23.7 Komponente gb.scanner

Die Gambas-Scanner-Komponente gb.scanner besteht aus den drei Klassen

- Scanners,
- Scanner und
- ScannerOption.

Sie ermöglicht die einfache Verwaltung und Steuerung von Scanner-Geräten im Gambas-Code. Die Klassen nutzen das Programm *ScanImage*, das im SANE-Projekt als zeilenkommando-orientiertes Frontend-Tool bereitgestellt wird.

23.7.1 Klasse Scanners

Um Scanner-Objekte anlegen und mit ihnen arbeiten zu können, müssen Sie zunächst in Erfahrung bringen, wie sich der eingesetzte Scanner als Gerät im System registriert hat. Hierfür steht Ihnen die Klasse *Scanners* zur Verfügung. Die Klasse können Sie erzeugen und agiert als Read-Only-Array, das eine Liste des Objekt-Typs Scanner enthält und mittels FOR-EACH enumerabel ist. Sie stellt keine Eigenschaften zur Verfügung, verfügt aber über zwei Methoden und ein Ereignis.

23.7.1.1 Methoden

Die Methoden der Klasse Scanners:

Methode	Beschreibung
Close ()	Diese statische Methode schließt einen eventuell laufenden Prozess von *ScanImage*.
Search ([bWait as Boolean])	Veranlasst die Suche von Scannern. Wird die Option mit bWait = True verwendet, blockiert die Anwendung solange, bis die Scanner-Liste vollständig ist. Wird die Option nicht verwendet oder bWait = False gesetzt, dann erfolgt die Suche im Hintergrund.

Tabelle 23.7.1.1.1 : Methoden der Klasse Scanners

23.7.1.2 Ereignisse

Die Klasse *Scanners* besitzt nur dieses Ereignis:

Ereignis	Beschreibung
Found ( )	Das Ereignis wird ausgelöst, wenn die Suche nach Scannern abgeschlossen wurde.

Tabelle 23.7.1.2.1 : Ereignis der Klasse Scanners

Der durch die Methode Search veranlasste Suchprozess erfolgt im Hintergrund und endet im Ereignis Scanners_Found(), in dem die Auswertung stattfinden kann. Damit die Klasse dieses Ereignis in der Mutterklasse auslösen kann, muss es mit der Anweisung Objekt.Attach entsprechend registriert werden:

```
'-- Attach Scanner static class to the form class to catch events
Object.Attach(Scanners, Me, "Scanners")

'-- Start searching for scanners
Scanners.Search
...
```

```
'' End of search and enumeration of scanners
Public Sub Scanners_Found()
  Dim s As String

  For Each s In Scanners
    ComboBoxScanners.Add(Scanners[s].name)
  Next
  If ComboBoxScanners.count > 0 Then
     ComboBoxScanners.Index = 0
```

```
  Else
    Message.Error("No scanner found.")
    Quit
  Endif

End
```

Alternativ lässt sich das auch in der gewohnten Weise umsetzen:

```
Dim MyScanners As Scanners
MyScanners = New Scanners As "MyScanners"
MyScanners.Search()
...
```

```
'' End of search and enumeration of scanners
Public Sub MyScanners_Found()

Dim s As String

  For Each s In Scanners
    ComboBoxScanners.Add(Scanners[s].Name)
  Next
  If ComboBoxScanners.count > 0 Then
     ComboBoxScanners.Index = 0
  Else
     Message.Error("No scanner found.")
     Quit
  Endif

End
```

23.7.2 Klasse Scanner

Die Klasse Scanner repräsentiert einen Scanner als Gerät und liefert mit dem Scanner-Objekt die Grundlage für die Arbeit mit dem Gerät.

23.7.2.1 Eigenschaften

Die Klasse *Scanner* verfügt über diese Eigenschaften:

Eigenschaft	Datentyp	Beschreibung
Async	Boolean	Gibt den zu verwendenden Prozess-Modus zurück oder legt ihn fest. Im Falle von True werden alle Prozesse im Hintergrund abgewickelt, während beim Default-Wert False der Programmablauf für die Zeit des Prozesses angehalten wird.
Debug	Boolean	Gibt den Debug-Modus zurück oder legt ihn fest. Bei True werden die intern verwendeten *ScanImage*-CLI-Kommandos in der IDE-Konsole für Kontrollzwecke ausgegeben. Der Default-Wert ist False.
Model	String	Gibt das Modell des Scanners zurück.
Name	String	Gibt den Geräte-Namen des Scanners zurück.
Progress	Float	Gibt den Fortschritt des Scan-Prozesses zurück. Der Wert liegen zwischen 0 und 1; entsprechend 0% bis 100%.
Type	String	Dieser Wert wird vom Scanner geliefert und und gibt den Typ des Scanners zurück, zum Beispiel "flatbed scanner".
Vendor	String	Dieser Wert wird vom Scanner geliefert und gibt den Herstellernamen zurück.

Tabelle 23.7.2.1.1 : Eigenschaften der Klasse Scanner

So erzeugen Sie ein neues Scanner-Objekt:

```
Dim hScanner As Scanner
hScanner = New Scanner ( sDevice As String )  As "EventName"
```

Auf Basis der oben beschriebenen Erkennung vorhandener Scanner könnte dies konkret wie folgt aussehen:

```
Dim hScanner As Scanner
hScanner = New Scanner(ComboBoxScanners.Text) As "hScanner"
hScanner.Async = True
```

Die Klasse ist mit FOR-EACH enumerabel und liefert dabei die Namen aller verfügbaren Scanner-Optionen:

```
Dim sOption As String

For Each sOption In hScanner
  Print sOption
Next
```

23.7.2.2 Methoden

Die Klasse *Scanner* verfügt über folgende Methoden:

Methode	Rückgabetyp	Beschreibung
Exist (Key As String)	Boolean	Gibt zurück, ob der Scanner über eine Eigenschaft/Funktion verfügt, die durch einen Key, zum Beispiel "Contrast" oder "Brightness" definiert wird.
Find (Key As String)	ScannerOption	Führt eine Suche nach einer Scanner-Option durch, die durch den Key definiert wird. Bei Erfolg wird eine Scanner-Option zurück gegeben. Bei Misserfolg ist die Scanner-Option leer.
IsAvailable ( )	Boolean	Gibt durch True zurück, ob der Scanner verfügbar ist.
Peek ( )	Image	Gibt das gescannte Image zurück. Die Methode kann im Modus Async = True nur im PageEnd-Ereignis oder im Modus Async = False nach Abschluss des Scan-Prozesses sinnvoll verwendet werden.
Scan ( )	Image	Startet den Scan-Prozess.

Tabelle 23.7.2.2.1 : Methoden der Klasse Scanner

Mit Hilfe der Scanner-Klasse können Sie u.a. feststellen, ob der eingesetzte Scanner die Einstellung der *Helligkeit* anbietet:

```
If hScanner.Exist("Brightness") Then
   Print hScanner["Brightness"].MinValue       ' Minimum value of this option
   Print hScanner["Brightness"].MaxValue       ' Maximum value of this option
   Print hScanner["Brightness"].Value          ' Default value of this option
Endif
```

Alternative:

```
Print hScanner.Find("Brightness").MinValue
Print hScanner.Find("Brightness").MaxValue
Print hScanner.Find("Brightness").Value
```

23.7.2.3 Ereignisse

Die Klasse *Scanner* besitzt diese Ereignisse:

Ereignis	Beschreibung
Begin ( )	Diese Ereignis wird unmittelbar nach Anwendung der Scan-Methode ausgelöst.
End ( )	Dieses Ereignis ist das letzte eines Scan-Prozesses und wird nach dem Finished()-Ereignis ausgelöst.
Error (ErrorText As String)	Dieses Ereignis wird bei Fehlern ausgelöst. Die Variable *ErrorText* liefert die dazugehörige Fehlermeldung.
Finished ( )	Dieses Ereignis wird am Ende des Seiten-Scans ausgelöst und schließt gegebenenfalls den Rücklauf der Scanner-Mechanik mit ein.
PageBegin ( )	Dieses Ereignis wird unmittelbar vor dem Scan einer Seite ausgelöst.
PageEnd ( )	Dieses Ereignis wird unmittelbar nach dem Scan einer Seite ausgelöst. Die Ereignis-Routine wird typischerweise dazu verwendet, das gescannte Image auszulesen.
Progress ( )	Dieses Ereignis wird während des Scans periodisch zwecks Fortschrittskontrolle

Ereignis	Beschreibung
	ausgelöst. Innerhalb der Ereignis-Routine steht mit *Last.Progress* ein Wert zwischen 0 und 1 (entsprechend 0% bis 100%) zur Fortschrittskontrolle zur Verfügung.

Tabelle 23.7.2.3.1 : Ereignisse der Klasse Scanner

Hier ein Beispiel, wie man in der PageEnd-Ereignis-Routine das gescannte Image liest und anzeigt:

```
Public Sub hScanner_PageEnd()

  Dim hImage As Image

  Try hImage = Last.Peek()

  If Not hImage Then
     Message.Error("Can't load image")
     Return
  Endif

  PictureBoxScan.Picture = hImage.Picture

End
```

23.7.3 Klasse ScannerOption

Die verfügbaren Optionen eines Scanners variieren erfahrungsgemäß sehr stark zwischen den Modellen und den Herstellern. Es obliegt dem Programmierer, diese zu ermitteln und entsprechend einzusetzen. Hierfür steht die Klasse *ScannerOptionen* zur Verfügung, die mithilfe eines Parsers versucht, alle verfügbaren Optionen als entsprechende Eigenschaften zur Verfügung zu stellen.

Hierzu erhält der Parser einen scanner-spezifischen Datensatz vom Programm *ScanImage* zugespielt. Dieser sieht für einen (fiktiven) Scanner beispielsweise so aus

```
All options specific to device `quickscan:QS2000_192.xxx.xxx.xxx':
  Scan mode:
    --resolution 75|150|300|600|1200|2400|4800dpi [75]
        Sets the resolution of the scanned image.
    --mode auto|Color|Gray|Lineart [Color]
        Selects the scan mode (e.g., lineart, monochrome, or color).
    --source Flatbed [Flatbed]
        Selects the scan source (such as a document-feeder). Set source before
        mode and resolution. Resets mode and resolution to auto values.
  Geometry:
    -l auto|0..216.069mm [0]
        Top-left x position of scan area.
    -t auto|0..297.011mm [0]
        Top-left y position of scan area.
    -x auto|0..216.069mm [216.069]
        Width of scan-area.
    -y auto|0..297.011mm [297.011]
        Height of scan-area.
  Extras:
    --threshold auto|0..100% (in steps of 1) [inactive]
...
```

und liefert dem Parser neben vielen anderen Parametern auch Werte für einstellbare Auflösungen (rot markiert), die als Eigenschaft zur Verfügung gestellt werden.

Werte für die verfügbaren Auflösungen können im Gegensatz zu den diskreten Werten im obigen Beispiel auch als Bereich ("Range") spezifiziert sein, wie das folgenden Beispiel zeigt:

```
...
  --resolution 50..1200dpi [50]
...
```

Zur Nutzung der Auflösungswerte ist somit eine Unterscheidung dieser Fälle erforderlich. Hierfür bietet die Klasse die Eigenschaft *IsRange*.

Die Optionen der Datengruppe "Geometry" sind zur Festlegung des zu scannenden Ausschnitts vorgesehen. Hierfür gibt die Klasse feste Namen vor:

- Left = Abstand zum linken Rand (in mm)
- Top = Abstand zum oberen Rand (in mm)
- Width = Breite des Scans (in mm)
- Height = Höhe des Scans (in mm)

23.7.3.1 Eigenschaften

Die Klasse *ScannerOption* verfügt über folgende Eigenschaften, wobei ihre Werte mit Ausnahme der Eigenschaft *Value* nur gelesen werden können:

Eigenschaft	Datentyp	Beschreibung
Group	String	Gibt den Namen der Datengruppe zurück, in der sich die Option befindet, wie etwa "Scan Mode" für die Option "Resolution".
Info	String	Gibt Informationen zu einer Option zurück - sofern verfügbar.
IsActive	Boolean	Gibt zurück, ob eine Option aktiv ist.
IsRange	Boolean	Gibt zurück, ob die Option als Bereich zwischen 2 Werten angegeben wird – im Gegensatz zu diskreten Einzelwerten.
List	String[]	Gibt ein String-Array zurück, das eine Liste diskreter Werte einer Option enthält.
MaxValue	Float	Gibt den ermittelten Maximalwert der Option zurück, wie zum Beispiel 4800 • dpi als größte Auflösung.
MinValue	Float	Gibt den ermittelten Minimalwert der Option zurück, wie zum Beispiel 75 • dpi als geringste Auflösung.
Modified	Boolean	Gibt zurück, ob der Wert der Eigenschaft "Value" verändert wurde.
Name	String	Gibt den Namen der Option zurück.
Steps	Integer	Gibt die Schrittgröße zurück, mit der die Option zwischen MinValue und MaxValue aufgelöst wird.
Unit	String	Gibt die Maßeinheit für die Option zurück, sofern diese über Zahlenwerte definiert ist, wie zum Beispiel "mm" für die Option "Width".
Value	Variant	Gibt den aktuellen Wert der Option zurück oder setzt ihn.

Tabelle 23.7.3.1.1 : Eigenschaften der Klasse ScannerOption

Hinweis:

- Der Name der Option beginnt immer mit einem Großbuchstaben wie zum Beispiel "Resolution". Das gilt auch, wenn die vom Scanner gelieferte Option mit einem Kleinbuchstaben beginnt!

23.7.4 Das Projekt gb.Scan

In dem Projekt *gb.Scan*, dessen Quelltext-Archiv als Download zur Verfügung gestellt wird, werden ausschließlich native Mittel von Gambas eingesetzt, um eine möglichst vielseitige Scan-Applikation für den täglichen Bedarf vorzustellen.

Die Anwendung gb.Scan verfügt über die folgenden Funktionen:

- Scannen ganzer Seiten oder markierter Ausschnitte.
- Scannen in den Formaten A4, A5, A6, US Letter und nutzer-definiert.
- Speicherung einseitiger oder mehrseitiger Scans im JPG-, PNG-, BMP- oder PDF-Format.
- Versendung einseitiger oder mehrseitiger Scans im PDF-Format als EMail-Anhang.
- Drucken einseitiger oder mehrseitiger Scans.
- Optionale OCR-Funktion zur Erstellung durchsuchbarer PDFs (experimentell).
- Auswahl des Scanners, falls mehr als ein Scanner verfügbar ist.
- Abspeichern und Wiederherstellung der eingestellten Parameter für jeden angeschlossenen bzw. ausgewählten Scanner.
- Wiederherstellung der Größe und Position der Applikationsfenster.
- Darstellung der Scans in einer Multi-Select-Liste mit

- Drehen der Scans in 90°-Schritten,
- Ändern der Reihenfolge der Scans. Beispiel: Vor der Erzeugung einer mehrseitigen PDF-Datei,
- Löschen von Scans. Beispiel: Vor der Erzeugung einer mehrseitigen PDF-Datei.

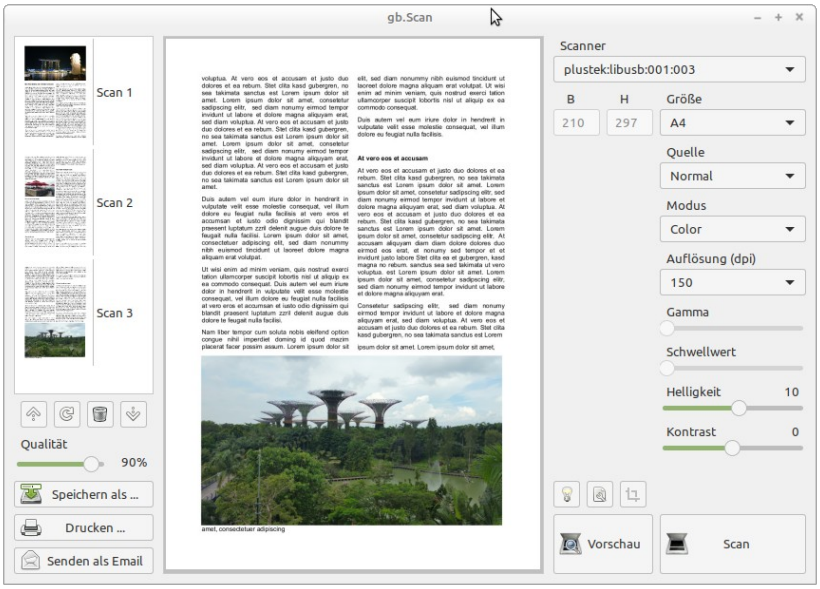

Abbildung 23.7.4.1: Hauptfenster der Applikation gb.Scan

Hinweise:

- Die Applikation erfordert QT5.
- Die Applikation verwendet derzeit den Quelltext der beschriebenen Klassen, weil bei zwei Scannern des Herstellers Canon, die bei der Entwicklung zur Verfügung standen, Fehler innerhalb der Klasse Scanner auftraten. Diese konnten jeweils durch eine kleine Modifikation behoben werden und sind im Quelltext der Klasse mit "Modified" gekennzeichnet. Zudem stellte sich der Umfang der ausgegebenen Fehlermeldungen im Ereignis *Error* als Problem heraus, das durch eine weitere Ergänzung gelöst werden konnte. Entsprechende Nachbesserungen wurden dem Entwickler bereits vorgeschlagen.
- Zum Drehen, Löschen oder Verschieben gescannter Dokumente dürfen einzelne oder mehrere Einträge mit der Maus markiert werden. Die Liste verwendet hierfür die Multi-Select-Option. Für alle anderen Operationen wie EMailen, Drucken oder Abspeichern werden grundsätzlich alle Einträge verarbeitet.
- Beim EMailen wird die erzeugte PDF-Datei mit einem sekundengenauen Zeitstempel versehen.
- Beim Abspeichern mehrerer Scans im JPG-, PNG- oder BMP-Format werden die einzelnen Scans gemäß ihrer Reihenfolge mit einem zusätzlichen dreistelligen Index versehen – wie zum Beispiel Scan_001.jpg, Scan_002.jpg usw. .
- Mit einem Doppelklick auf einen Scan in der Liste, wird dieser im großen Scan-Display angezeigt.
- Ein aufrufbarer Dialog stellt einen Helligkeits- und Kontrast-Steller für die Nachbehandlung von Scans zur Verfügung. Diese Option hat sich bei den verwendeten *Canon-Scannern* als nützlich erwiesen.
- Die optionale und experimentelle OCR-Funktion beschränkt sich auf die Erstellung durchsuchbarer PDFs und kann innerhalb der App nachinstalliert werden.
- Das mehrseitige Scannen mit Geräten, die über einen Einzugsschacht verfügen, wurde nicht umgesetzt, weil ein derartiger Scanner-Typ bei der Entwicklung nicht zur Verfügung stand.
- Sollte die eine oder andere Geräteoptionen nicht wie erwartet funktionieren, so ist die Ursache hierfür eher im Bereich des SANE-Backends zu suchen. Hierfür stellten sich aber leider SANE-Kompatibilitätslisten in zwei Einzelfällen als unzuverlässige Referenz heraus.