# A ColorPicker Widget

## In this Supplement:

- ❑ Designing a ColorPicker Widget
- ❑ Coding a ColorPicker Widget
- ❑ Testing a ColorPicker Widget

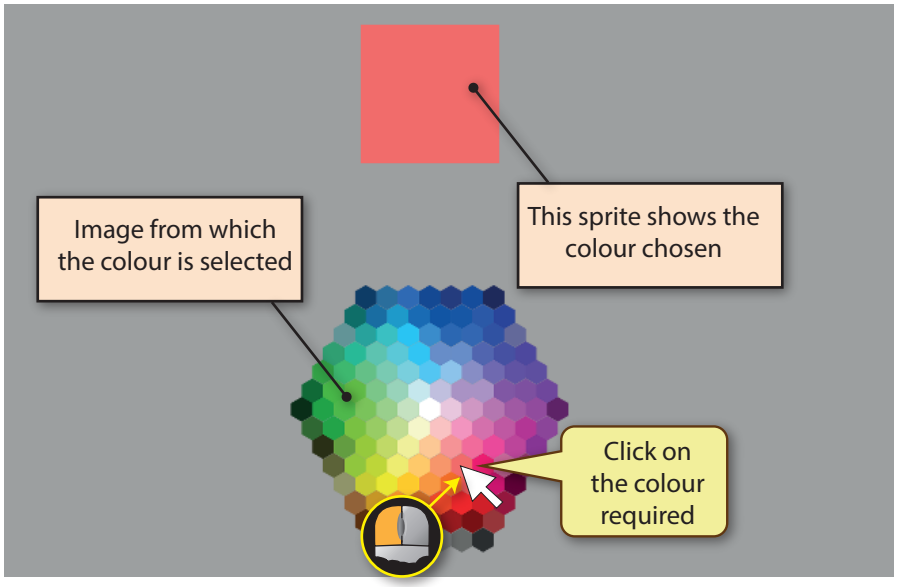
# Creating a Colour Picker Widget

## Introduction

In the first supplement to **Hands On AppGameKit Studio Volume 2** we created a Stopwatch widget. This supplement adds another control to that collection by creating a **ColorPicker** widget. A screen shot of the widget in use is shown in FIG-1.1.

**FIG-1.1**

ColorPicker Screenshot




## Identifying the Required Operations

The colour picker is such a simple widget, that it requires very few operations.

**CreateColorPicker()**
Creates the ColorPicker widget at a specified point, using a named image (or default image) of a given size.

**HandleGUIColorPicker()**
Detects user clicks on the sprite containing the colour image and returns the colour selected as a single integer value. The value returned is in the same format as that used by AGK's `MakeColor()` function. The opacity of any colour is assumed to be 255 (opaque).

**GetGUIColorPickerRed()**
Returns the red component of the last colour selected.

**GetGUIColorPickerGreen()**
Returns the green component of the last colour selected.

**GetGUIColorPickerBlue()**
Returns the blue component of the last colour selected.

**SetGUIColorPickerDepth()**
Sets the layer on which the ColorPicker's sprite is drawn.

**GetGUIColorPickerDepth()**
Returns the layer on which the ColorPicker's sprite is currently drawn.

**DeleteGUIColorPicker()**
Marks the ColorPicker widget as deleted as well as deleting its image and sprite.

**GetGUIColorPickerExists()**
Returns 1 if the ColorPicker object currently exists, else zero is returned.

**BuildDefaultGUIColorPicker()**
This helper function creates a default image to be used by the ColorPicker if no image is specified in the call to *CreateColorPicker()*.

We could add other functions such as *SetGUIColorPickerPosition()* and *SetGUIColorPickerSize()* but these values are unlikely to need adjusting after the ColorPicker has been created.

# Implementing the ColorPicker

## Data

We need to store the ID of the ColorPicker's sprite. Holding the red, green, and blue values of the last colour selected would also be useful as is a deleted indicator. This gives us the following data structure:

```
type GUIColorPickerType
   spr      as integer  //Sprite ID
   red      as integer  //Red of last selected
   green    as integer  //Green of last selected
   blue     as integer  //Blue of last selected
   exists   as integer  //(0:no, 1:yes)
endtype
```

Since we'll assume there will only be a single ColorPicker showing at any one time, we'll only need a single global variable to store the information:

```
global GUIColorPicker as GUIColorPickerType
```

## Functions

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

As in the previous supplement, we'll start by producing a mini-spec for each function and then create the code before going on to test each routine that we build.

Our testing will be informal since exhaustive testing would obscure the purpose of this supplement.

## BuildDefaultGUIColorPicker()

Mini-spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | BuildDefaultGUIColorPicker |
| PARAMETERS | | |
| In | : | None |
| Out | : | result |
| PRE-CONDITION | : | None |
| DESCRIPTION | : | Creates a rectangle using `DrawBox()`. The corner colours from top-left in a clockwise direction: red, green, black, blue. The box is then captured as an image and *result* is set to the ID assigned to that image. |

Code:

When colour parameters for *DrawBox()* are given as numbers, hex digit pairs are in the order alpha,blue,green,red.

```
//*** Creates a default image for ColorPicker ***
//*** Used when no image supplied             ***
function BuildDefaultGUIColorPicker()
   ClearScreen()
   DrawBox(0,0,20,20,0xFF0000FF,0xFF00FF00,0xFFFF0000,0xFF000000,1)
   Render()
   result = GetImage(0,0,20,20)
   ClearScreen()
endfunction result
```

## CreateGUIColorPicker()

Mini-spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | CreateGUIColorPicker |
| PARAMETERS | | |
| In | : | x : real |
| | | y : real |
| | | w : real |
| | | h : real |
| | | img : string |
| Out | : | result |
| PRE-CONDITION | : | None |
| DESCRIPTION | : | Creates a sprite containing image *img* positioned at (*x*,*y*) and sized as *w* by *h*. If *img* is an empty string, the default image is used. Sets the initially selected colour to black. |

Code:

```
//*** Creates a sprite for the ColorPicker ***
function CreateGUIColorPicker(x as float, y as float, w as float,
↳h as float, img as string)
   //*** Create sprite ***
   if img = ""
      sprID = CreateSprite(BuildDefaultGUIColorPicker())
```

```
   else
      sprID = CreateSprite(LoadImage(img))
   endif
   SetSpriteSize(sprID,w,h)
   SetSpritePosition(sprID,x,y)
   //*** Store sprite ID ***
   GUIColorPicker.spr = sprID
   //*** Set selected colour to black ***
   GUIColorPicker.red = 0
   GUIColorPicker.green = 0
   GUIColorPicker.blue = 0
   //*** Mark as exists ***
   GUIColorPicker.exists = 1
endfunction
```

**Activity 1.1**

Create a new project called *TestColorPicker*. Create a standard main program and add the code for *ColorPickerType*, *ColorPicker* global variable, *CreateGUIColorPicker()* and *BuildDefaultGUIColorPicker()*.

Also add a blank sprite (image ID = 0) at the top-centre of the screen which will eventually be used to show the selected colour.

Check that the code compiles and that the blank sprite appears on the screen.

## GetGUIColorPickerExists()

We are only safe to call ColorPicker functions if we know that a ColorPicker has already been created, so the next function to code is *GetGUIColorPickerExists()*:

| | | |
|---|---|---|
| FUNCTION NAME | : | GetGUIColorPickerExists |
| PARAMETERS | | |
| In | : | None |
| Out | : | result |
| PRE-CONDITION | : | None |
| DESCRIPTION | : | Sets *result* to 1 if a ColorPicker currently exists, otherwise *result* is set to zero. |

Code:

```
//*** Returns 1 if a ColorPicker exists ***
function GetColorPickerExists()
   result = GUIColorPicker.exists
endfunction result
```

## HandleGUIColorPicker()

This is the function that does the most import part: returning the numeric value of the colour selected by the user.

| | | |
|---|---|---|
| FUNCTION NAME | : | HandleGUIColorPicker |
| PARAMETERS | | |
| In | : | None |
| Out | : | result |
| PRE-CONDITION | : | ColorPicker exists |
| DESCRIPTION | : | Sets *result* to an integer value representing the colour clicked on by the user. If the ColorPicker does not exist or no selection, *result* is set to -1. |

Code:

```
//*** Returns colour under mouse when clicked ***
function HandleGUIColorPicker()
   result = -1
   if NOT GetGUIColorPickerExists()
      exitfunction result
   endif
   //*** If clicked on colour picker ...
   if GetPointerPressed() and GetSpriteHit(GetPointerX(),
   ↳GetPointerY()) = GUIColorPicker.spr
      Render()
      //*** Capture this part of screen as an image ***
      imgID = GetImage(GetPointerX(), GetPointerY(),0.1,0.1)
      //*** Convert to a memblock ***
      memID = CreateMemblockFromImage(imgID)
      //*** Get colour of first pixel's data ***
      GUIColorPicker.red = GetMemblockByte(memID,12)
      GUIColorPicker.green = GetMemblockByte(memID,13)
      GUIColorPicker.blue = GetMemblockByte(memID,14)
      //*** Create a single number from colours ***
      result = 0xFF000000+(GUIColorPicker.blue<<16) +
      (GUIColorPicker.green<<8) + GUIColorPicker.red
      //*** Delete the memblock and image***
      DeleteMemblock(memID)
      DeleteImage(imgID)
   endif
endfunction result
```

Note that when a valid colour value is returned it is given in the format used by AGK's *Draw* commands such as `DrawBox()` and `DrawEllipse()`. That is, the bytes that make up the value are in the order: alpha, blue, green, red.

## GetGUIColorPickerRed()

Mini-Spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | GetGUIColorPickerRed |
| PARAMETERS | | |
| In | : | None |
| Out | : | result |
| PRE-CONDITION | : | ColorPicker exists |
| DESCRIPTION | : | Sets *result* to the value of the red component in the last colour to be picked. If the ColorPicker does not exist, *result* is set to zero. |

Code:

```
//*** Returns the red value of last selection ***
function GetGUIColorPickerRed()
   if NOT GetGUIColorPickerExists()
      exitfunction 0
   endif
   result = GUIColorPicker.red
endfunction result
```

## GetGUIColorPickerGreen()

Mini-Spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | GetGUIColorPickerGreen |
| PARAMETERS | | |
| In | : | None |
| Out | : | result |
| PRE-CONDITION | : | ColorPicker exists |
| DESCRIPTION | : | Sets *result* to the value of the green component in the last colour to be picked. If the ColorPicker does not exist, *result* is set to zero. |

Code:

```
//*** Returns the green value of last selection ***
function GetGUIColorPickerGreen()
   if NOT GetGUIColorPickerExists()
      exitfunction 0
   endif
   result = GUIColorPicker.green
endfunction result
```

## GetGUIColorPickerBlue()

Mini-Spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | GetGUIColorPickerBlue |
| PARAMETERS | | |
| In | : | None |
| Out | : | result |
| PRE-CONDITION | : | ColorPicker exists |
| DESCRIPTION | : | Sets *result* to the value of the blue component in the last colour to be picked. If the ColorPicker does not exist, *result* is set to zero. |

Code:

```
//*** Returns the blue value of last selection ***
function GetGUIColorPickerBlue()
   if NOT GetGUIColorPickerExists()
      exitfunction 0
   endif
   result = GUIColorPicker.blue
endfunction result
```

**Activity 1.2**

Modify *TestColorPicker* by adding the code for *GetGUIColorPickerExists()*, *HandleGUIColorPicker()*, *GetGUIColorPickerRed()*, *GetGUIColorPickerGreen()*, and *GetGUIColorPickerBlue()*.

Add a *GetUserInput()* function which calls *HandleGUIColorPicker()* and a *HandleUserInput()* which assigns the colour selected to the top-centre sprite.

Copy the image *colours.png* into the project's *media* folder.

In *CreateInitialLayout()* add a call to *CreateGUIColorPicker()* using the file *colours.png* as the image from which the colour is selected.

Does the top sprite change colour correctly when a colour is selected?

What happens when an empty string is used in the call to *CreateGUIColorPicker()*?

Using the current version of AGK Studio (20-03-27) the program terminates when no file is specified in the call to *CreateGUIColorPicker()*. Use the OpenGL renderer to correct this problem.

## SetGUIColorPickerDepth()

Mini-spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | SetGUIColorPickerDepth |
| PARAMETERS | | |
| In | : | d : integer |
| Out | : | None |
| PRE-CONDITION | : | ColorPicker exists AND 0 <= *d* <= 10000 |
| DESCRIPTION | : | Sets the layer on which the ColorPicker sprite is drawn to *d*. |

Code:

```
//*** Sets layer of ColorPicker sprite ***
function SetGUIColorPickerDepth(d as integer)
   if NOT GetGUIColorPickerExists() OR d < 0 OR d > 10000
      exitfunction
   endif
   //*** Set sprite depth ***
   SetSpriteDepth(GUIColorPicker.spr,layer)
endfunction
```

## GetGUIColorPickerDepth()

Mini-spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | GetGUIColorPickerDepth |
| PARAMETERS | | |
| In | : | None |
| Out | : | result |
| PRE-CONDITION | : | ColorPicker exists |
| DESCRIPTION | : | Sets *result* to the value of the current layer used. If ColorPicker does not exist, *result* is set to -1. |

Code:

```
//*** Returns layer of ColorPicker sprite ***
function SetGUIColorPickerDepth(layer as integer)
   if NOT GetGUIColorPickerExists()
      exitfunction -1
   endif
   result = GetSpriteDepth(GUIColorPicker.spr)
endfunction result
```

## DeleteGUIColorPicker()

Mini-spec:

| | | |
|---|---|---|
| FUNCTION NAME | : | DeleteGUIColorPicker |
| PARAMETERS | | |
| In | : | None |
| Out | : | None |
| PRE-CONDITION | : | None |
| DESCRIPTION | : | Marks the ColorPicker as deleted (sets *exists* to zero) and deletes the sprite and image used by the ColorPicker. |

Code:

```
//*** Deletes ColorPicker widget ***
function DeleteGUIColorPicker()
   if NOT GetGUIColorPickerExists()
      exitfunction
   endif
   DeleteImage(GetSpriteImageID(GUIColorPicker.spr))
   DeleteSprite(GUIColorPicker.spr)
   GUIColorPicker.exists = 0
endfunction
```

**Activity 1.3**

Modify *TestColorPicker* by adding the code for *SetGUIColorPickerDepth(), GetGUIColorPickerDepth()* and *DeleteGUIColorPicker().*

Add the code for the ColorPicker widget to *GUILibrary.agc* with suitable documentation for each function.

# Solutions

### Activity 1.1

Code for *TestColorPicker*:

```
// Project: TestColourPicker
// Created: 20-05-25

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

type GameType
  spr    as integer  //ID of sprite showing selected
                     ↳colour
endtype


global g as GameType

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

//**************************************
//***        Main functions           ***
//**************************************
function CreateInitialLayout()
  g.spr = CreateSprite(0)
  SetSpritePosition(g.spr, 40, 10)
  SetSpriteSize(g.spr, 10, -1)
endfunction



//**************************************
//***     ColorPicker functions       ***
//**************************************
type GUIColorPickerType
  spr    as integer  //Sprite ID
  red    as integer  //Red of last selected
  green  as integer  //Green of last selected
  blue   as integer  //Blue of last selected
  exists as integer  //(0:no, 1:yes)
endtype

global GUIColorPicker as GUIColorPickerType

//*** Creates a sprite for the ColorPicker ***
function CreateGUIColorPicker(x as float, y as float,
↳w as float, h as float, img as string)
  //*** Create sprite ***
  if img = ""
    sprID = CreateSprite(
    ↳BuildDefaultGUIColorPicker())
  else
    sprID = CreateSprite(LoadImage(img))
  endif
  SetSpriteSize(sprID,w,h)
  SetSpritePosition(sprID,x,y)
  //*** Store sprite ID ***
  GUIColorPicker.spr = sprID
  //*** Set selected colour to black ***
  GUIColorPicker.red = 0
  GUIColorPicker.green = 0
  GUIColorPicker.blue = 0
  //*** Mark as exists ***
  GUIColorPicker.exists = 1
endfunction

//*** Creates a default image for ColorPicker ***
//*** Used when no image supplied             ***
function BuildDefaultGUIColorPicker()
  ClearScreen()
  DrawBox(0,0,20,20,0xFF0000FF, 0xFF00FF00,
  ↳0xFFFF0000,0xFF000000,1)
  Render()
  result = GetImage(0,0,20,20)
  ClearScreen()
endfunction result
```

### Activity 1.2

Modified code for *TestColorPicker*:

```
// Project: TestColourPicker
// Created: 20-05-25

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

type GameType
  spr    as integer  //ID of sprite showing selected
colour
endtype


global g as GameType

//**************************************
//***           Main program          ***
//**************************************
SetErrorMode(2)
InitialiseScreen(0,0,1024,768,"TestColourPicker",
0xA8A8A8,%1111)
CreateInitialLayout()
do
  in = GetUserInput()
  HandleUserInput(in)
  Sync()
loop

//**************************************
//***        Main functions           ***
//**************************************

function CreateInitialLayout()
  g.spr = CreateSprite(0)
  SetSpritePosition(g.spr, 40, 10)
  SetSpriteSize(g.spr, 10, -1)
  CreateGUIColorPicker(35,30,20,-1,"")
endfunction

function GetUserInput()
  result = HandleGUIColorPicker()
endfunction result

function HandleUserInput(in as integer)
  if GetGUIColorPickerExists()
    SetSpriteColor(g.spr,GetGUIColorPickerRed(),
    ↳GetGUIColorPickerGreen(),
    ↳GetGUIColorPickerBlue(), 255)
  endif
endfunction

//**************************************
//***     ColorPicker functions       ***
//**************************************
type GUIColorPickerType
  spr    as integer  //Sprite ID
  red    as integer  //Red of last selected
  green  as integer  //Green of last selected
  blue   as integer  //Blue of last selected
  exists as integer  //(0:no, 1:yes)
endtype

global GUIColorPicker as GUIColorPickerType

//*** Creates a sprite for the ColorPicker ***
function CreateGUIColorPicker(x as float, y as float,
↳w as float, h as float, img as string)
  //*** Create sprite ***
  if img = ""
    sprID =
    ↳CreateSprite(BuildDefaultGUIColorPicker())
  else
    sprID = CreateSprite(LoadImage(img))
  endif
  SetSpriteSize(sprID,w,h)
  SetSpritePosition(sprID,x,y)
  //*** Store sprite ID ***
  GUIColorPicker.spr = sprID
  //*** Set selected colour to black ***
  GUIColorPicker.red = 0
  GUIColorPicker.green = 0
  GUIColorPicker.blue = 0
  //*** Mark as exists ***
  GUIColorPicker.exists = 1
endfunction
```

```
//*** Creates a default image for ColorPicker ***
//*** Used when no image supplied             ***
function BuildDefaultGUIColorPicker()
  ClearScreen()
  DrawBox(0,0,20,20,0xFF0000FF,0xFF00FF00,
  ↳0xFFFF0000, 0xFF000000,1)
  Render()
  result = GetImage(0,0,20,20)
  ClearScreen()
endfunction result


//*** Returns 1 if a ColorPicker exists ***
function GetGUIColorPickerExists()
  result = GUIColorPicker.exists
endfunction result


//*** Returns colour under mouse when clicked ***
function HandleGUIColorPicker()
  result = -1
  if NOT GetGUIColorPickerExists()
    exitfunction result
  endif
  //*** If clicked on colour picker ...
  if GetPointerPressed() and
  ↳GetSpriteHit(GetPointerX(), GetPointerY()) =
  ↳GUIColorPicker.spr
    Render()
    //*** Capture this part of screen as image ***
    imgID = GetImage(GetPointerX(), GetPointerY(),
    ↳0.1,0.1)
    //*** Convert to a memblock ***
    memID = CreateMemblockFromImage(imgID)
    //*** Get colour of first pixel's data ***
    GUIColorPicker.red = GetMemblockByte(memID,12)
    GUIColorPicker.green = GetMemblockByte(memID,13)
    GUIColorPicker.blue = GetMemblockByte(memID,14)
    //*** Create a single number from colours ***
    result = 0xFF000000+(GUIColorPicker.blue<<16) +
    ↳(GUIColorPicker.green<<8) + GUIColorPicker.red
    //*** Delete the memblock and image ***
    DeleteMemblock(memID)
    DeleteImage(imgID)
  endif
endfunction result

//*** Returns the red value of last selection ***
function GetGUIColorPickerRed()
  if NOT GetGUIColorPickerExists()
    exitfunction 0
  endif
  result = GUIColorPicker.red
endfunction result

//*** Returns the green value of last selection ***
function GetGUIColorPickerGreen()
  if NOT GetGUIColorPickerExists()
    exitfunction 0
  endif
  result = GUIColorPicker.green
endfunction result

//*** Returns the blue value of last selection ***
function GetGUIColorPickerBlue()
  if NOT GetGUIColorPickerExists()
    exitfunction 0
  endif
  result = GUIColorPicker.blue
endfunction result
```

**Activity 1.3**

Final code for the *ColorPicker* widget (*BuildDefaultGUIColorPicker()* has moved to a "helper functions" section ):

```
//**************************************
//***     GUIDialogBox Description    ***
//**************************************
//CreateGUIColorPicker(x#,y#,w#,h#,f$)
//   Shows image f$ at (x#,y) sized
//   as w# by h#. If f$ = "" uses a
//   default DrawBox() image

//GetGUIColorPickerExists()
//   Returns 1 if a ColorPicker exists

//HandleGUIColorPicker()
//   Allows user to select a colour
//   the displayed spriteand returns
//   the colour value in the form
//   alpha-blue-green-red

//GetGUIColorPickerRed()
//   Returns the red value of last
//   selected colour

//GetGUIColorPickerGreen()
//   Returns the green value of last
//   selected colour

//GetGUIColorPickerBlue()
//   Returns the blue value of last
//   selected colour

//SetGUIColorPickerDepth(d)
//   Sets ColorPicker's sprite to
//   layer d

//GetGUIColorPickerDepth()
//   Returns the ColorPicker's sprite's
//   depth

//DeleteGUIColorPicker()
//   Marks ColorPicker as deleted and
//   deletes its image and sprite

//BuildDefaultGUIColorPicker()
//   Creates a box and converts it to an
//   image and returns the image's ID
//   Called by CreateGUIColorPicker()

//**************************************
//***     ColorPicker functions       ***
//**************************************
type GUIColorPickerType
  spr    as integer  //Sprite ID
  red    as integer  //Red of last selected
  green  as integer  //Green of last selected
  blue   as integer  //Blue of last selected
  exists as integer  //(0:no, 1:yes)
endtype

global GUIColorPicker as GUIColorPickerType

//*** Creates a sprite for the ColorPicker ***
function CreateGUIColorPicker(x as float, y as float,
↳w as float, h as float, img as string)
  //*** Create sprite ***
  if img = ""
    sprID =
    ↳CreateSprite(BuildDefaultGUIColorPicker())
  else
    sprID = CreateSprite(LoadImage(img))
  endif
  SetSpriteSize(sprID,w,h)
  SetSpritePosition(sprID,x,y)
  //*** Store sprite ID ***
  GUIColorPicker.spr = sprID
  //*** Set selected colour to black ***
  GUIColorPicker.red = 0
  GUIColorPicker.green = 0
  GUIColorPicker.blue = 0
  //*** Mark as exists ***
  GUIColorPicker.exists = 1
endfunction

//*** Returns 1 if a ColorPicker exists ***
function GetGUIColorPickerExists()
  result = GUIColorPicker.exists
endfunction result


//*** Returns colour under mouse when clicked ***
function HandleGUIColorPicker()
  result = -1
  if NOT GetGUIColorPickerExists()
    exitfunction result
  endif
  //*** If clicked on colour picker ...
  if GetPointerPressed() and
  ↳GetSpriteHit(GetPointerX(), GetPointerY()) =
  ↳GUIColorPicker.spr
    Render()
    //*** Capture this part of screen as image ***
    imgID = GetImage(GetPointerX(), GetPointerY(),
    ↳0.1,0.1)
    //*** Convert to a memblock ***
```

```
    memID = CreateMemblockFromImage(imgID)
    //*** Get colour of first pixel's data ***
    GUIColorPicker.red = GetMemblockByte(memID,12)
    GUIColorPicker.green = GetMemblockByte(memID,13)
    GUIColorPicker.blue = GetMemblockByte(memID,14)
    //*** Create a single number from colours ***
    result = 0xFF000000+(GUIColorPicker.blue<<16) +
    ↳(GUIColorPicker.green<<8) + GUIColorPicker.red
    //*** Delete the memblock and image ***
    DeleteMemblock(memID)
    DeleteImage(imgID)
  endif
endfunction result

//*** Returns the red value of last selection ***
function GetGUIColorPickerRed()
  if NOT GetGUIColorPickerExists()
    exitfunction 0
  endif
  result = GUIColorPicker.red
endfunction result

//*** Returns the green value of last selection ***
function GetGUIColorPickerGreen()
  if NOT GetGUIColorPickerExists()
    exitfunction 0
  endif
  result = GUIColorPicker.green
endfunction result

//*** Returns the blue value of last selection ***
function GetGUIColorPickerBlue()
  if NOT GetGUIColorPickerExists()
    exitfunction 0
  endif
  result = GUIColorPicker.blue
endfunction result

//*** Sets layer of ColorPicker sprite ***
function SetGUIColorPickerDepth(layer as integer)
  if NOT GetGUIColorPickerExists() OR layer < 0 OR
  ↳layer > 10000
    exitfunction
  endif
  //*** Set sprite depth ***
  SetSpriteDepth(GUIColorPicker.spr,layer)
endfunction

//*** Returns layer of ColorPicker sprite ***
function GetGUIColorPickerDepth(layer as integer)
  if NOT GetGUIColorPickerExists()
    exitfunction -1
  endif
  result = GetSpriteDepth(GUIColorPicker.spr)
endfunction result


//*** Deletes ColorPicker widget ***
function DeleteGUIColorPicker()
  if NOT GetGUIColorPickerExists()
    exitfunction
  endif
  DeleteImage(GetSpriteImageID(GUIColorPicker.spr))
  DeleteSprite(GUIColorPicker.spr)
  GUIColorPicker.exists = 0
endfunction



//************************************
//***      Helper functions          ***
//************************************

//*** Creates a default image for ColorPicker ***
//*** Used when no  image supplied            ***
function BuildDefaultGUIColorPicker()
  ClearScreen()
  DrawBox(0,0,20,20,0xFF0000FF, 0xFF00FF00,
  ↳0xFFFF0000,0xFF000000,1)
  Render()
  result = GetImage(0,0,20,20)
  ClearScreen()
endfunction result
```