24.9.3 DBusApplication

Die Klasse *DBusApplication* (gb.dbus) repräsentiert eine Anwendung, die am D-Bus registriert ist. Diese Klasse können Sie erzeugen. Sie verhält sich wie Array, das Sie nur lesen können.

24.9.3.1 Eigenschaften

Die Klasse *DBusApplication* verfügt über die folgenden Eigenschaften:

Eigenschaft	Datentyp	Beschreibung
Connection	DBusConnection	Objekt vom Typ *DBusConnection*, dessen Anwendung mit dem D-Bus verbunden ist.
Index	Integer	Index im (internen) Anwendungscache.
Name	String	Name der Anwendung auf dem D-Bus und entspricht DBus.Name.

Tabelle 24.9.3.1.1 : Eigenschaften der Klasse DBusApplication

24.9.3.2 Methoden

Die Klasse *DBusApplication* besitzt diese drei Methoden:

Methode	Beschreibung
Register (Object As DBusObject, Path As String [, Interfaces As String[]])	Registriert ein D-Bus-Object am D-Bus – seit Gambas 3.9. Für die Parameter gilt: Object ist das zu registrierende D-Bus-Objekt, Path ist der D-Bus-Pfad für das D-Bus-Objekt und Interface (optional) ermöglicht die Angabe der Namen von zusätzlichen Schnittstellen, wenn diese für das verwendete D-Bus-Objekt existieren.
Unregister (Object As DBusObject)	Meldet das im Parameter genannte D-Bus-Objekt vom D-Bus ab.
Raise (Object As DBusObject, Signal As String [, Arguments As Variant[]])	Löst ein D-Bus-Signal aus. Für die Parameter gilt: Object ist das D-Bus-Objekt, welches das Signal sendet, Signal ist der Name des gesendeten Signals und Arguments (optional) repräsentiert die Argumente des Signals vom Daten-Typ Variant-Array.

Tabelle 24.9.3.2.1 : Methoden der Klasse DBusApplication

24.9.3.3 Beispiel 1 – New DBusApplication

Mit *New DBusApplication(Parameterliste)* können Sie ein DBusApplication-Objekt erzeugen:

```
Dim hDBusApplication As DBusApplication
hDBusApplication = New DBusApplication ( Connection As DBusConnection, ApplicationName As String )
```

Das erzeugte Objekt repräsentiert eine Anwendung, die mit dem Session-Bus oder dem System-Bus verbunden ist. Für die beiden Parameter gilt:

- Connection ist ein D-Bus-Connection-Objekt (DBus.Session oder DBus.System)
- ApplicationName ist der Name der Anwendung auf dem D-Bus.

In gleicher Weise können Sie alternativ auch DBus[ApplicationName] einsetzen.

24.9.3.4 Beispiel 2 – DBusApplication[ ]

Die Klasse *DBusApplication* verhält sich wie ein Nur-Lesen-Array. Diesen Sachverhalt nutzen die beiden folgenden Quelltexte – der erste nur formal:

```
Dim hDBusApplication As DBusApplication
Dim hDBusProxy As DBusProxy
hDBusProxy = hDBusApplication [ ObjectPath As String [ , Interface As String ] ]
```

Zurückgegeben wird mit *hDBusProxy* ein Proxy zu einem (existierenden) D-Bus-Objekt, das im ersten Parameter angegeben wird.

- ObjectPath ist der D-Bus-Pfad zum Objekt.
- Interface ist das Interface des Objektes, das Sie nutzen wollen.

Ist das Argument für den optionalen Parameter *Interface* nicht angegeben, so haben Sie Zugriff auf jede Methode und alle Eigenschaften des (exportierten) Objekts. Sonst haben Sie nur Zugriff auf Methoden und Eigenschaften des angegebenen Interfaces – wie im folgenden Beispiel:

```
  Dim hDBusApplication As DBusApplication
  Dim hDBusProxy As DBusProxy
  Dim sDBusName, sDBusObjectPath, sDBusInterface As String
  Dim hConnection As DBusConnection

  hConnection =  DBus.Session
  sDBusName = "org.freedesktop.Notifications"
  sDBusObjectPath = "/org/freedesktop/Notifications"
  sDBusInterface = "org.freedesktop.Notifications"

  hDBusApplication = New DBusApplication(hConnection, sDBusName)
  hDBusProxy = New DBusProxy(hDBusApplication, sDBusObjectPath, sDBusInterface)
```

Im Sinne einer guten Daten-Kapselung ist die Erzeugung eines Interfaces auf der Seite der Gambas-Anwendung, die das Objekt zum D-Bus exportiert, stets von Vorteil.

Im Kapitel 24.9.7.0 wird Ihnen das Projekt DBusNMChildren vorgestellt, dass auch einen D-Bus-Proxy einsetzt:

```
  sDBusName = "org.freedesktop.NetworkManager"
  hConnection = DBus.System
  hDBusApplication = New DBusApplication(hConnection, sDBusName)
  sDBusObjectPath = "/org/freedesktop/NetworkManager"
  sDBusInterface = "org.freedesktop.NetworkManager"
  hDBusProxy = New DBusProxy(hDBusApplication, sDBusObjectPath, sDBusInterface)
' hDBusProxy = DBus["system://" & sDBusName][sDBusObjectPath, sDBusInterface] ' <<---  Alternative
```

24.9.3.5 Beispiel 3

Sie können die Klasse DBusApplication.Connection.Register(…) genau dann einsetzen, wenn Sie vorher eine Verbindung einer Anwendung zum D-Bus (Session) mit Hilfe der Klasse *DBusApplication* erzeugt haben:

```
Public hDBApplication As DBusApplication
Public hDBObject As DBusObject

Public Sub _new()
  hDBApplication = New DBusApplication(DBus.Session, Application.Path)
  hDBObject = New DBusObject

  DBus.Unique = True
  If Not hDBApplication.Connection.Applications.Exist(DBus.Name) Then
     hDBApplication.Connection.Register(hDBObject, Application.Path)
  Else
    Message.Error("Es existiert bereits eine Instanz von '" & Application.Name & "' !")
    Quit
  Endif

End

...

Public Sub Form_Close()
  If DBus.IsRegistered(hDBObject) Then hDBApplication.Connection.Unregister(hDBObject)
  FMain.Close()
End
```