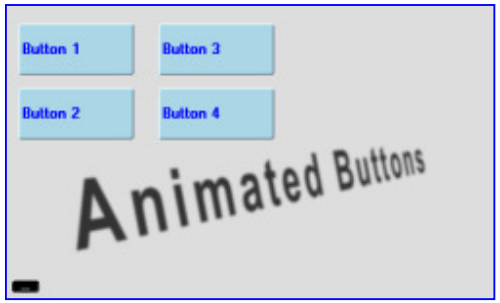
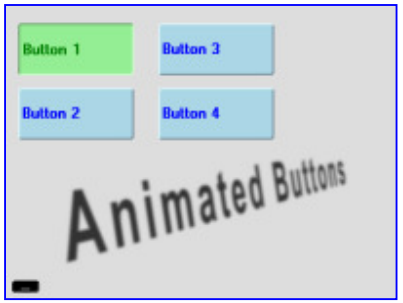
Inhalt:

Erstellt: 10. Januar 2011
Letztmals geändert am : 12. Januar 2011
Autor: Christoph Angst, S-TEC electronics AG



ScreenShot der Applikation BTTN
auf dem eigerPanel70C (WVGA)



ScreenShot der Applikation BTTN
mit “eingedrücktem” Button
auf dem eigerPanel57H/C (VGA)

Wissenswertes zu animierten Buttons

„Animierte Buttons“ auf einem Touchscreen verändern Ihr Aussehen, wenn sie berührt werden. Dadurch erkennt die bedienende Person, dass das System die Berührung erkannt hat und damit der Befehl entgegengenommen worden ist.

Das zentrale Stichwort im Zusammenhang mit animierten Buttons auf dem Touchscreen heisst: „HotSpot“. Damit bezeichnen wir einen „berührungsempfindlich“ Bereich auf dem Bildschirm. Ein HotSpot wird in eigerScript mit Position (X- und Y-Koordinaten), Breite und Höhe definiert.

Wir unterscheiden vier Berührungsarten:

- **Down** : Der Finger trifft auf dem Touchscreen innerhalb des HotSpots auf, d.h. der Button wird „eingedrückt“.
- **Up** : Der Finger hebt vom HotSpot ab, d.h. der Button wird „losgelassen“.
- **Leave** : Der Finger bleibt auf dem Touchscreen und hält den Druck aufrecht, während er aus dem HotSpot hinausgeschoben wird, d.h. den Button verlässt.
- **Enter** : Der Finger liegt auf dem Touchscreen ausserhalb des HotSpots und wird nun in den HotSpot hinein verschoben, d.h. auf den Button geschoben.

Programmieren eines animierten Buttons in vier Schritten

Zu Beginn gleich eine typische Button-Subroutine in eigerScript, mit welcher ein Button gezeichnet wird.

eigerScript-Code 1: Subroutine, die ein Button mit der Aufschrift „Button 1“ zeichnet.

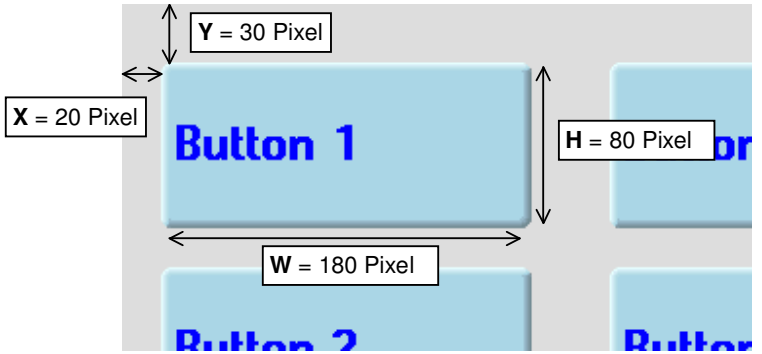
```
SUB    B01_Leave
       Load.Geometry_XYWH(Button_X,Button_Y,Button_W,Button_H)     ; Position, Breite und Höhe von Button und HotSpot
       Fill.LabelParameter(Button_UP_Style)                        ; zeigt unberührten Button
       Label.Text('Button 1')                                  ; HotSpot wird gezeichnet mit der Aufschrift „Button 1“
       HotSpot.Install(NIL,Subroutine_1,Subroutine_2,Subroutine_3) ; Subroutinen-Aufruf bei Leave-, Down- und Up-Ereignis
ENDSUB
```

In der Button-Subroutine, wie sie in eigerScript-Code 1 gezeigt ist, werden zunächst die Geometrie und weitere Eigenschaften des Buttons geladen, bevor er dann mit dem Befehl `Label.Text('Button 1')` gezeichnet wird. Diese Vorbereitungsschritte werden nachfolgend der Reihe nach erklärt. In der Regel wird mit den gleichen Geometrie-Werten auch gleich der HotSpot installiert. Das kann entweder in der gleichen Subroutine oder unmittelbar anschliessend erfolgen, solange die Geometrie-Register noch nicht verändert worden sind.

1. **Geometrie des Buttons und des berührungsempfindlichen Bereichs festlegen**

   Die Geometrie eines Buttons wird mit dessen Position, d.h. den X-/Y-Koordinaten (0/0 ist die linke obere Ecke des Bildschirms) sowie mit dessen Breite (W) und Höhe (H) bestimmt.

   Diese vier Geometrie-Parameter werden in eigerScript den dafür bestimmten Geometrie-Registern zugewiesen:



eigerScript-Code 2: Zuweisung der Button-Geometrie zu den entsprechenden Registern

```
eI.Pos_X1  := 20     ; Zuweisung der X-Position ins Register eI.Pos_X1
eI.Pos_Y1  := 30     ; Zuweisung der Y-Position ins Register eI.Pos_Y1
eI.Width   := 180    ; Zuweisung der Button-Breite ins Register eI.Width
eI.Height  := 80     ; Zuweisung der Button-Höhe ins Register eI.Height
```

Es ist oft von Vorteil, die einzelnen Werte zu Beginn als Konstanten mit einleuchtenden Namen zu definieren:

eigerScript-Code 3: Deklaration von Konstanten für die Button-Geometrie

```
CONST   Button_X        = 20     ; Konstanten-Deklaration für die X-Position
CONST   Button_Y        = 30     ; Konstanten-Deklaration für die Y-Position
CONST   ButtonWidth     = 180    ; Konstanten-Deklaration für die Button-Breite
CONST   ButtonHeight    = 80     ; Konstanten-Deklaration für die Button-Höhe
```

Die vier Zuweisungszeilen von eigerScript-Code 2 können in einer einzigen Zeile, d.h. in einem eigerScript-Befehl zusammengefasst werden:

eigerScript-Code 4: Zuweisung der Button-Geometrie zu den entsprechenden Registern, mit dem gleichen Effekt wie eigerScript-Code 2

```
Load.Geometry_XYWH(20, 30, 180, 80)
```

oder, nachdem für die Button-Geometrie entsprechende Konstanten definiert worden sind (vgl. eigerScript-Code 3):

```
Load.Geometry_XYWH(Button_X, Button_Y, ButtonWidth, ButtonHeight)
```

Anstelle der Befehle von eigerScript-Code 4 können die Geometrie-Eigenschaften auch direkt im Style eingegeben werden (vgl. Punkt 2).

2. Weitere Button-Eigenschaften festlegen mit Hilfe eines Styles

„Styles“ sind der effizienteste Weg, die Eigenschaften eines Labels – in unserem Fall eines Buttons – festzulegen. Ein Style ist in eigerScript eine einfache Auflistung der Buttoneigenschaften und Ihrer Werte (vgl. eigerScript-Code 5). Jede Zeile ist für eine bestimmte Eigenschaftszuweisung ins entsprechende Register zuständig. Ein Style ist ein fester Zeilenblock, d.h. klar definiert durch die Anzahl der aufgelisteten Label-Eigenschaften sowie durch die Reihenfolge innerhalb der Liste. Daran darf nichts verändert werden. Einzig die Werte der Eigenschaften können variieren.

Die Styles benennen wir mit einer sinngemässen Bezeichnung und platzieren diese immer innerhalb einer Style-Subroutine (z.B. `SUB Styles`, vgl. eigerScript-Code 5). Üblicherweise fassen wir mehrere oder gleich alle Styles einer View in einer einzigen Subroutine zusammen.

Ein Beispiel einer Style-Subroutine mit zwei Button-Styles ist in eigerScript-Code 5 gezeigt. Die Subroutine „Styles“ enthält in diesem Beispiel die beiden Styles für den „ruhenden“ und den „eingedrückten“ Button-Zustand.

eigerScript-Code 5: Style-Subroutinen mit zwei Styles und deren Bezeichnung – ein Style für den inaktiven Button und ein Style für den „eingedrückten“ Button.

```
SUB Styles

Button_UP_Style:
     INLINEWORDS    (no_change)                          ; entspricht eI.Pos_X1
     INLINEWORDS    (no_change)                          ; entspricht eI.Pos_Y1
     INLINEWORDS    (no_change)                          ; entspricht eI.Width
     INLINEWORDS    (no_change)                          ; entspricht eI.Height
     INLINEWORDS    (8)                                  ; entspricht eI.SpaceLeft
     INLINEWORDS    (8)                                  ; entspricht eI.SpaceRight
     INLINEWORDS    (0)                                  ; entspricht eI.HorizontalAdjust
     INLINEWORDS    (0)                                  ; entspricht eI.VericalAdjust
     INLINEWORDS    (lightblue)                          ; entspricht eI.FillColor
     INLINEWORDS    (as_FillColor)                       ; entspricht eI.BackColor
     INLINEWORDS    (as_FillColor)                       ; entspricht eI.LineColor
     INLINEWORDS    (autocolor)                          ; entspricht eI.TextColor
     INLINEWORDS    (Pos_left)                           ; entspricht eI.Position
     INLINEWORDS    (Orientation_0deg)                   ; entspricht eI.Orientation
     INLINEWORDS    (normal)                             ; entspricht eI.Appearance
     INLINEWORDS    (border_color_button_soft_raised_big ) ; entspricht eI.BorderStyle
     INLINEWORDS    (Font_System_9bd)                    ; entspricht eI.FontNumber
     INLINEWORDS    (as_Skin_FormBodyColor)              ; entspricht eI.BackgroundColor
```

```
Button_DN_Style:
     INLINEWORDS    (no_change)                              ; entspricht eI.Pos_X1
     INLINEWORDS    (no_change)                              ; entspricht eI.Pos_Y1
     INLINEWORDS    (no_change)                              ; entspricht eI.Width
     INLINEWORDS    (no_change)                              ; entspricht eI.Height
     INLINEWORDS    (8)                                      ; entspricht eI.SpaceLeft
     INLINEWORDS    (8)                                      ; entspricht eI.SpaceRight
     INLINEWORDS    (1)                                      ; entspricht eI.HorizontalAdjust
     INLINEWORDS    (1)                                      ; entspricht eI.VericalAdjust
     INLINEWORDS    (lightgreen)                             ; entspricht eI.FillColor
     INLINEWORDS    (as_FillColor)                           ; entspricht eI.BackColor
     INLINEWORDS    (as_FillColor)                           ; entspricht eI.LineColor
     INLINEWORDS    (autocolor)                              ; entspricht eI.TextColor
     INLINEWORDS    (Pos_left)                               ; entspricht eI.Position
     INLINEWORDS    (Orientation_0deg)                       ; entspricht eI.Orientation
     INLINEWORDS    (normal)                                 ; entspricht eI.Appearance
     INLINEWORDS    (border_color_button_soft_sunk_big )     ; entspricht eI.BorderStyle
     INLINEWORDS    (Font_System_9bd)                        ; entspricht eI.FontNumber
     INLINEWORDS    (as_Skin_FormBodyColor)                  ; entspricht eI.BackgroundColor

ENDSUB
```

Anstelle eines Styles können wir die Eigenschaften eines Labels oder Buttons auch einzeln den entsprechenden Registern zweisen. So bedeutet beispielsweise die neunte Zeile des Button_DN_Styles von eigerScript-Code 5 nichts anderes als:

eigerScript-Code 6: Dem Register eI.FillColor wird „lightgreen“ zugewiesen. Dadurch erhält der Button (im gedrückten Zustand) die grüne Farbe.

```
eI.FillColor := lightgreen
```

Oft füllt man auch mit Hilfe eines Styles erst mal alle Eigenschaften auf einmal ab und ändert danach noch einzelne Eigenschaften, die für den nächst zu zeichnenden Button abweichen (vgl. eigerScript-Code 8, S.6).

3. Button-Eigenschaften laden

Bevor der Button dann gezeichnet werden soll, laden wir die Button-Eigenschaften in die entsprechenden Register, indem wir auf den gewünschten Style verweisen. Dazu verwenden wir folgenden eigerScript-Befehl:

eigerScript-Code 7: eigerScript-Befehl, mit welchem die Eigenschaften geladen werden, welche im „Button_UP_Style“ vordefiniert sind.

```
Fill.LabelParameter(Button_UP_Style)    ; zeigt unberührten Button
```

Wenn wir gleich mehrere Buttons mit demselben Style definieren wollen, legen wir die Geometrie nicht im Style fest, da zumindest die Position ja für jeden Button unterschiedlich ist. Diese Style-Zeilen dürfen wir aber nicht einfach weglassen, weil sie unzertrennlich zum Style gehören. In diesem Fall fügen wir innerhalb des Styles anstelle konkreter Werte die Konstante `no_change` ein. Dadurch bleiben beim Laden eines Styles die mit `no_change` definierten Register unverändert.

Wir können auch konkrete Wert-Zuweisungen im Style belassen und nach dem Laden des Styles die vom Style abweichenden Geometrie-Werte für den konreten Button einfach nochmals separat laden. Damit wird dann z.B. der Inhalt der Geometrie-Register einfach nochmals mit den gewünschten Werten überschrieben (vgl. eigerScript-Code 8).

eigerScript-Code 8: Zunächst werden mit dem Style „Button_UP_Style“ gleichzeitig alle Button-Eigenschaften geladen und danach im speziellen die Geometrie-Register nochmals mit den Geometrie-Werten des nächsten Buttons überschrieben.

```
Fill.LabelParameter(Button_UP_Style)    ; zeigt unberührten Button
Load.Geometry_XYWH(Button_X, Button_Y, ButtonWidth, ButtonHeight)
```

4. HotSpot des berührungsempfindlichen Bereichs festlegen

Wenn die Geometrie-Register für einen Button geladen sind, können wir mit folgendem Befehl den Bereich des Buttons als berührungsempfindlichen HotSpot installieren:

eigerScript-Code 9: eigerScript-Methode zum Installieren eines HotSpots, je nach Art des Berührungs-Events wird eine andere Subroutine aufgerufen.

```
HotSpot.Install(NIL,Subroutine_1,Subroutine_2,Subroutine_3)
```

Enter-Event **Leave**-Event **Down**-Event **Up**-Event

Zusammen mit dem Installationsbefehl legen wir auch fest, welche Aktion bei den verschiedenen Berührungs-Events ausgelöst werden soll. Dazu können wir innerhalb der Klammer Adressen angeben – sog. „LabelRelative24“ –, zu denen die Anwendung bei Eintreffen eines Events springen soll. Als Adressen benutzen wir die Namen der Subroutinen, in denen dann ein bestimmter Aktionsablauf ausgeführt wird. Diese Subroutinen bezeichnen wir auch als Event-Handler. Ein EventHandler zeichnet beispielsweise den „eingedrückten“ Button, führt einen View-Wechsel aus oder versendet einen String über die COM2-Schnittstelle.

Im eigerScript-Code 9 ist für den Enter-Event die Konstante `NIL` eingetragen. Dies bedeutet „not implemented“, d.h. es ist für diesen Event keine Reaktion vorgesehen.

Der Befehl `HotSpot.Install()` ist in eigerScript-Code 1 (S.2) direkt in der Subroutine enthalten, mit welcher der Button dann auch gezeichnet ist. Es ist jedoch häufig besser, die Installation des HotSpots ausserhalb der Button-Subroutine vorzunehmen. Während ein animierter Button bei jeder Berührung neu gezeichnet werden muss, genügt es, den HotSpot nur einmal zu definieren. In der Beispiel-Anwendung BTTN (vgl. eigerScript-Code 10) wird beim Aufbau der View zuerst vom Hauptprogramm aus ein Button mit Hilfe des entsprechenden Subroutinen-Aufrufs gezeichnet. Gleich nachdem die Subroutine abgearbeitet, d.h. der Button gezeichnet ist, wird dann ebenfalls im Hauptprogramm der HotSpot installiert. Zu diesem Zeitpunkt sind dann noch alle Geometrie-Register mit den Werten des Buttons gefüllt. Dadurch kann diese Geometrie eleganterweise auch noch für den HotSpot verwendet werden, bevor ein weiterer Button gezeichnet wird und sich die Register-Inhalte wieder verändern.

Source Code für animierte Buttons auf dem eigerPanel:

eigerScript-Code 10: Source Code der Beispiels-Applikation BTTN mit animierten Buttons.

```
;-------------------------------------------------------------------
; Titel       : Animierte Buttons (Script-Beispiel)   View 1
; File        : BTTN_001.EVS
;-------------------------------------------------------------------
; Compiler    : eigerScript
;
; System      : eigergraphics.com; FOX embedded computer
;
; Beschreibung: Dieses Projekt zeigt die Programmierung animierter Buttons
;
; Version     : Initialversion: 10.01.2011
;
; Autor       : Christoph Angst, S-TEC electronics AG
;
;-------------------------------------------------------------------
; (c) 2005-2011     S-TEC electronics AG, CH-6314 Unterägeri; +41 41 754 50 10
;-------------------------------------------------------------------


EIGERPROJECT 'BTTN' ; Projektbezeichnung: erster Teil des EVI-Dateinamens
EIGERVIEW 1         ; Viewnummer: zweiter Teil des EVI-Dateinamens: BTTN_001.EVI

IMPORT       'DEF/DEF_eVM_OpCodes.h'          ; Token
IMPORT       'DEF/DEF_eVM_Registers.h'        ; Register
FUNCLIB      'DEF/DEF_eVM_Functions.lib'            ; Funktionsbibliothek

INCLUDEFILE  'DEF/DEF_eiger_Colors.INC'       ; Farbdefinitionen 12.03.2006
INCLUDEFILE  'DEF/DEF_eiger_Types.INC'        ; eiger Definitionen
INCLUDEFILE  'DEF/DEF_eiger_StackMachine.INC' ; Stack Machine


; D E K L A R A T I O N E N ____________________________________________

; Geometrie für Buttons ..................................................
CONST        ButtonWidth  = 190  ; Button-Breite
CONST        ButtonHeight = 85   ; Button-Höhe
CONST        Space_X      = 40   ; Button-Abstand in X-Richtung
```

```
CONST         Space_Y       = 20   ; Button-Abstand in Y-Richtung


CONST         ButtonLeft_X = 20
CONST         ButtonMid_X = ButtonLeft_X + ButtonWidth + Space_X
CONST         ButtonRight_X = ButtonLeft_X + 2*(ButtonWidth + Space_X)

CONST         ButtonRow01_Y = 30                                  ; Y-Koord für Button-Zeile 1
CONST         ButtonRow02_Y = ButtonRow01_Y + ButtonHeight + Space_Y     ; Y-Koord für Button-Zeile 2
CONST         ButtonRow03_Y = ButtonRow02_Y + ButtonHeight + Space_Y     ; Y-Koord für Button-Zeile 3



; S U B R O U T I N E N ____________________________________________________


; Buttons zeichnen ..........................................................


; Button 1:

STRING [32] B01_Text = 'Button 1'                    ; String-Deklaration für Button-Aufschrift

SUB    B01_Leave
       Fill.LabelParameter(Button_UP_Style)          ; zeigt unberührten Button
       Load.Geometry_XYWH(ButtonLeft_X,ButtonRow01_Y,ButtonWidth,ButtonHeight)
       Label.Text(B01_Text)
ENDSUB


SUB    B01_Down
       Fill.LabelParameter(Button_DN_Style)          ; zeigt eingedrückten Button
       Load.Geometry_XYWH(ButtonLeft_X,ButtonRow01_Y,ButtonWidth,ButtonHeight)
       Label.Text(B01_Text)
ENDSUB


SUB    B01_Up                                        ; Subroutine wird aufgerufen bei Up-Ereignis
       CallSubroutine(B01_Leave)
       ;GotoView(2)                                  ; Wechsel zur View BTTN_002.EVS (gibt es noch nicht)
ENDSUB


;___________________________________________________________________________
```

```
; Button 2:

STRING [32] B02_Text = 'Button 2'                    ; String-Deklaration für Button-Aufschrift

SUB    B02_Leave                                     ; Subroutine wird aufgerufen bei Leave-Ereignis
       Fill.LabelParameter(Button_UP_Style)          ; zeigt unberührten Button
       Load.Geometry_XYWH(ButtonLeft_X,ButtonRow02_Y,ButtonWidth,ButtonHeight)
       Label.Text(B02_Text)

ENDSUB


SUB    B02_Down
       Fill.LabelParameter(Button_DN_Style)          ; zeigt eingedrückten Button
       Load.Geometry_XYWH(ButtonLeft_X,ButtonRow02_Y,ButtonWidth,ButtonHeight)
       Label.Text(B02_Text)

ENDSUB


SUB    B02_Up                                        ; Subroutine wird aufgerufen bei Up-Ereignis
       CallSubroutine(B02_Leave)
       ;GotoView(3)                                  ; Wechsel zur View BTTN_003.EVS (gibt es noch nicht)

ENDSUB


;_____________________________________________________________________________


; Button 3:

STRING [32] B03_Text = 'Button 3'                    ; String-Deklaration für Button-Aufschrift

SUB    B03_Leave                                     ; Subroutine wird aufgerufen bei Leave-Ereignis
       Fill.LabelParameter(Button_UP_Style)          ; zeigt unberührten Button
       Load.Geometry_XYWH(ButtonMid_X,ButtonRow01_Y,ButtonWidth,ButtonHeight)
       Label.Text(B03_Text)
ENDSUB


SUB    B03_Down
       Fill.LabelParameter(Button_DN_Style)          ; zeigt eingedrückten Button
```

```
       Load.Geometry_XYWH(ButtonMid_X,ButtonRow01_Y,ButtonWidth,ButtonHeight)
       Label.Text(B03_Text)
ENDSUB


SUB    B03_Up                                          ; Subroutine wird aufgerufen bei Up-Ereignis
       CallSubroutine(B03_Leave)
       ;GotoView(4)                                    ; Wechsel zur View BTTN_004.EVS (gibt es noch nicht)
ENDSUB


; Button 4:

STRING [32] B04_Text = 'Button 4'                      ; String-Deklaration für Button-Aufschrift

SUB    B04_Leave                                       ; Subroutine wird aufgerufen bei Leave-Ereignis
       Fill.LabelParameter(Button_UP_Style)            ; zeigt unberührten Button
       Load.Geometry_XYWH(ButtonMid_X,ButtonRow02_Y,ButtonWidth,ButtonHeight)
       Label.Text(B04_Text)
ENDSUB


SUB    B04_Down
       Fill.LabelParameter(Button_DN_Style)            ; zeigt eingedrückten Button
       Load.Geometry_XYWH(ButtonMid_X,ButtonRow02_Y,ButtonWidth,ButtonHeight)
       Label.Text(B04_Text)
ENDSUB


SUB    B04_Up                                          ; Subroutine wird aufgerufen bei Up-Ereignis
       CallSubroutine(B04_Leave)
       ;GotoView(5)                                    ; Wechsel zur View BTTN_005.EVS (gibt es noch nicht)
ENDSUB


; S T Y L E S ____________________________________________________________________

SUB Styles

Button_UP_Style:
       INLINEWORDS   (no_change)               ; entspricht eI.Pos_X1
       INLINEWORDS   (no_change)               ; entspricht eI.Pos_Y1
       INLINEWORDS   (ButtonWidth)             ; entspricht eI.Width
       INLINEWORDS   (ButtonHeight)            ; entspricht eI.Height
```

```
       INLINEWORDS   (8)                       ; entspricht eI.SpaceLeft
       INLINEWORDS   (8)                       ; entspricht eI.SpaceRight
       INLINEWORDS   (0)                       ; entspricht eI.HorizontalAdjust
       INLINEWORDS   (0)                       ; entspricht eI.VericalAdjust
       INLINEWORDS   (lightblue)               ; entspricht eI.FillColor
       INLINEWORDS   (as_FillColor)            ; entspricht eI.BackColor
       INLINEWORDS   (as_FillColor)            ; entspricht eI.LineColor
       INLINEWORDS   (blue)                    ; entspricht eI.TextColor
       INLINEWORDS   (Pos_left)                ; entspricht eI.Position
       INLINEWORDS   (Orientation_0deg)        ; entspricht eI.Orientation
       INLINEWORDS   (normal)                  ; entspricht eI.Appearance
       INLINEWORDS   (border_color_button_soft_raised_big ) ; entspricht eI.BorderStyle
       INLINEWORDS   (Font_System_18bd)        ; entspricht eI.FontNumber
       INLINEWORDS   (as_Skin_FormBodyColor)   ; entspricht eI.BackgroundColor


Button_DN_Style:
       INLINEWORDS   (no_change)               ; entspricht eI.Pos_X1
       INLINEWORDS   (no_change)               ; entspricht eI.Pos_Y1
       INLINEWORDS   (ButtonWidth)             ; entspricht eI.Width
       INLINEWORDS   (ButtonHeight)            ; entspricht eI.Height
       INLINEWORDS   (8)                       ; entspricht eI.SpaceLeft
       INLINEWORDS   (8)                       ; entspricht eI.SpaceRight
       INLINEWORDS   (1)                       ; entspricht eI.HorizontalAdjust
       INLINEWORDS   (1)                       ; entspricht eI.VericalAdjust
       INLINEWORDS   (lightgreen)              ; entspricht eI.FillColor
       INLINEWORDS   (as_FillColor)            ; entspricht eI.BackColor
       INLINEWORDS   (as_FillColor)            ; entspricht eI.LineColor
       INLINEWORDS   (green)                   ; entspricht eI.TextColor
       INLINEWORDS   (Pos_left)                ; entspricht eI.Position
       INLINEWORDS   (Orientation_0deg)        ; entspricht eI.Orientation
       INLINEWORDS   (normal)                  ; entspricht eI.Appearance
       INLINEWORDS   (border_color_button_soft_sunk_big ) ; entspricht eI.BorderStyle
       INLINEWORDS   (Font_System_18bd)        ; entspricht eI.FontNumber
       INLINEWORDS   (as_Skin_FormBodyColor)   ; entspricht eI.BackgroundColor

ENDSUB


; H A U P T P R O G R A M M ____________________________________________________
BEGINVIEW
       EVE.Init()                      ; EVE ANNA initialisieren
       Load.Pos_X1Y1(0,0)              ; linke obere Ecke
       Display.Prepare()               ; Nachfolgender Layout-Aufbau zunächst nur im unsichtbaren Videospeicher (AVR)
```

```
      IF eI.DisplayWidth == 640 THEN   ; Hat das eigerPanel ein VGA- oder ein WVGA Display?
        File.Read_EGI('C:\\BTTN\\PICT\\BCKGRND1.EGI')      ; zeichnet VGA-Hintergrund-Bild "BCKGRND1.EGI"
      ELSE
        File.Read_EGI('C:\\BTTN\\PICT\\BCKGRND2.EGI')      ; zeichnet WVGA-Hintergrund-Bild "BCKGRND2.EGI"
      ENDIF


      ; Button für ScreenShot  (Code ist in der Includedatei "PRINTSCREEN.EVS".............
      CallSubroutine(PrintScreen_Init)

      ; Buttons und HotSpots generieren ....................................................
      ; (die Geometrie für HotSpots ist nach dem Ausführen der Subroutinen bereits vorgeladen)

      CallSubroutine(B01_Down)               ; Button 1 zeichnen
      HotSpot.Install(NIL,B01_Leave,B01_Down,B01_Up) ; Subroutinen-Aufruf bei Leave-, Down- und Up-Ereignis

      CallSubroutine(B02_Leave)              ; Button 2 zeichnen
      HotSpot.Install(NIL,B02_Leave,B02_Down,B02_Up) ; Subroutinen-Aufruf bei Leave-, Down- und Up-Ereignis

      CallSubroutine(B03_Leave)              ; Button 3 zeichnen
      HotSpot.Install(NIL,B03_Leave,B03_Down,B03_Up) ; Subroutinen-Aufruf bei Leave-, Down- und Up-Ereignis

      CallSubroutine(B04_Leave)              ; Button 4 zeichnen
      HotSpot.Install(NIL,B04_Leave,B04_Down,B04_Up) ; Subroutinen-Aufruf bei Leave-, Down- und Up-Ereignis


      ; Layout vom unsichtbaren Videospeicher AVR in den sichtbaren Videospeicher RVR kopieren - die View erscheint auf Display
      Display.Show()    ; Im AVR aufgebautes Bild als Ganzes in den Refresh Video Speicher RVR kopieren und damit sichtbar machen.

      HotSpot.TableEnable()     ; Macht die HotSpots "berührungsempfindlich"

      LOOP
      ENDLOOP
ENDVIEW
```