3.3 Gambas-Programme mit Parametern

Den Start von Programmen mit Parametern in der Konsole kennen Sie sicher. So startet der folgende Befehl das Editor-Programm 'gedit', öffnet die Datei 'set_color.sh' im Home-Verzeichnis und setzt dann den Cursor in der Zeile 5 an die 9. Position:

```
hans@linux:~$ gedit ./set_color.sh +5:9
```

3.3.1 Exkurs – Programme mit Parametern

Die Funktion des vorgestellten Programms (Bash-Skript) 'set_color.sh' besteht darin, in Abhängigkeit von der Anzahl der übergebenen Parameter entweder den Parametern $1 bis $3 über den Befehl 'let' 3 Werte intern zuzuweisen und diese danach den Variablen R, G und B oder die Variablen R, G und B mit den Werten der genau 3 übergebenen Parameter zu belegen. Anschließend wird mit den Variablen R, G und B ein (Farb-)Wert berechnet und angezeigt.

```
#!/bin/bash
#
if [ $# -ne 3 ]      # Wenn die Anzahl der Parameter ungleich 3 ist, dann ...
  then
    set 220 20 180   # → $1=220 , $2=20 , $3=180
    R=$1
    G=$2
    B=$3
  else
    R=$1
    G=$2
    B=$3
fi
let color=$R*256*256+$G*256+$B
echo Farbwert = $color
echo Weiter mit ENTER...
read dummy
```

Allgemein gilt:

- $1 bis $k bilden eine Parameterliste der übergebenen Parameter, in der die Reihenfolge auch einer Rangfolge entspricht.
- Das erste Argument *$0* enthält den Namen des gestarteten Programms.
- Der Wert von *$#* gibt die Anzahl der *übergebenen* Parameter $1 … $k zurück.

Der Start des Skripts 'set_color.sh' erfolgt zuerst mit genau drei Parametern und dann nur mit 2 Parametern. Die Parameter werden nicht auf Validität im Kontext mit RGB-Farbwerten geprüft:

```
hans@linux:~$ chmod +x set_color.sh # Das Skript ausführbar machen

hans@linux:~$ ./set_color.sh 10 20 128
Farbwert = 660608
Weiter mit ENTER...

hans@linux:~$ $HOME/set_color.sh 0 128
Farbwert = 14423220
Weiter mit ENTER...
```

3.3.2 Gambas-Programme mit Parametern

Selbstverständlich können Sie auch ein Gambas-Programm mit Parametern in einer Konsole starten:

```
hans@linux:~$ gbx3 $HOME/color_select -- 225 110 60
hans@linux:~$ gbx3 ./color_select -- 225 110 60
hans@linux:~$ gbr3 $HOME/color_select/color_select.gambas -- 225 110 60
```

Als Trennsymbol zwischen dem Gambas-Projekt-Pfad und den Argumenten wird "--" benutzt. Das wird in der Hilfe von *gbx3* auch angedeutet: *Usage: gbx3 [options] [<project file>] [-- <arguments>]*, wenn man <arguments> als Platzhalter für eine durch Leerzeichen getrennte Parameter-Liste interpretiert.

Gambas stellt mit *Application.Args* ein Array mit den *übergebenen* Argumenten des Programms aus der Konsole zur Verfügung. Das erste Argument Application.Args[0] (≡ $0) ist immer der Name des gestarteten Gambas-Programms und die Eigenschaft *Application.Args.Count* gibt die Anzahl aller – nicht nur der übergebenen – Parameter zurück. In diesem Punkt unterscheidet sich die Arbeit mit dem Array der Argumente in Gambas von der Ermittlung der Anzahl der Argumente in einem Bash-Skript!

Der Quelltext des Gambas-Projekts 'color_select' ist analog zu dem o.a. Bash-Skript aufgebaut. Mit den Werten der 3 Parameter – die in dieser Variante nicht geprüft werden – wird ein Farbwert berechnet, der als Vorgabewert für die eingesetzte Komponente *ColorChooser1* dient:

```
Public Sub Form_Open()

  Dim R, G, B As Integer

  FColor.Center
  FColor.Resizable = False

  If Application.Args.Count <> 4 Then
     R = 220
     G = 20
     B = 180
  Else
     R = Val(Application.Args[1])
     G = Val(Application.Args[2])
     B = Val(Application.Args[3])
  Endif '  Application.Args.Count <> 4 ?

  ColorChooser1.Value = Color.RGB(R, G, B)

End ' Form_Open
```

Der Aufruf

```
hans@linux:~$ gbx3 ./color_select -- 33 133 33
```

startet das Programm zur Farbauswahl mit der Vorgabe-Farbe mittel-grün:

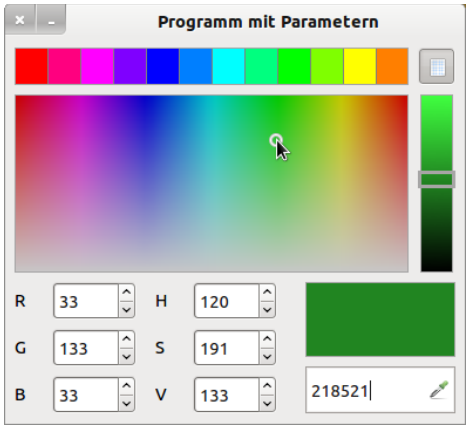

Abbildung 3.3.2.1: Programm zur Farbauswahl – Vorgabe-Farbe über 3 Parameter

Der folgende modifizierte Quelltext-Ausschnitt enthält eine Prüfung der Parameter-Werte:

```
Public Sub Form_Open()
  Dim sMessage As String

  FColor.Center
  FColor.Resizable = False

  If Application.Args.Count <> 4 Then
     ColorChooser1.Value = Color.RGB(220, 20, 180) ' Farbe: pink
  Else
     If ParameterTest(Application.Args[1], Application.Args[2], Application.Args[3]) = True Then
        ColorChooser1.Value = Color.RGB(Val(Application.Args[1]), Val(Application.Args[2]),
        Val(Application.Args[3]))
     Else
        sMessage = "Mindestens ein Parameter hat einen falschen Wert!"
```

```
        sMessage &= gb.NewLine
        sMessage &= "Parameterliste: " & Application.Args[1] & Chr(32) & Application.Args[2] & Chr(32) &
                    Application.Args[3]
        sMessage &= gb.NewLine
        sMessage &= "Das Programm wird sofort beendet!"
        Message.Error(sMessage)
        FColor.Close
     Endif
  Endif ' Application.Args.Count = 4 ?

End ' Form_Open

Public Function ParameterTest(p_1 As String, p_2 As String, p_3 As String) As Boolean
  Dim bOK As Boolean

  If IsInteger(p_1) = True And IsInteger(p_2) = True And IsInteger(p_3) = True Then
     If (Val(p_1) >= 0) And (Val(p_1) <= 255) Then
        bOK = True
     Else
        bOK = False
     Endif
     If (Val(p_2) >= 0) And (Val(p_2) <= 255) Then
        bOK = True
     Else
        bOK = False
     Endif
     If (Val(p_3) >= 0) And (Val(p_3) <= 255) Then
        bOK = True
     Else
        bOK = False
     Endif
  Else
     bOK = False
  Endif

  Return bOK

End ' Function ParameterTest(...) As Boolean
```

Damit folgt einem Aufruf des Programms in der Konsole:

```
hans@linux:~$ gbx3 ./color_select -- 128 128 294
```

diese Fehlermeldung:

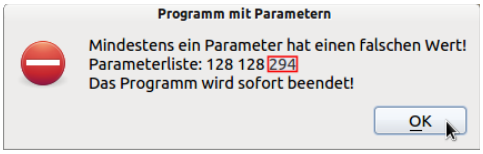

Abbildung 3.3.2.2: Fehlermeldung – 294 ∉ [0...255]