//*** else zero returned                        ***
function DeleteGUITrafficLights(id as integer)
   if NOT GetGUITrafficLightsExists(id)
      exitfunction 0
   endif
   //*** Record as deleted ***
   GUITrafficLights[id].state = -1
   //*** Delete visuals ***
   DeleteImage(GetSpriteImageID(GUITrafficLights[id].spr))
   DeleteSprite(GUITrafficLights[id].spr)
endfunction 1
```

Activity 1.5

Add the last of the routines to the library file and modify *TestGUITrafficLights* by adding the lines

```
SetViewOffset(5,-6)
SetViewZoom(1.5)
```

at the end of *CreateInitialLayout()*. Now test how the *TrafficLight* widget is affected when unfixed/fixed to the screen.

Solutions

NOTE: The complete code for the *TrafficLights* widget which is inserted into the *GUILibrary.agc* file is given at the end of the solutions.

NOTE 2: When using AGK Studio, the line

```
#renderer "Basic"
```

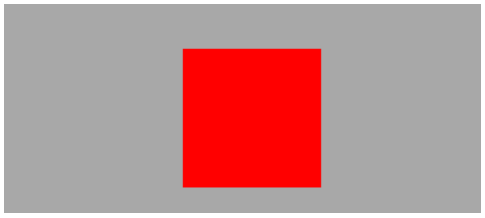
must be included in the main program.

Activity 1.1

Code for *TestGUITrafficLights*:

```
// Project: TestGUITrafficLights
// Created: 20-10-31

#renderer "Basic"

//*** Load required libraries ***
#include "../Function Library/CoreLibrary.agc"
#include "../Function Library/GUILibrary.agc"

type GameType
  tl as integer  //TrafficLights ID
endtype


global g as GameType

//**************************************
//***           Main program          ***
//**************************************
InitialiseScreen(1024,768,"TestGUITrafficLights",
↳0xA8A8A8,%1111)
CreateInitialLayout()
do
  Sync()
loop


//**************************************
//***        Main functions           ***
//**************************************
function CreateInitialLayout()
  g.tl = CreateGUITrafficLights(45,45,10,"")
endfunction
```

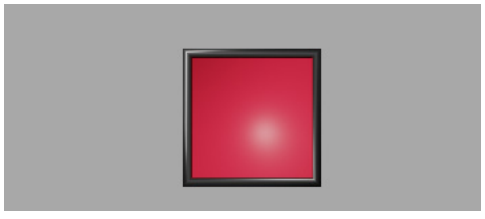
This version uses the default image, giving the output shown below.



By specifying a filename when creating the widget, such as in the line

```
g.tl = CreateGUITrafficLights(45,45,10,
↳"TrafficLights.png")
```

the output makes use of the image:



Activity 1.2

Modified code for *TestGUITrafficLights:*

```
// Project: TestGUITrafficLights
// Created: 20-10-31

#renderer "Basic"

//*** Load required libraries ***
#include "../Function Library/CoreLibrary.agc"
#include "../Function Library/GUILibrary.agc"

type GameType
  tl as integer  //TrafficLights ID
endtype


global g as GameType

//**************************************
//***           Main program          ***
//**************************************
InitialiseScreen(1024,768,"TestGUITrafficLights",
↳0xA8A8A8,%1111)
CreateInitialLayout()
do
  in = GetUserInput()
  Sync()
loop


//**************************************
//***        Main functions           ***
//**************************************
function CreateInitialLayout()
  g.tl = CreateGUITrafficLights(45,45,10,
  ↳"TrafficLights.png")
endfunction


function GetUserInput()
  result = HandleGUITrafficLights(g.tl)
endfunction result
```

The traffic lights should change when the mouse pointer is over the widget and the left mouse button is clicked.

Activity 1.3

Modified code for *TestGUITrafficLights:*

```
// Project: TestGUITrafficLights
// Created: 20-10-31

#renderer "Basic"

//*** Load required libraries ***
#include "../Function Library/CoreLibrary.agc"
#include "../Function Library/GUILibrary.agc"

type GameType
  tl   as integer  //TrafficLights ID
  lab  as integer  //Text label ID
endtype


global g as GameType

//**************************************
//***           Main program          ***
//**************************************
InitialiseScreen(1024,768,"TestGUITrafficLights",
↳0xA8A8A8,%1111)
CreateInitialLayout()
do
  in = GetUserInput()
  HandleUserInput(in)
  Sync()
loop


//**************************************
//***        Main functions           ***
//**************************************
function CreateInitialLayout()
  g.tl = CreateGUITrafficLights(45,45,10,
```

```
  ↳"TrafficLights.png")
  g.lab = CreateText("1")
  SetTextPosition(g.lab,45,40)
endfunction


function GetUserInput()
  result = HandleGUITrafficLights(g.tl)
endfunction result


function HandleUserInput(in as integer)
  if in = 1
    SetTextString(g.lab,
    ↳Str(GetGUITrafficLightsState(g.tl)))
  endif
endfunction
```

Activity 1.4

Modified code for *TestGUITrafficLights:*

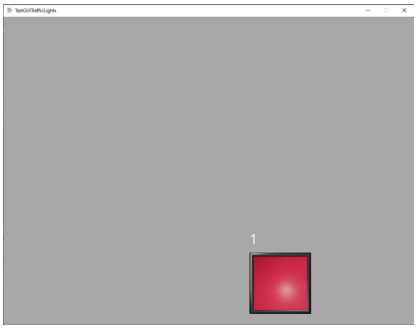
```
// Project: TestGUITrafficLights
// Created: 20-10-31

#renderer "Basic"

//*** Load required libraries ***
#include "../Function Library/CoreLibrary.agc"
#include "../Function Library/GUILibrary.agc"

type GameType
  tl   as integer  //TrafficLights ID
  lab  as integer  //Text label ID
endtype


global g as GameType

//**************************************
//***           Main program          ***
//**************************************
InitialiseScreen(1024,768,"TestGUITrafficLights",
↳0xA8A8A8,%1111)
CreateInitialLayout()
do
  in = GetUserInput()
  HandleUserInput(in)
  HandleOther()
  Sync()
loop


//**************************************
//***        Main functions           ***
//**************************************
function CreateInitialLayout()
  g.tl = CreateGUITrafficLights(45,45,10,
  ↳"TrafficLights.png")
  g.lab = CreateText("1")
  SetTextPosition(g.lab,45,40)
endfunction


function GetUserInput()
  result = HandleGUITrafficLights(g.tl)
endfunction result


function HandleUserInput(in as integer)
  if in = 1
    SetTextString(g.lab,
    ↳Str(GetGUITrafficLightsState(g.tl)))
  endif
endfunction


function HandleOther()
  global disabled
  if NOT disabled AND GetSeconds()>= 5
    SetGUITrafficLightsEnabled(g.tl,0)
    SetTextString(g.lab,
    ↳Str(GetGUITrafficLightsState(g.tl)))
    disabled = 1
  endif
endfunction
```

Activity 1.5

Only the function *CreateInitialLayout()* needs to be changed:

```
function CreateInitialLayout()
  g.tl = CreateGUITrafficLights(45,45,10,
  ↳"TrafficLights.png")
  g.lab = CreateText("1")
  SetTextPosition(g.lab,45,40)
  SetViewOffset(5,-6)
  SetViewZoom(1.5)
endfunction
```

With offset and zoom effects, the display changes to the layout shown below:

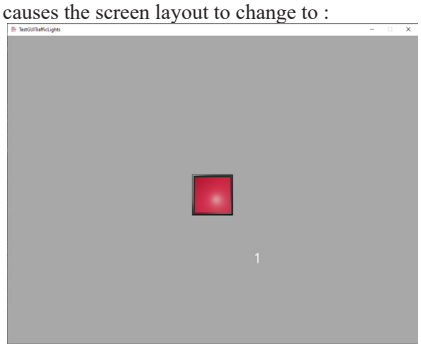


Both the label and *TrafficLights* widget have moved position on the screen and become larger.

To fix the widget, *CreateInitialLayout()* is changed to :

```
function CreateInitialLayout()
  g.tl = CreateGUITrafficLights(45,45,10,
  "TrafficLights.png")
  FixGUITrafficLightsToScreen(g.tl,1)
  g.lab = CreateText("1")
  SetTextPosition(g.lab,45,40)
  SetViewOffset(5,-6)
  SetViewZoom(1.5)
endfunction
```

This causes the screen layout to change to :



The widget is no longer affected by the offset and zoom commands. On the other hand, the text resource continues to be displaced and enlarged since it is not fixed.

Code for *TrafficLights* widget (added to *GUILibrary.agc* in *Functions Library* folder):

```
//**************************************
//*** TrafficLights Data & Functions ***
//**************************************
type TrafficLightsType
  spr    as integer  //Traffic light (4 frame)
  state  as integer  //-1:del, 0:off, 1:red, 2:amber,
                     ↳3:green
endtype

global GUITrafficLights as TrafficLightsType[0]
```

```
//*** Creates a four-frame traffic light (r,a,g,grey)
function CreateGUITrafficLights(x as float, y as float,
↳w as float, img as string)
  //*** Add cell to TrafficLights list ***
  GUITrafficLights.length = GUITrafficLights.length+1
  //*** Last position in array is new trafficlight's
  ↳ID ***
  id = GUITrafficLights.length
  //*** Create 4-frame sprite using default/specified
  ↳image ***
  if img = ""
    GUITrafficLights[id].spr =
    ↳BuildDefaultGUITrafficLights(w)
  else
    GUITrafficLights[id].spr =
    ↳CreateSprite(LoadImage(img))
    //*** Convert to an animated sprite (4 frames)
    ↳***
    imgID = GetSpriteImageID(GUITrafficLights[
    ↳id].spr)
    SetSpriteAnimation(GUITrafficLights[id].spr,
    ↳GetImageWidth(imgID),GetImageHeight(imgID)/4,
    ↳4)
  endif
  //*** Show first frame ***
  SetSpriteFrame(GUITrafficLights[id].spr,1)
  //*** Size sprite ***
  SetSpriteSize(GUITrafficLights[id].spr,w,-1)
  //*** Position Sprite ***
  SetSpritePosition(GUITrafficLights[id].spr,x,y)
  //*** Draw on layer 5 ***
  SetSpriteDepth(GUITrafficLights[id].spr,5)
  //*** Trafficlights state = red showing ***
  GUITrafficLights[id].state = 1
endfunction id


//*** Deletes TrafficLights. Returns 1 if okay ***
//*** else zero returned                        ***
function DeleteGUITrafficLights(id as integer)
  if NOT GetGUITrafficLightsExists(id)
    exitfunction 0
  endif
  //*** Record as deleted ***
  GUITrafficLights[id].state = -1
  //*** Delete visuals ***
  DeleteImage(GetSpriteImageID(GUITrafficLights[
  ↳id].spr))
  DeleteSprite(GUITrafficLights[id].spr)
endfunction 1


//*** Fixes lights to screen ***
function FixGUITrafficLightsToScreen(id as integer,
↳fl as integer)
  if NOT GetGUITrafficLightsExists(id) OR fl<0 OR fl>1
    exitfunction 0
  endif
  FixSpriteToScreen(GUITrafficLights[id].spr,fl)
endfunction 1


//*** Returns 1 if enabled,zero:disabled,-1:fail ***
function GetGUITrafficLightsEnabled(id as integer)
  if NOT GetGUITrafficLightsExists(id)
    result = -1
  elseif  GUITrafficLights[id].state = 0
    result = 0
  else
    result = 1
  endif
endfunction result


//*** Returns 1 if lights exist, else zero ***
function GetGUITrafficLightsExists(id as integer)
  if id < 1 OR id > GUITrafficLights.length
    result = 0
  elseif  GUITrafficLights[id].state = -1
    result = 0
  else
    result = 1
  endif
endfunction result


//*** Returns state of lights 1:red, 2:amber,
↳3:green ***
//*** 0: disabled, -1: does not exist ***
function GetGUITrafficLightsState(id as integer)
  if NOT GetGUITrafficLightsExists(id)
    result = -1
  else
    result = GUITrafficLights[id].state
  endif
endfunction result


//*** Changes colour on each click ***
//*** Returns 1 if okay, else zero ***
function HandleGUITrafficLights(id as integer)
  if NOT GetGUITrafficLightsExists(id)
    exitfunction 0
  elseif NOT GetGUITrafficLightsEnabled(id)
    exitfunction 0
  endif
  result = 0
  hit = GetSpriteHit(ScreenToWorldX(GetPointerX()),
  ↳ScreenToWorldY(GetPointerY()))
  if GetPointerPressed() AND hit = GUITrafficLights[
  ↳id].spr
    GUITrafficLights[id].state = Mod(GUITrafficLights[
    ↳id].state,3)+1
    SetSpriteFrame(GUITrafficLights[id].spr,
    ↳GUITrafficLights[id].state)
    result = 1
  endif
endfunction result


//*** Sets the TrafficLights' colour(1:red/ 2:amber/
↳3:green) ***
//*** Returns 1 if okay, otherwise zero        ***
function SetGUITrafficLightsColor(id as integer,
↳cf as integer)
  if NOT GetGUITrafficLightsExists(id) OR cf<1 OR cf>3
    exitfunction 0
  elseif GetGUITrafficLightsEnabled(id) <> 1
    exitfunction 0
  endif
  GUITrafficLights[id].state = cf
  SetSpriteFrame(GUITrafficLights[id].spr,cf)
endfunction 1


//*** Sets lights' depth. Returns 1 if okay ***
//*** else returns zero                     ***
function SetGUITrafficLightsDepth(id as integer,
↳ly as integer)
  if NOT GetGUITrafficLightsExists(id) OR ly<0 OR
  ↳ly>10000
    exitfunction 0
  endif
  SetSpriteDepth(GUITrafficLights[id].spr,ly)
endfunction 1


//*** Enables/Disables lights           ***
//*** Returns 1 if okay, otherwise zero ***
function SetGUITrafficLightsEnabled(id as integer,
↳fl as integer)
  if NOT GetGUITrafficLightsExists(id) OR fl<0 OR fl>1
    exitfunction 0
  endif
  SetSpriteFrame(GUITrafficLights[id].spr,
  ↳Mod(fl+3,4)+1)
  GUITrafficLights[id].state = fl
endfunction 1


//*** Positions lights at (x,y). Returns 1 if ok ***
//*** else returns zero                          ***
function SetGUITrafficLightsPosition(id as integer,
↳x as float, y as float)
  if NOT GetGUITrafficLightsExists(id)
    exitfunction 0
  endif
  SetSpritePosition(GUITrafficLights[id].spr,x,y)
endfunction 1


//*** Resizes lights. Returns 1 if okay ***
//*** else returns zero                 ***
function SetGUITrafficLightsSize(id as integer,
↳sz as float)
  if NOT GetGUITrafficLightsExists(id)
    exitfunction 0
  endif
```

```
  SetSpriteSize(GUITrafficLights[id].spr,sz,-1)
endfunction 1



//*************************************
//*** GUIBTrafficLights helper function ***
//*************************************

//*** Creates a TrafficLights sprite using default
↳images ***
//*** Returns ID of button sprite created      ***
function BuildDefaultGUITrafficLights(w as float)
  //***Create sprite ***
  tempspr = CreateSprite(0)
  //** Set size of sprite **
  h as float : h = w*GetDisplayAspect()
  SetSpriteSize(tempspr,w,h)
  //*** First frame (red) ***
  SetSpriteColor(tempspr,255,0,0,255)
  //***Grab first frame ***
  ClearScreen()
  DrawSprite(tempspr)
  img1 = GetImage(0,0,w,h)
  ClearScreen()
  //*** Second frame (amber) ***
  SetSpriteColor(tempspr,0xFF,0xBF,0,255)
  ClearScreen()
  DrawSprite(tempspr)
  img2 = GetImage(0,0,w,h)
  ClearScreen()
  //*** Third frame (green) ***
  SetSpriteColor(tempspr,0,255,0,255)
  ClearScreen()
  DrawSprite(tempspr)
  img3 = GetImage(0,0,w,h)
  ClearScreen()
  //*** Fourth frame (grey) ***
  SetSpriteColor(tempspr,100,100,100,255)
  ClearScreen()
  DrawSprite(tempspr)
  img4 = GetImage(0,0,w,h)
  result = CreateSprite(img1)
  SetSpriteAnimation(result,GetImageWidth(img1),
  ↳GetImageHeight(img1),1)
  AddSpriteAnimationFrame(result,img2)
  AddSpriteAnimationFrame(result,img3)
  AddSpriteAnimationFrame(result,img4)
  //*** Delete elements used ***
  DeleteSprite(tempspr)
endfunction result
```