

### 23.1.5 Arbeit mit Farben

In diesem Kapitel erfahren Sie exemplarisch, wie Sie in Gambas die Klassen `Color` (`gb.image`), `Color` (`gb.qt4`) und `ImageStat` (`gb.image`) einsetzen, um

- Farbwerte zu ermitteln (auszulesen),
- Farben zuzuweisen,
- Farben zu mischen,
- Farben zu analysieren, indem Sie deren Rot-Grün-Blau-Farbanteile bestimmen,
- Farben zu analysieren, indem Sie deren Alpha-Wert bestimmen,
- Farbinformationen für ein Pixel eines Bildes zu bestimmen,
- die Farbe für ein Pixel eines Bildes zu setzen,
- Farben aus unterschiedlichen Farbräumen zu konvertieren (RGB ↔ HSV),
- Farb-Konstanten einzusetzen,
- Farbpaletten zu generieren oder
- die Klasse `ImageStat` (`gb.image`) zu nutzen, um die Farbtiefe eines Bildes zu bestimmen.

Die Reihenfolge der Anwendungen widerspiegelt keine Rangfolge. Farbe und Farbwert werden synonym verwendet.

#### 23.1.5.1 Farbwerte ermitteln

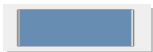
Folgende Klassen verfügen über die Farb-Eigenschaften `Background` und `Foreground`, die Sie auslesen oder setzen können:

```
Button, ButtonBox, CheckBox, ColorButton, ColorChooser, ColumnView, ComboBox, Container, Control, DateBox,
DateChooser, Dial, DirChooser, DirView, DrawingArea, Editor, Embedder, Expander, FMain, FileChooser, File-
Properties, FileView, FontChooser, Form, Frame, GridView, HBox, HPanel, HSplit, IconPanel, IconView, Image-
View, LCDLabel, LCDNumber, Label, ListBox, ListContainer, ListView, MaskBox, MenuButton, MovieBox, Panel,
PictureBox, ProgressBar, RadioButton, ScrollArea, ScrollBar, ScrollView, Separator, SidePanel, Slider, Sli-
derBox, SpinBox, TabPanel, TabStrip, TableView, TextArea, TextBox, TextEdit, TextLabel, ToggleButton, Tool-
Button, ToolPanel, TrayIcon, TreeView, UserContainer, UserControl, VBox, VPanel, VSplit, ValueBox, Window,
Wizard, _IconPanelContainer, _Split, _TabPanelContainer, _TreeView, _WizardContainer.
```

```
iColor = ColorButton.Color
iColor = PictureBox1.Picture.Image[Mouse.X, Mouse.Y]
iColorHex = Hex(Color.Background, 6)
```

#### 23.1.5.2 Farben zuweisen

Farben aus den beiden Farbräumen RGB und HSV können Sie unmittelbar zuweisen oder als sichere Eingaben über `ColorButton` sowie `ColorChooser` oder den direkten Aufruf des Farbauswahl-Dialogs `Dialog.SelectColor()`. Genauso zuverlässig ist die Verwendung von Farb-Konstanten. Die Farbwerte (Datentyp `Integer`) können Sie als dezimale, hexadezimale oder binäre Zahl schreiben. Der hexadezimalen Zahl ist der Präfix `&` oder `&H` oder `&h` voranzustellen.



```
Textarea.Background = Color.SelectedBackground
Textarea.Background = &H86ABD9
Textarea.Background = 8825817
Textarea.Background = Color.RGB(134, 171, 217)' » Rot: 86→ 134, Grün: AB→ 171, Blau: D9→ 217
Textarea.Background = Color.HSV(213, 97, 217)

iBGColor = Color.RGB(134, 171, 217) ' Farbwert generieren
Textarea.Background = iBGColor ' Farbwert zuweisen
Textarea.Foreground = Color.Red
```

#### 23.1.5.3 Farben analysieren

Bei Farbanalyse interessieren sowohl der Farbwert als auch die Rot-Grün-Blau-Farbanteile und optional auch der Alpha-Wert. Sie können die Klassen `Color` (`gb.image`) und `ColorInfo` (`gb.image`) nutzen, um die aufgeführten Farbinformation im interessierenden Farbraum (RGBA und HSV) zu ermitteln. Setzen Sie die Klasse `ColorInfo` ein, dann ist es ein guter Plan, mit einem Objekt vom Typ `ColorInfo` zu arbeiten. Es wird davon ausgegangen, dass die zu analysierende Farbe in der Variablen `iColor` in der Analyse-Prozedur bereitgestellt wird (→ `ColorChooser`) und die Farbanteile in `iRed`, `iGreen`, `IBlue`, `IHue`, `iSaturation`, `iValue` sowie `iAlpha`.

```

Dim iColor As Integer
Dim hColorInfo As ColorInfo
Dim iRed, iGreen, iBlue, iHue, iSaturation, iValue, iAlpha As Integer

iColor = ColorChooser1.SelectedColor

' Variante 1 im RGB-Farbraum hilft immer ...
iRed = CInt(iColor / 256 / 256) Mod 256
iGreen = CInt(iColor / 256) Mod 256
iBlue = iColor Mod 256

' Variante 2 - Sie nutzen den []-Operator der Color-Klasse
iRed = Color[iColor].Red ' Rot-Farbanteil auslesen
iGreen = Color[iColor].Green ' Green-Farbanteil auslesen
iBlue = Color[iColor].Blue ' Blau-Farbanteil auslesen
iAlpha = Color[iColor].Alpha ' Alpha-Wert auslesen
iHue = Color[iColor].Hue
iSaturation = Color[iColor].Saturation
fValue = Color[iColor].Value

' Variante 3 - Klasse ColorInfo
hColorInfo = Color[iColor] ' Ein ColorInfo-Objekt wird zurückgegeben
iRed = hColorInfo.Red
iGreen = hColorInfo.Green
iBlue = hColorInfo.Blue
iAlpha = hColorInfo.Alpha
iHue = hColorInfo.Hue
iSaturation = hColorInfo.Saturation
fValue = hColorInfo.Value

' Variante 3b

' Möchten Sie keine Variable für hColorInfo anlegen, ist auch die Verwendung von WITH denkbar
With Color[iColor]
    iRed = .Red
    iRed += 20
' Auch Zuweisungen funktionieren hier wie erwartet
    .Red = iRed

' Am Ende kann die veränderte Farbe zugewiesen werden
ColorChooser1.SelectedColor = .Color
End ' With

```

#### 23.1.5.4 Farbpaletten generieren

Eine Farb-Palette ist eine Auswahl bestimmter Farben aus dem verfügbaren Farbraum. Zwei Farbpaletten und deren Generatoren werden vorgestellt: Palette mit Grautönen und eine Palette websicherer Farben.

Die Palette der Grautöne generieren Sie mit diesem Quelltext, bei der die RGB-Farbanteile stets den *gleichen* Wert haben:

```

Public Sub btnSetGrayPalette_Click()
    Dim panStartX, panX, panY, iRow, iColumn, iCount As Integer
    Dim pPanel As Panel

    panStartX = 424
    panX = panStartX
    panY = 16
    iCount = 0

    For iRow = 1 To 16
        For iColumn = 1 To 16
            pPanel = New Panel(FMain)
            pPanel.X = panX
            pPanel.Y = panY
            pPanel.H = 12
            pPanel.W = 12
            pPanel.Background = Color.RGB(iCount, iCount, iCount)
            panX += 16
            Inc iCount
        Next ' iColumn

        panX = panStartX
        panY += 16

    Next ' iRow

End ' btnSetGrayPalette_Click()

```

Hier das Ergebnis:

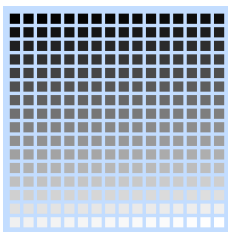


Abbildung 23.1.5.4.1: Farbpalette Grautöne

Hier der Quelltext für die websicheren Farben ...

```
Public Sub btnSetWebFarbPalette_Click()
    Dim R, G, B, panStartX, panStartY, panX, panY, iCount As Integer
    Dim pPanel As Panel

    panX = 16
    panStartY = 16
    panY = panStartY

    For R = 0 To 255 Step 51
        For G = 0 To 255 Step 51
            For B = 0 To 255 Step 51
                pPanel = New Panel(FMain)
                pPanel.X = panX
                pPanel.Y = panY
                pPanel.H = 12
                pPanel.W = 12
                pPanel.Background = Color.RGB(R, G, B)
                panY += 16
            Next ' Blue
            panY = panStartY
            panX += 16
        Next ' Green
        panY = panStartY
        panX += 16
    Next ' Red
End ' btnSetWebFarbPalette_Click()
```

... und so sehen die 216 (6x6x6) Farben aus:

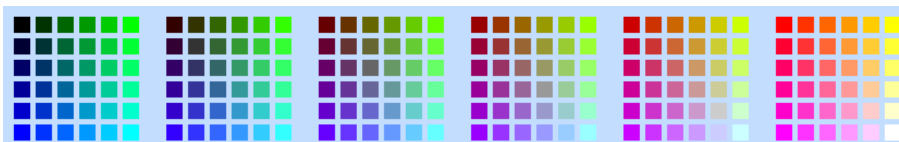


Abbildung 23.1.5.4.2: Web-Farb-Palette

### 23.1.5.5 Bestimmung der Farbtiefe eines Bildes

Nach <http://www.e-teaching.org/> wird 'Farbtiefe' so festgelegt: *"Die Farbtiefe gibt an, wie viele unterschiedliche Farbstufen für jeden einzelnen Bildpunkt einer Grafik zur Verfügung stehen. Da die "Feinheit" der Abstufungen davon abhängt, wie viel Speicherplatz pro Bildpunkt verwendet wird, gibt man die Farbtiefe in Bits an. Mit 8 Bit lassen sich z. B. 256 Farbtöne für einen Farbkanal unterscheiden. Eine Farbe entsteht dabei durch Mischung mehrerer Farbkanäle eines Farbraumes. Bei Computergrafiken wird dabei üblicherweise der RGB-Farbraum verwendet, in dem sich Farben durch additive Mischung der drei Grundfarben Rot, Grün und Blau zusammensetzen."*

```
Public Sub btnGetDepth_Click()
    Dim hImageStat As ImageStat

    hImageStat = ImageStat("Images/color.png") ' Bildpfad
    Print "Farbtiefe = " & hImageStat.Depth & " Bits"
    Print "Farbtiefe = " & ImageStat("Images/color.png").Depth & " Bits" ' Klasse ImageStat (gb.image)
    ' Wechsel der Klasse!
    Print "Farbtiefe = " & PictureBox1.Picture.Image.Depth & " Bits" ' Klasse Image (gb.image)
End ' btnGetDepth_Click()
```

### 23.1.5.6 Farben mischen

Im Kapitel 23.1.1 *Klassen Color* finden Sie weitere Hinweise für das Mischen von Farben sowie deren Veränderung:

```
Public Sub btnMerge_Click()
    Dim fWeight As Float

    fWeight = 0.3
    ColorButton3.Color = Color.Merge(ColorButton1.Color, ColorButton2.Color, fWeight)
End ' btnMerge_Click()
```

### 23.1.5.7 Farb-Manipulationen für ein Bild-Pixel

Mit diesem Quelltext lesen Sie Farbinformationen für ein Bild-Pixel aus und setzen eine neue Farbe für das selektierte Pixel. Die neue Farbe können Sie in einem Farbauswahl-Dialog (ColorButton) festlegen.

```
' Gambas class file

Public Sub Form_Open()

    FMain.Center
    FMain.Resizable = False
    ColorButton1.Color = Color.White

End ' Form_Open

Public Sub GetSetPixelColor(x As Integer, y As Integer)
    Dim iColor, hColor As Integer
    Dim hImage As Image

    ' Print "Mouse.X = " & Mouse.X
    ' Print "Mouse.Y " & Mouse.Y

    ' GetPixelColor
    iColor = PictureBox1.Picture.Image[x, y]

    ' Print "Aktuelle Pixel-Farbe = " & iColor
    ' Print "Rot-Anteil = " & Color[iColor].Red
    ' Print "Grün-Anteil = " & Color[iColor].Green
    ' Print "Blau-Anteil = " & Color[iColor].Blue

    ' SetPixelColor
    ' Konvertieren nach Image, da nur die Image-Klasse den Zugriff auf Pixel gestattet.
    hImage = PictureBox1.Picture.Image

    ' Pixel-Farbe setzen -> Block-Pixel aus 9 Pixeln
    hColor = ColorButton1.Color
    If x > 1 And y > 1
        For i = -1 To 1
            For k = -1 To 1
                hImage[x + i, y + k] = hColor
            Next ' k
        Next ' i
    Endif ' x > 1 And y > 1 ?

    ' Verändertes Image wieder in ein Picture umwandeln und in die PictureBox setzen
    PictureBox1.Picture = hImage.Picture

End ' GetSetPixelColor(x As Integer, y As Integer)

Public Sub PictureBox1_MouseDown()
    GetSetPixelColor(Mouse.X, Mouse.Y)
End ' PictureBox1_MouseDown()

Public Sub PictureBox1_MouseMove()
    If Mouse.Left = True Then
        GetSetPixelColor(Mouse.X, Mouse.Y)
    Endif ' Mouse.Left = True ?
End ' PictureBox1_MouseMove()

Public Sub btnClose_Click()
    FMain.Close
End ' btnClose_Click()
```

- Sie können mit einem Mausklick ein Pixel – genauer ein Blockpixel aus 9 einzelnen Pixeln – umfärben (grüne Punkte), wobei das Block-Pixel nur den Effekt hervorheben soll.

- Mit gedrückter linker Maustaste können Sie auf dem Bild mit einem 3Pixelx3Pixel-Pinsel malen (gelbe Linie).
- Sie können sich die (absoluten) Maus-Koordinaten (bezogen auf die PictureBox) und die Farb-informationen des Originalpixels in der Konsole ausgeben lassen.

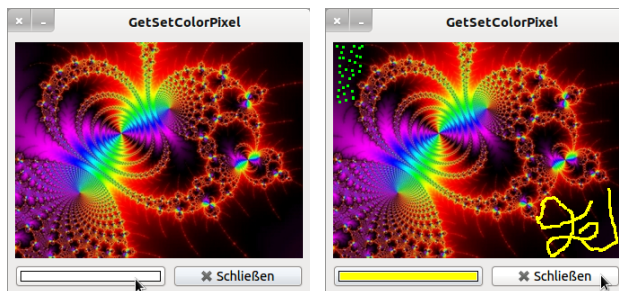


Abbildung 23.1.5.7.1: Originalbild und manipuliertes Bild