

# cyProvider®

## HOW-TO-GUIDE

---

© cyFlex GmbH  
Maastrichter Straße 24 • DE-41464 Neuss  
Telefon: +49 2131 980501 • Telefax: +49 2131 980502  
E-Mail: [info@cyflex.de](mailto:info@cyflex.de)

---

# Inhaltsverzeichnis

<a href="#">1.Einleitung</a>	<a href="#">4</a>
<a href="#">2.Begriffsdefinitionen</a>	<a href="#">6</a>
<a href="#">3.Das Template</a>	<a href="#">10</a>
<a href="#">3.1.Vier kleine Schritte zur ersten Anwendung</a>	<a href="#">10</a>
<a href="#">3.1.1.Kopieren des Templates ins cyProvider-Sources-Verzeichnis</a>	<a href="#">12</a>
<a href="#">3.1.2.Anpassen der Datei cyGuiApl.ini</a>	<a href="#">14</a>
<a href="#">3.1.3.Datei cyProvider.ini pflegen</a>	<a href="#">16</a>
<a href="#">3.1.4.Datei myApplication.ini erstellen</a>	<a href="#">17</a>
<a href="#">3.1.5.Informationen zur Anwendung hinterlegen</a>	<a href="#">18</a>
<a href="#">3.1.6.Anwendung „myApplication“ mit dem cyAgent präsentieren</a>	<a href="#">18</a>
<a href="#">4.Definition des Anwendungsmenüs</a>	<a href="#">21</a>
<a href="#">5.Selektionsautomaten</a>	<a href="#">23</a>
<a href="#">5.1.Standard-Selektionsautomaten</a>	<a href="#">23</a>
<a href="#">5.1.1.Die Templatedatei Def-Files\Tpl_Frm_SAM.cdd</a>	<a href="#">23</a>
<a href="#">5.1.2.Die Templatedatei Roles\Admin\Tpl_Frm_SAM.cdd</a>	<a href="#">24</a>
<a href="#">5.2.Selektionsautomaten mit Master-Detail-Beziehung</a>	<a href="#">29</a>
<a href="#">5.2.1.Die Templatedatei Def-Files\Tpl_Frm_SAM_Master_Detail.cdd</a>	<a href="#">29</a>
<a href="#">5.2.2.Die Templatedatei Roles\Admin\Tpl_Frm_SAM_Master_Detail.cdd</a>	<a href="#">30</a>
<a href="#">5.3.Selektionsautomaten und Container</a>	<a href="#">34</a>
<a href="#">5.3.1.Die Templatedatei Def-Files\Tpl_Frm_SAM_Container.cdd</a>	<a href="#">36</a>
<a href="#">5.3.2.Die Templatedatei Roles\Admin\Tpl_Frm_SAM_Container.cdd</a>	<a href="#">37</a>
<a href="#">5.4.Selektionsautomaten und FastReport®</a>	<a href="#">42</a>
<a href="#">5.5.Selektionsautomaten und SpecialBands</a>	<a href="#">48</a>
<a href="#">5.6.Referenz-Selektionsautomaten</a>	<a href="#">49</a>
<a href="#">6.Pflegeautomaten</a>	<a href="#">52</a>
<a href="#">6.1.Standard-Pflegeautomaten</a>	<a href="#">52</a>
<a href="#">6.1.1.Die Templatedatei Def-Files\Tpl_Frm_PAM.cdd</a>	<a href="#">53</a>
<a href="#">6.1.2.Die Templatedatei Roles\Admin\Tpl_Frm_PAM.cdd</a>	<a href="#">54</a>
<a href="#">6.2.Pflegeautomaten und Container</a>	<a href="#">63</a>

---

---

<a href="#">6.2.1.Die Templatedatei Def-Files\Tpl_Frm_PAM_Container.cdd</a>	<a href="#">64</a>
<a href="#">6.2.2.Die Templatedatei Roles\Admin\Tpl_Frm_PAM_Container.cdd</a>	<a href="#">67</a>
<a href="#">6.3.Pflegeautomaten und Clones</a>	<a href="#">69</a>
<a href="#">6.4.Weitere Parameter für Pflegeautomaten</a>	<a href="#">71</a>
<a href="#">6.4.1.InsertOnEmpty</a>	<a href="#">71</a>
<a href="#">6.4.2.Die Formate fRichMemo und fListView</a>	<a href="#">71</a>
<a href="#">6.4.3.Parameter für den Email-Versand</a>	<a href="#">71</a>
<a href="#">6.4.4.Modifyconflict-Prüfung einschränken</a>	<a href="#">73</a>
<a href="#">6.4.5.Lokale Dokumente direkt aus dem PAM öffnen</a>	<a href="#">73</a>
<a href="#">7.Zusätzliche Features</a>	<a href="#">74</a>
<a href="#">7.1.Clickvalue und Clickmap</a>	<a href="#">74</a>
<a href="#">7.2.Drag &amp; Drop</a>	<a href="#">75</a>
<a href="#">7.3.Parameter</a>	<a href="#">80</a>
<a href="#">7.4.Storefile</a>	<a href="#">81</a>
<a href="#">8.Sprachdateien</a>	<a href="#">82</a>
<a href="#">8.1.Anwendungsübergreifende Sprachdateien</a>	<a href="#">82</a>
<a href="#">8.2.Anwendungsabhängige Sprachdateien</a>	<a href="#">84</a>
<a href="#">9.Styles</a>	<a href="#">88</a>
<a href="#">10.Design</a>	<a href="#">90</a>
<a href="#">11.Zugriff auf SAP® Systeme</a>	<a href="#">91</a>

## 1. Einleitung

Mit cySystem, bestehend aus den beiden Hauptkomponenten cyProvider als Backend und cyAgent als Frontend, lassen sich datenbankbasierende Windows-Anwendungen entwickeln und betreiben. Dabei agiert der cyAgent als Applikations-Browser, der über das Internet/Intranet Anwendungen und Daten vom cyProvider anfordert und auf dem Client-PC zur Verfügung stellt.

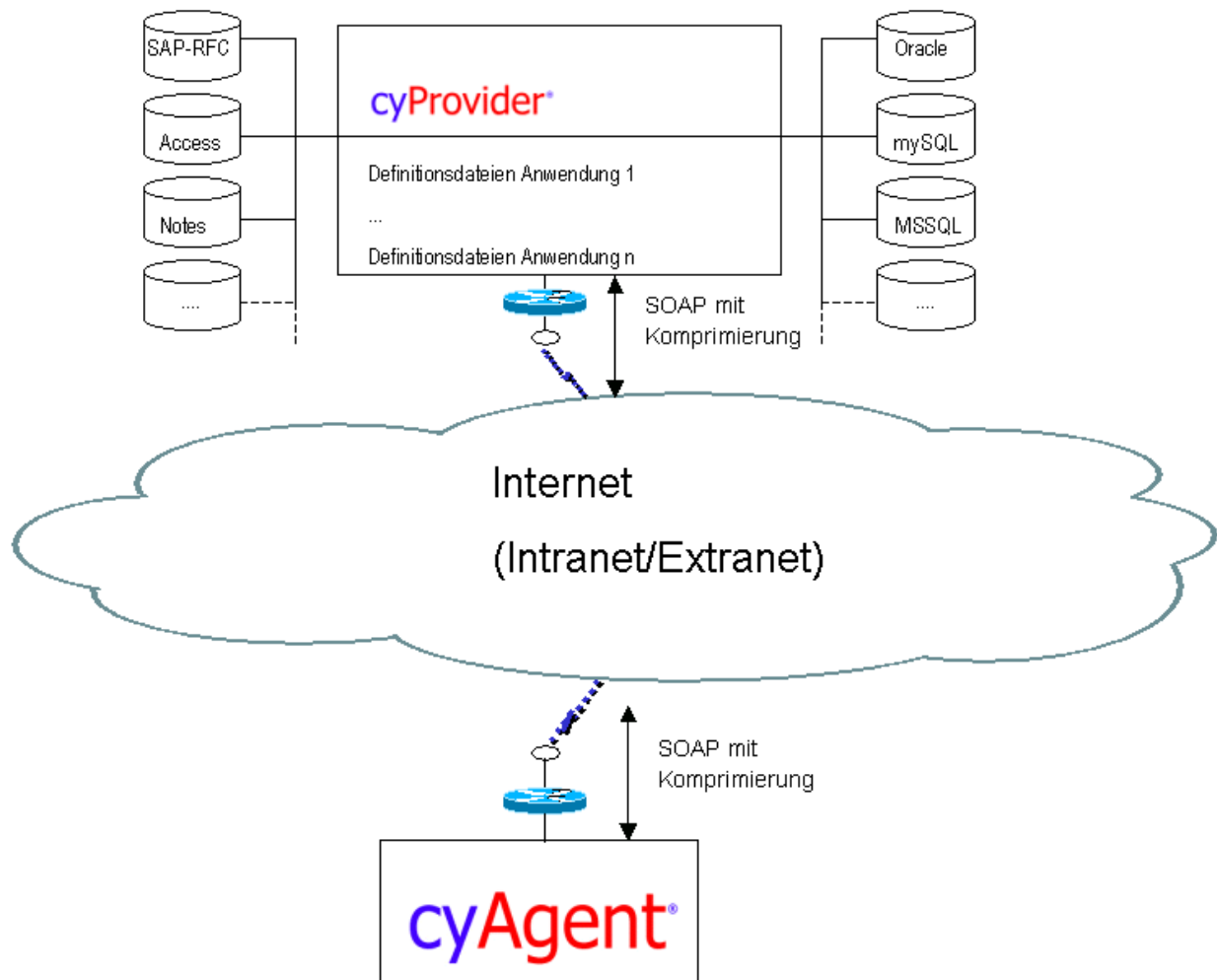


Abb. 1

Ein wesentliches Merkmal einer cySystem-Anwendung liegt darin, dass sie nicht aus ausführbarem Code, sondern aus Definitionen, abgelegt in Textdateien, besteht. Diese Definitionen beschreiben die Eigenschaften einer Anwendung und werden erst auf dem Client-PC vom cyAgent interpretiert und ausgeführt.

---

Neben den sicherheitsrelevanten Vorteilen gegenüber der herkömmlichen Internettechnologie, hat dieses Verfahren auch aus entwicklungstechnischer Sicht seine Vorzüge. Dadurch, dass wir Anwendungen definieren und nicht programmieren, sind keinerlei Programmierkenntnisse erforderlich. Alles, was wir benötigen, ist ein Text-Editor und einige SQL-Kenntnisse, der Datenbankabfragesprache für relationale Datenbanken. Zudem brauchen wir uns nur um die anwendungsspezifischen Inhalte zu kümmern, da bereits das Basissystem eine Fülle an Funktionalität bereitstellt. Letztendlich steht noch ein Template zur Verfügung, um die Entwicklung von cySystem-Anwendungen zu unterstützen. Gegenüber der Programmierung ermöglicht diese Vorgehensweise einen erheblich geringeren Entwicklungsaufwand und liefert sehr schnell stabile Anwendungen, da Fehler in einer Definition in der Regel schon beim ersten Test sichtbar werden, was für die Programmierung leider nicht immer zutrifft. Hinsichtlich der üblichen Vorgehensweise in Softwareentwicklungs-Projekten (Pflichtenheft) ergibt sich durch den geringen Entwicklungsaufwand die Möglichkeit, dem Anwender bereits mit Projektbeginn erste Module der gewünschten Anwendung zu präsentieren und damit Probleme bzw. Missverständnisse rechtzeitig zu vermeiden und den Anwender schrittweise bei der Realisierung seiner Anwendung mitzunehmen.

Für den Fall, dass die Komplexität einer Anwendung dennoch die Programmierung einiger Programmerroutinen erforderlich macht, gibt es den cyScripter. Mit Hilfe des cyScripter können Scripte in PascalScript, JScript, BasicScript oder C++Script erstellt werden, die uns die Möglichkeit eröffnet, mit der cySystem-Anwendung zu interagieren, auf Events zu reagieren und individuelle Funktionalität (Business Rules) zu integrieren. Der cyScripter wird Bestandteil der Entwicklungsumgebung cyCreator sein, die wahrscheinlich Ende 2007 zur Verfügung stehen wird. Auf Anfrage kann der cyScripter auch vorab bereitgestellt werden.

Welche Definitionsdateien mit welchen Definitionen für eine Anwendung erforderlich sind, werden wir in den folgenden Kapiteln anhand eines Beispiels erarbeiten. Mit Hilfe des Templates werden wir eine neue Anwendung anlegen und diese schrittweise mit Beispielen zu den Möglichkeiten, die cySystem bietet, erweitern.

Voraussetzung für die Entwicklung ist selbstverständlich ein zuvor installierter cyProvider, wie im Administrations- und Installationsleitfaden (siehe [www.cyflex.de](http://www.cyflex.de)) beschrieben, sowie ein cyAgent zum Testen der neuen Anwendung.

## 2. Begriffsdefinitionen

Bevor wir nun unsere erste Anwendung erstellen, einige Anmerkungen zur Namensgebung und der Bedeutung der verwendeten Begriffe:

Eine cySystem-Anwendung besteht aus verschiedenen Formularen, die auf der Anwenderseite vom cyAgent präsentiert werden. Wir unterscheiden die folgenden vier Formulartypen:

- Anwendungsfenster (MDI-Form)
- Selektionsautomat (SAM)
- Pflegeautomat (PAM)
- Referenz-Selektionsautomat

Das Anwendungsfenster (Abb. 2) ist das übergeordnete Formular, in dem die übrigen Formulartypen sowie das Hauptmenü und seine Untermenüs dargestellt werden.

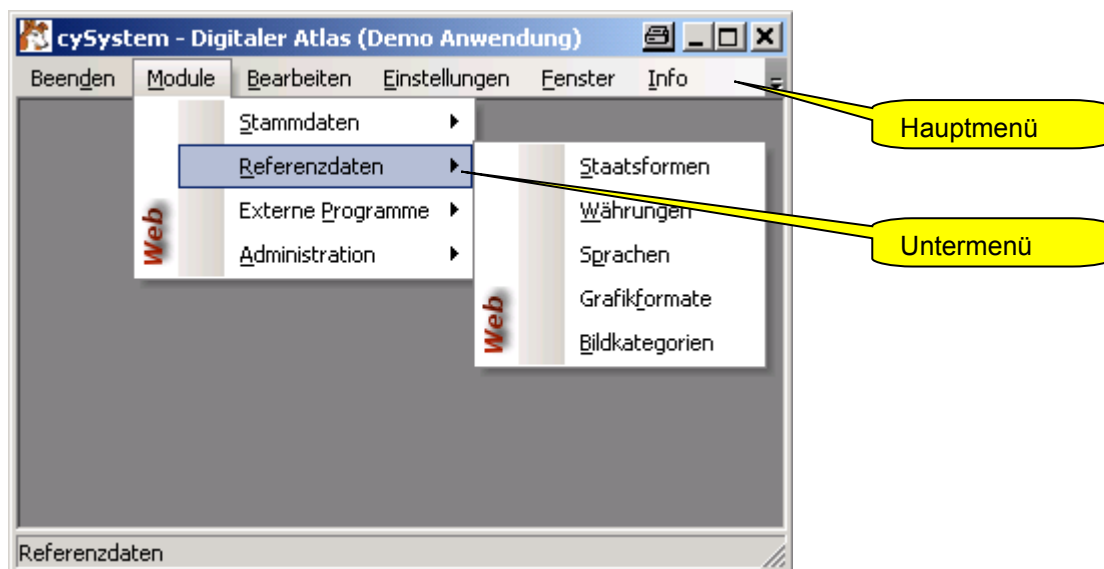


Abb. 2

Ein **Selektionsautomat** präsentiert in einem Grid (Tabelle) die Daten einer (Datenbank-) View, die wiederum über eine hinterlegte SQL-Anweisung definiert wird. Oder anders gesagt, die View stellt dem Grid eine vordefinierte Sicht auf die Daten der Datenbank zur Verfügung. In jedem Grid stehen umfangreiche Funktionen (Sortieren, Filtern, Gruppieren, ...) zur Auswertung der Daten zur Verfügung. Aus einem Selektionsautomaten heraus können neben weiteren Selektionsautomaten auch Pflegeautomaten aufgerufen werden. Standardmäßig werden Selektionsautomaten zur Anzeige und Auswertung von Daten genutzt. Es besteht aber auch die Möglichkeit mit Hilfe von Drag & Drop Daten zu verändern (siehe Kapitel 7.2).

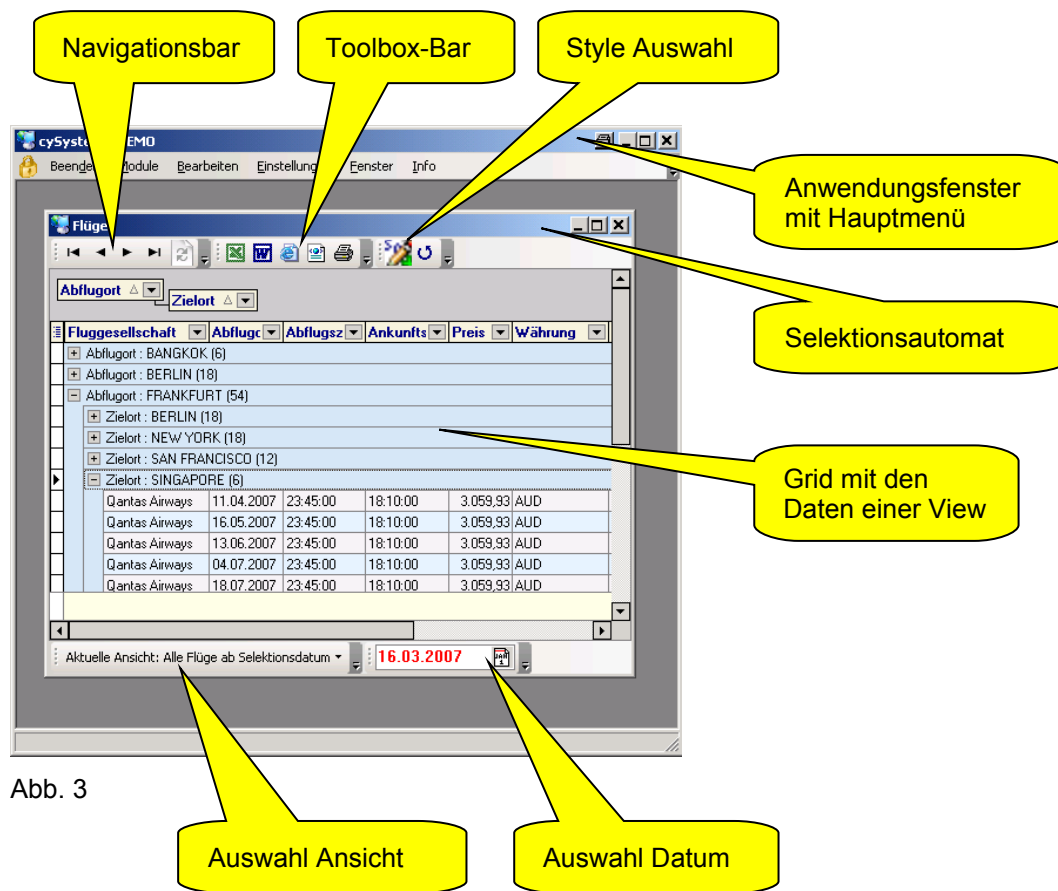


Abb. 3

Ein **Pflegeautomat** beinhaltet die Daten eines einzelnen Datensatzes mit der Möglichkeit diesen zu ändern, zu löschen oder einen neuen Satz hinzuzufügen. Ein Pflegeautomat kann **nur** aus einem Selektionsautomaten, in dem der Datensatz zuvor selektiert wurde, heraus aufgerufen werden. Zudem kann sich ein Pflegeautomat aus mehreren Pflegeautomaten zusammensetzen, wie die folgende Abbildung zeigt. Hier verbergen sich hinter jedem Tabreiter weitere Pflegeautomaten. Zur Auswahl von Referenzdaten (Schlüsselfelder, deren Werte in eigenen Tabellen gepflegt werden) können aus Pflegeautomaten Referenz-Selektionsautomaten aufgerufen werden.

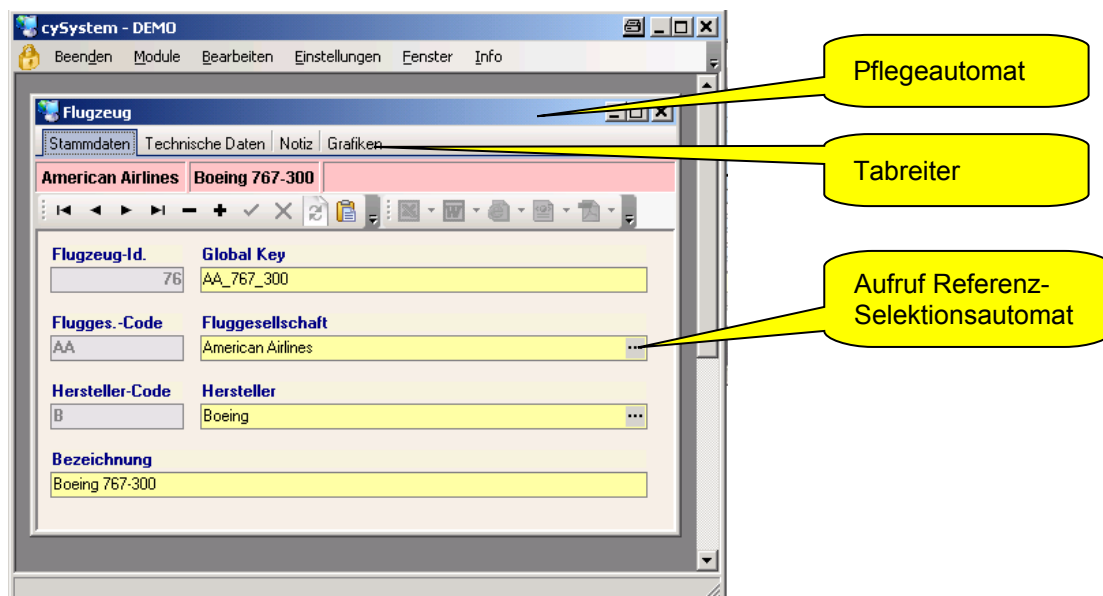


Abb. 4

**Referenz-Selektionsautomaten** zur Auswahl von Daten aus Referenztabelle öffnen sich in einem modalen Fenster und unterscheiden sich von einem Selektionsautomaten nur durch den zusätzlichen OK-Button und den Abbrechen-Button zur Bestätigung der getroffenen Auswahl.

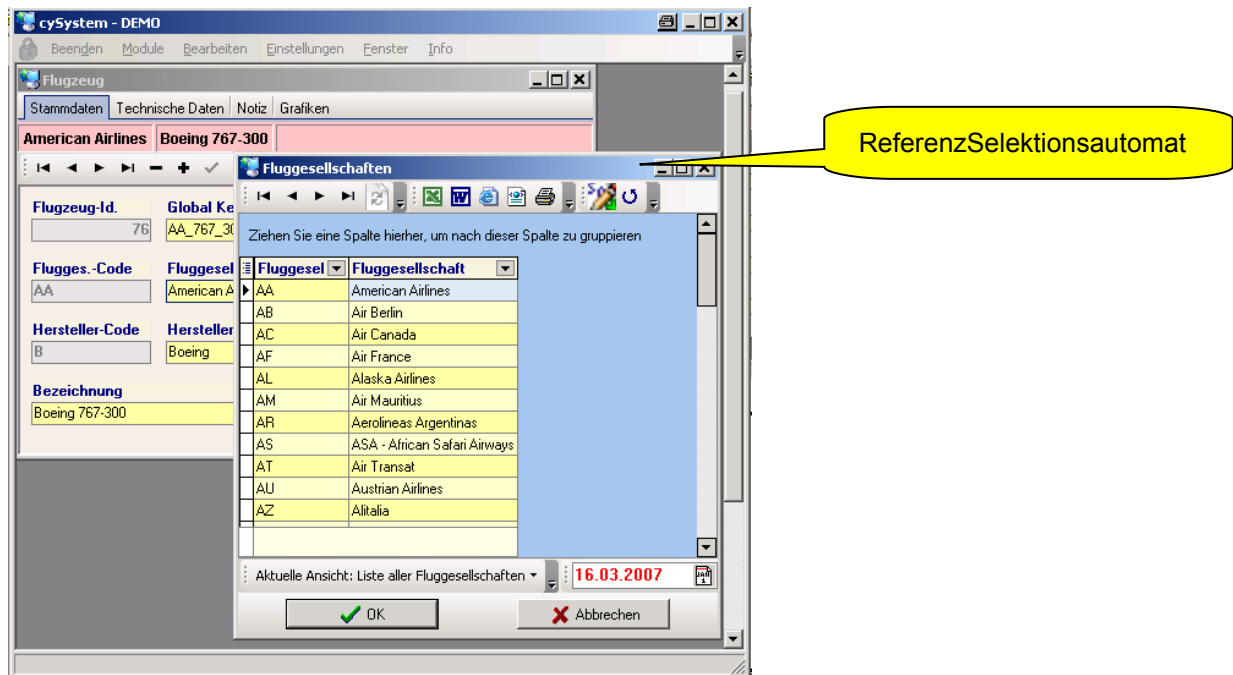


Abb. 5

### 3. Das Template

Mit der Installation von cyProvider wurde im cyProvider-Verzeichnis ein Template abgelegt, das als Basis für neue, eigene Anwendungen verwendet werden kann. Dieses Template enthält bereits die für eine Anwendung erforderliche Verzeichnisstruktur, sowie eine Vielzahl von generell benötigten Definitionsdateien.

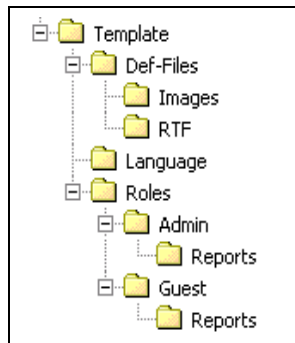


Abb. 6

Die Definitionsdateien, die im Template für die unterschiedlichen Formulartypen als Kopiervorlage zur Verfügung stehen, sind durch die folgenden Präfixe gekennzeichnet:

- Tpl\_Frm\_SAM\_      Formulare für Selektionsautomaten (SAM)
- Tpl\_Frm\_PAM\_      Formulare für Pflegeautomaten (PAM)
- Tpl\_Ref\_SAM\_      Formulare für Referenz-Selektionsautomaten

Obwohl Namen für Formulare frei vergeben werden können, empfiehlt es sich, unterschiedliche Formulartypen mit Hilfe von Namenszusätzen zu klassifizieren.

Bevor nun die vielfältigen Möglichkeiten, eigene Anwendungen mit Hilfe des Templates zu definieren, im Detail beschrieben werden, wollen wir mit einigen wenigen Handgriffen unsere erste kleine Anwendung fertig stellen.

#### 3.1. Vier kleine Schritte zur ersten Anwendung

Die Anwendung „myApplication“ soll aus einem Anwendungsfenster, einer kompletten Menüstruktur, einem Formular zur Anzeige der Benutzerdaten und der Möglichkeit, zwischen deutscher oder englischer Sprache auszuwählen, bestehen. Als Datenbank dient die auf dem cyProvider bereits vorhandene *DemoDB.mdb*, deren Tabelle *USR* auch die Login-Informationen der Benutzer liefert. Voraussetzung für den Datenbankzugriff ist die installierte Microsoft Jet Engine. Die aktuelle Jet Engine kann über das Jet Engine Servicepack 8 installiert werden.

---

Nun zu unseren vier kleinen Schritten:

Das Template kopieren wir ins cyProvider\Sources-Verzeichnis und benennen es in myApplication um.

In der *cyGuiApl.ini* des myApplication\Def-Files-Verzeichnisses ersetzen wir myProvider (dreimal) durch DemoDB und myhost.mydomain (zweimal) durch die URL des Rechners, auf dem der cyProvider läuft.

Die Datei *myApplication.ini* verschieben wir aus dem myApplication-Verzeichnis in das Verzeichnis cyProvider\Sources\Browser\Def-Files und ersetzen in der ini-Datei erneut myhost.mydomain durch die URL des Rechners, auf dem der cyProvider läuft.

Abschließend verschieben wir die Datei *DE\_myApplication.RTF* aus dem myApplication-Verzeichnis in das Verzeichnis cyProvider\Sources\Browser\Def-Files\RTF.

Fertig!

Zur Kontrolle starten wir noch den cyAgent, geben die URL unseres cyProviders ein, klicken myApplication an und sollten nach der erfolgreichen Anmeldung mit Admin/Admin folgendes Anwendungsfenster sehen:

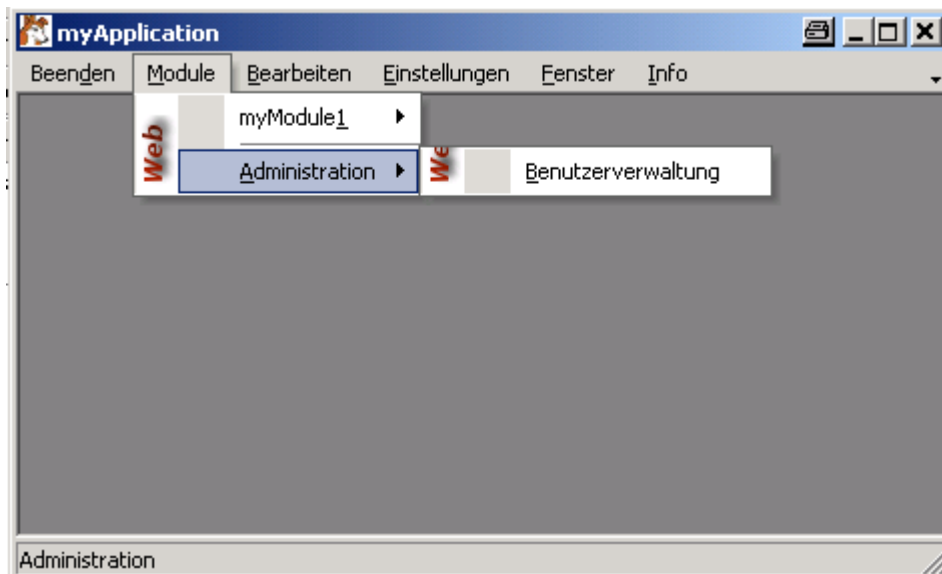


Abb. 7

Wem das nun zu schnell ging und wer ein wenig mehr zu den kopierten Dateien und den darin enthaltenen Definitionen und Parametern erfahren möchte, erhält in den folgenden Kapiteln die gewünschten Informationen.

---

### 3.1.1. Kopieren des Templates ins cyProvider-Sources-Verzeichnis

Bei dieser Kopieraktion ist nur zu beachten, dass das kopierte Verzeichnis Template tatsächlich in myApplication umbenannt wird.

Soll ein anderer Name vergeben werden, d.h. das kopierte Verzeichnis heißt AndererName, müssen noch folgende Umbenennungen vorgenommen werden:

Die Datei *myApplication.cdd* im Verzeichnis \AndererName\Def-Files muss selbst in *AndererName.cdd* umbenannt werden. Innerhalb der Datei muss die Zeichenfolge myApplication durch AndererName (ein- bzw. zweimal) ersetzt werden.

Die Datei *myApplication.cdd* im Verzeichnis \AndererName\Roles\Admin muss selbst in *AndererName.cdd* umbenannt werden. Innerhalb der Datei muss die Zeichenfolge myApplication durch AndererName (einmal) ersetzt werden.

Die Datei *myApplication.cdd* im Verzeichnis \AndererName\Roles\Guest muss selbst in *AndererName.cdd* umbenannt werden. Innerhalb der Datei muss die Zeichenfolge myApplication durch AndererName (einmal) ersetzt werden.

Innerhalb der im Verzeichnis \AndererName\Language stehenden Sprachdateien muss die Zeichenfolge myApplication durch AndererName (viermal) sinnvoll ersetzt werden.

Durch das Kopieren des Templates wurde die für jede Anwendung notwendige Verzeichnisstruktur erzeugt (vgl. Abb. 4), deren Bedeutung bzw. deren Ordner und Inhalte nun kurz beschrieben werden:

- Def-Files

Dieses Verzeichnis enthält die rollenunabhängigen Definitionen der Anwendung. Folgende Dateien sind grundsätzlich vorhanden:

- *cyGuiApl.ini*, in der die für die Anwendung grundlegenden Parameter enthalten sind (definierte Sprachen, benötigte Provider).
- *myApplication.cdd*, mit der Definition des Hauptmenüs. Der Name dieser Datei muss mit dem Verzeichnisnamen, der gleichzeitig dem Namen der Anwendung entspricht, übereinstimmen.
- *Frm\_SAM\_User.cdd* enthält bereits eine vollständige Definition eines Formulars zur Benutzerverwaltung.
- *Tpl\_Frm\*.cdd* bzw. *Tpl\_Ref\*.cdd* sind Templates für verschiedene Formulartypen und dienen als Kopiervorlage. Die Beschreibungen der einzelnen Formulare bezüglich Größe, Struktur (horizontal oder vertikal

---

unterteilt, mit oder ohne Tabreiter) und der enthaltenen Elemente (Links auf Selektions- oder Pflegeautomaten) liegen je Formular in einer separaten Datei vom Typ *Formularname.cdd*. Der Formularname ist frei zu vergeben. Die übrigen Formulare in diesem Verzeichnis dienen als Template für bestimmte Formulartypen und sind durch den Vorsatz Tpl\_ im Namen gekennzeichnet.

- *\_CONNECT.cdd* enthält die Definitionen für Datenbankänderungen beim Einsatz von Drag & Drop.
- *\_STYLES.cdd* enthält die vordefinierten Layout-Definitionen, die dem Anwender zur Auswahl zur Verfügung gestellt werden.
- *\_DESIGN.cdd*, mit den Zuordnungen von Styles zu bestimmten Formular-Elementen in Selektions- und Pflegeautomaten, der Definition von Critical-Colors für die Kennzeichnung von Muss- oder nicht eingabebereiten Feldern, sowie den Style-Conditions. Mit Hilfe von Style-Conditions kann das Layout von Feldern in Abhängigkeit von deren Inhalt dynamisch verändert werden.
- *\_PROGRAMS.cdd* enthält Definitionen für die Einbindung von externen Programmen.

Zusätzlich besitzt Def-Files noch die beiden Unterverzeichnisse Images und RTF. In Images abgelegt sind Bitmaps für die Sprachen (*Sprach-ID.bmp*), die als Icon im Menüpunkt Sprachauswahl dienen und ein Bitmap Barglyph, das in der ersten Spalte der Untermenüs angezeigt wird. RTF enthält Dokumente im RTF-Format, die im Menüpunkt Info angezeigt werden sollen.

- *Language*

Das Language-Verzeichnis enthält pro definierte Sprache eine Datei *<Sprach-ID>.cdd* mit allen sprachabhängigen Bezeichnungen der Anwendung (Überschriften, Feldnamen, Formularnamen, etc.). Im Template befinden sich die Sprachdateien *de.cdd* und *en.cdd* für die Pflege der deutschen und englischen Texte.

- *Roles*

Im Verzeichnis Roles befindet sich für jede definierte Rolle ein Unterverzeichnis mit den rollenspezifischen Definitionen. Für jedes im Verzeichnis Def-Files zuvor definierte Formular liegt im Roles-Verzeichnis eine namensgleiche cdd-Datei, in der die rollenabhängigen Inhalte mit Hilfe von SQL-Anweisungen definiert werden.

In der Datei *myApplication.cdd* wird das rollenspezifische Untermenü definiert.

---

Wurden mit dem FastReport® Designer erstellte Reports der Rolle zugeordnet, so befinden sich die Reportdefinitionen als XML-Datei im Unterverzeichnis Reports.

Für den anonymen Benutzer gibt es das Rollenverzeichnis mit dem reservierten Namen Guest. Damit erhalten alle Benutzer, die die Anwendung mit AutoLogon=True (siehe *myApplication.ini*) starten, ohne Authentifizierung die Rolle Guest zugeordnet. Guest enthält ebenfalls eine Datei *myApplication.cdd* mit dem rollenspezifischen Menü sowie den rollenspezifischen Formulardateien.

### 3.1.2. Anpassen der Datei *cyGuiApl.ini*

In der Datei *cyGuiApl.ini* sind die für die Anwendung grundlegenden Parameter enthalten:

- die unterstützten Sprachen,
- die von der Anwendung benötigten Provider und die zugehörigen Adressen (URLs),
- definierte Placeholder (cySystem-Variablen, auf die vom cyAgent, vom cyProvider oder von Skripten aus zugegriffen werden kann oder die in SQL-Anweisungen in der Form %<VariablenName>% einfließen können)

Die Definition der unterstützten Sprachen erfolgt durch die Angabe der Sprach-ID und der zugehörigen LCID (= locale identifiers). Zusätzlich ist neben der Sprachbezeichnung die Datenbankfeldnamen-Erweiterung anzugeben ( Z. B. de=1031; 0; deutsch, wobei die Ziffer 0 bedeutet, dass Datenbankfelder mit deutschen Inhalten eine Bezeichnung der Form <Feldname\_0> haben. Auf diese Ziffer kann in den Definitionen über den Placeholder %Languageld% zugegriffen werden.).

Provider (z. B. Microsoft ADO) sind auf die jeweilige Datenquelle zugeschnittene Komponenten, die dem cyProvider die gewünschten Daten liefern, der diese wiederum unserer Anwendung zur Verfügung stellt.

Für unser Beispiel ersetzen wir in der *cyGuiApl.ini* myProvider durch den bereits installierten DemoDB-Provider und beide Provider-Adressen erhalten die URL des Rechners, auf dem der cyProvider läuft.

Werden zusätzliche Provider in der Anwendung benötigt, ist die Providerliste durch senkrechten Strich und den Namen des neuen Providers zu erweitern und eine zusätzliche Provider-Sektion unter diesem Namen aufzunehmen.

Der Name der Datei *cyGuiApl.ini* kann per Definition in der Datei *myApplication.ini* geändert werden (siehe Kapitel 3.1.4). Ebenso können mehrere dieser ini-Dateien angelegt werden, die wiederum über unterschiedliche *myApplication.ini* Dateien angesprochen werden können.

Die Sektionen der *cyGuiApl.ini* für unsere Beispielanwendung sehen damit wie folgt aus:

**[Language]**

```
de=1031;0;deutsch;9905  
en=2057;1;english;9906
```

**[Application Initialization]**

```
Providers=DemoDB  
TraceLog=c:\temp\cyGuilog.txt
```

**[DemoDB]**

```
ConnectionType=SOAP  
URL=https://myhost.mydomain:443/soap/IMDCDataModule  
ExportedProvider=DemoDB
```

**[Placeholder]**

```
;ShowCDDInfo=True
```

Der Parameter `TraceLog` ermöglicht das Schreiben einer lokalen Logdatei, in der alle ausgeführten SQL-Anweisungen und der aktuellen Parameterwerte (siehe Kapitel 7.3) enthalten sind.

Der Placeholder `ShowCDDInfo` ist durch das vorangestellte Semikolon nicht aktiv. In der Entwicklungsphase von Anwendungen lassen sich mit seiner Hilfe beim Testen nützliche Informationen bezüglich der jeweiligen Definitionsdatei anzeigen. Die folgende Abbildung zeigt die Informationen, die erscheinen, wenn man mit dem Cursor über das Feld `Städte` fährt:

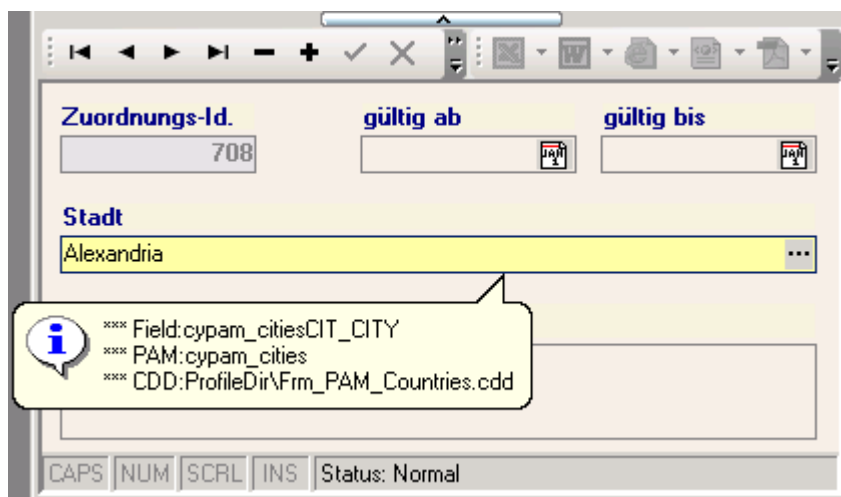


Abb. 8

---

### 3.1.3. Datei *cyProvider.ini* pflegen

Da unser Beispiel den bereits installierten Provider DemoDB nutzt, wurde dieser Schritt oben nicht erwähnt.

Für den Fall, dass ein anderer Name oder ein zusätzlicher Provider als ExportedProvider in der *cyGuiApl.ini* definiert wurde, müssen wir die *cyProvider.ini* wie folgt anpassen:

- In der MDC-Sektion erweitern wir die Providerliste um den ExportedProvider (z. B. myProvider).
- Dann fügen wir eine neue MDC.myProvider-Sektion ein, in der der Provider- Typ und der Connection-String definiert wird:

**[MDC]**

```
Providers=DemoDB | myProvider
```

**[MDC.DemoDB]**

```
ProviderType=ptADO  
ConnectionString=Provider=Microsoft.Jet.OLEDB.4.0;Data  
Source=C:\Programme\cyProvider\Data\DemoDB.mdb;Persist  
Security Info=False
```

**[MDC.myProvider]**

```
ProviderType=ptADO  
ConnectionString=Provider=Microsoft.Jet.OLEDB.4.0;Data  
Source=c:\Programme\cyProvider\Data\Demo.xls;Extended  
Properties=Excel 8.0;Persist Security
```

Die Standard-Installation von cyProvider nutzt DemoDB gleichzeitig als LogonProvider für eine Authentifizierung gegen die Tabelle USR. Die User Admin und Demo mit gleichlautendem Passwort sind bereits eingerichtet. Diese Datenbankauthentifizierung lässt sich leicht durch Änderungen in der Logon-Sektion auf andere, bereits existierende Datenbanken umstellen. Die Logon-Sektion für unser Beispiel sieht wie folgt aus:

**[Logon]**

```
LogonProvider=DemoDB  
LoginSQL=SELECT * FROM USR WHERE USR.USR_LOGON = :UN  
UserField=USR_LOGON  
PasswordField=USR_PASSWORD  
RoleField=USR_ROLE
```

Alternativ ist durch den Eintrag LogonProvider=Windows auch eine Windows-Authentifizierung möglich (siehe Administrations- und Installationsleitfaden).

---

**Achtung:** Damit Änderungen in der *cyProvider.ini* wirksam werden, muss der Service cyProvider neu gestartet werden.

### 3.1.4. Datei *myApplication.ini* erstellen

Die Datei *myApplication.ini* verschieben wir aus dem myApplications-Verzeichnis in das Verzeichnis cyProvider\Sources\Browser\Def-Files. In diesem Verzeichnis liegen für alle Anwendungen, die vom cyProvider einem anfragenden cyAgent angeboten werden sollen, entsprechende ini-Dateien. Mit Hilfe dieser ini-Datei wird der für das Logon zuständige cyProvider adressiert und die gewünschte Anwendung (Application=myApplication) für den download zum cyAgent bestimmt.

**[Application Initialization]**

```
Application=myApplication
Providers=Logon
AutoLogon=False
AppIni=cyGuiApl.ini
DefaultLanguage=de
;StoreFileDir=<vollständiger Verzeichnispfad>
```

**[Logon]**

```
ConnectionType=SOAP
URL=https://myhost.mydomain:443/soap/IMDCDataModule
```

Bei AutoLogon=True wird auf eine Anmeldung verzichtet, dafür erhalten alle User nur die Berechtigungen der Rolle Guest. Selbstverständlich lassen sich die beiden Verfahren, mit und ohne Anmeldung auch kombinieren. Während zum Beispiel anonyme Anwender die *myApplication.ini* aus dem Browser-Verzeichnis des cyProvider mit der Einstellung AutoLogon=True nutzen, starten Anwender, die sich anmelden müssen, die Anwendung über eine eigene lokale *myApplication.ini* mit AutoLogon=False (siehe Administrations- und Installationsleitfaden).

Der Parameter AppIni wird nur benötigt, wenn der Name, der in Kapitel 3.1.2 beschriebenen ini-Datei, von *cyGuiApl.ini* abweicht. Damit besteht die Möglichkeit eine Anwendung mit unterschiedlichen Parametern (z.B. unterschiedlichen URLs bei den ProviderVerbindungen) zu starten.

Über den Parameter DefaultLanguage kann die Sprache, in der die Anwendung standardmäßig starten soll, definiert werden. Fehlt der Parameter wird versucht die Anwendung in der Sprache des Client-PCs zu starten.

StoreFileDir bietet die Möglichkeit, für diese Anwendung ein vom <user>\cyGUI-Verzeichnis abweichendes Verzeichnis zum Abspeichern des Storefiles (siehe Kapitel 7.4), des Stylefiles (siehe Kapitel 9) und weiterer temporärer Dateien zu definieren. Dies kann zum

Beispiel sinnvoll sein, wenn Anwendungen unter gleichem Namen (Test-, Produktionssystem) gestartet werden.

### 3.1.5. Informationen zur Anwendung hinterlegen

Abschließend verschieben wir noch die Datei *DE\_myApplication.RTF* aus dem myApplication-Verzeichnis in das Verzeichnis Sources\Browser\Def-Files\RTF. Diese Datei enthält den deutschsprachigen Kurztext für die Präsentation der Anwendung im cyAgent Browser. Der Inhalt der Datei kann mit einem Texteditor wie z. B. Wordpad angepasst werden. DE bezieht sich dabei auf die eingestellte Sprache des Windows-Systems, auf dem der cyAgent gestartet wurde. Für weitere Sprachen müssen zusätzliche RTF-Files angelegt werden.

In dem Verzeichniss Sources\Browser\Def-Files\Images kann unter *myApplication.bmp* ein eigenes Bitmap als Anwendungs-Logo abgelegt werden.

### 3.1.6. Anwendung „myApplication“ mit dem cyAgent präsentieren

Startet der Anwender nun den cyAgent und gibt die URL des cyProvider ein (Abb. 9), der die Anwendung „myApplication“ und die zugehörige *myApplication.ini* bereithält, erscheint im folgenden Browser-Fenster (Abb. 10) die oben erwähnte Kurzbeschreibung und das Bitmap der Anwendung.

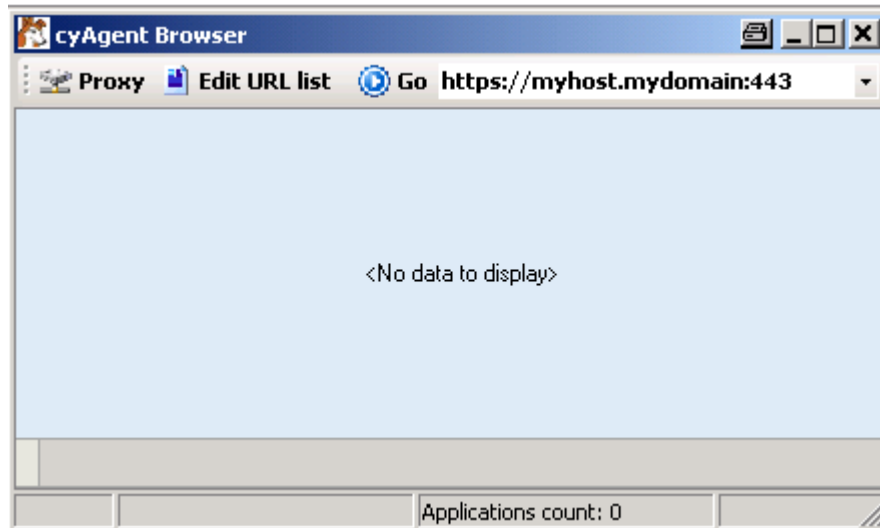


Abb. 9



Abb. 10

Nach Anklicken der Bitmap und der erfolgreichen Anmeldung mit Admin/Admin erscheint das Anwendungsfenster von „myApplication“ (Abb. 11), in dem unter dem Menüpunkt Module/Administration die Benutzerverwaltung aufgerufen werden kann (Abb. 12).

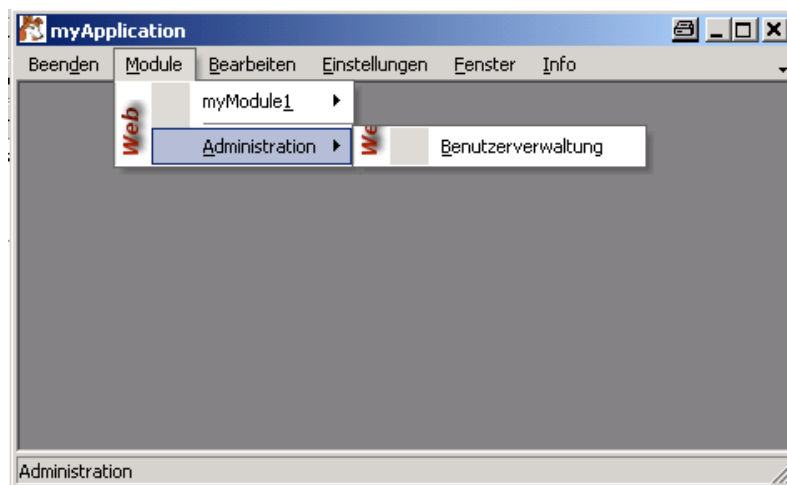


Abb. 11

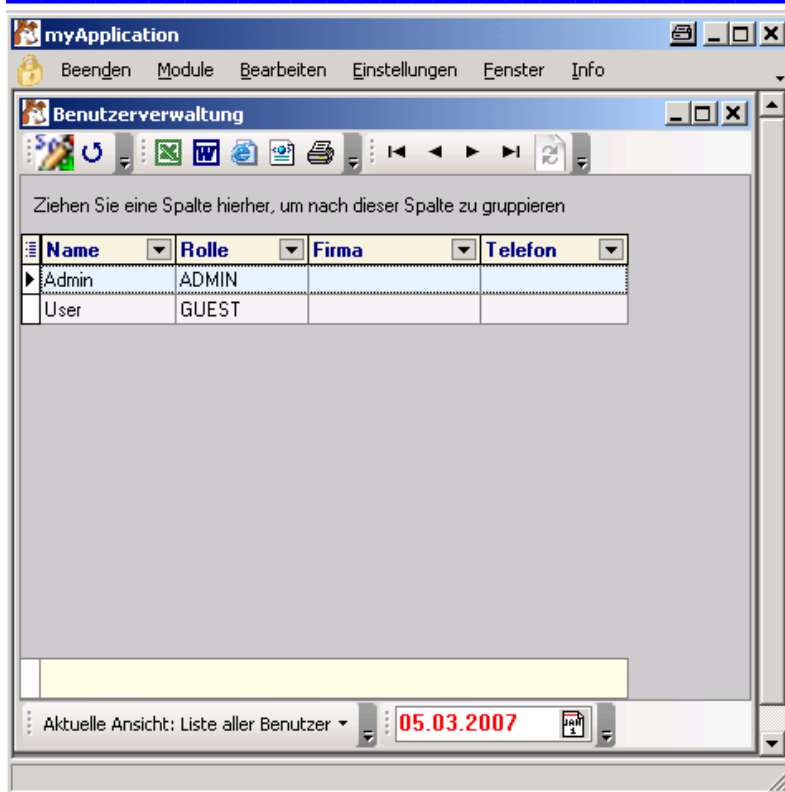


Abb. 12

---

## 4. Definition des Anwendungsmenüs

Das Menü einer Anwendung wird durch die Dateien mit der Bezeichnung `<Anwendungsname>.cdd`, hier `myApplication.cdd`, im Def-Files- bzw. in den jeweiligen Roles-Verzeichnissen definiert.

Die rollenunabhängige Definition des Hauptmenüs erfolgt in der `myApplication.cdd` des Def-Files-Verzeichnisses durch Aufzählung der einzelnen Menüpunkte. Jeder Menüpunkt erhält anschließend in der jeweiligen Sprachdatei seine Überschrift und einen Hinweistext (mouseover) zugewiesen. Zusätzlich wird noch der initiale Status des Anwendungsfensters (minimiert, maximiert, normal) und der Farbverlauf für die erste Spalte der Menüpunkte definiert.

In der `myApplication.cdd` des jeweiligen Rollenverzeichnisses erfolgen anschließend die rollenspezifischen Definitionen der Hauptmenüpunkte. Dabei kann jedem Menüpunkt ein Button, ein Menü zur Sprachauswahl oder ein weiteres Untermenü zugeordnet werden, so dass beliebige hierarchische Menüstrukturen erstellt werden können. Menüpunkte können durch eine Trennlinie (im Beispiel ein Bindestrich) untereinander abgegrenzt werden. Den Buttons werden letztlich die Aktionen zugeordnet. Aktionen beinhalten Aufrufe von Forms (Selektionsautomaten), InfoForms (Anzeige von RTF-Dokumenten) oder System-Kommandos (close, copy ...).

Für den Aufruf des Formulars `Frm_SAM_Countries_Cities` unter dem Menüpunkt `Module/Stammdaten (mnuModules11)` wird in der `myApplication.cdd` folgender Eintrag aufgenommen:

```
[mnuModules]
Type=TdxBarSubItem
Menus=mnuModules1 | - | mnuModules4

[mnuModules1]
Type=TdxBarSubItem
Menus=mnuModules11 | mnuModules12 | mnuModules13

[mnuModules11]
Type=TdxBarButton
PaintStyle=psCaptionGlyph
ButtonStyle=bsChecked

[mnuModules11.Action]
;Beispiel Kapitel 5.1 SAM
OnClick=Form.Frm_SAM_Countries_Cities
```

---

**[mnuModules12]**

```
Type=TdxBarButton  
PaintStyle=psCaptionGlyph  
ButtonStyle=bsChecked
```

**[mnuModules12.Action]**

```
;Beispiel Kapitel 5.2 SAM mit Master-Detail  
OnClick=Form.Frm_SAM_MD_Countries_Cities
```

**[mnuModules13]**

```
Type=TdxBarButton  
PaintStyle=psCaptionGlyph  
ButtonStyle=bsChecked
```

**[mnuModules13.Action]**

```
;Beispiel Kapitel 5.3 SAM mit Container  
OnClick=Form.Frm_SAM_CON_Countries_Cities
```

Anschließend werden in den Sprachdateien *de.cdd* bzw. *en.cdd* noch die Texte der Menüpunkte hinterlegt:

**[myApplication]**

```
...  
mnuModules.Caption=&Module  
mnuModules.Hint=Module der Anwendung  
mnuModules1.Caption=&Stammdaten  
mnuModules1.Hint=Stammdaten  
mnuModules11.Caption=&Länder / Städte  
mnuModules11.Hint=Länder / Städte  
mnuModules12.Caption=&Länder / Städte (Master-Detail)  
mnuModules12.Hint=Länder / Städte  
mnuModules13.Caption=&Länder / Städte mit Container  
mnuModules13.Hint=Länder / Städte
```

---

## 5. Selektionsautomaten

Im Folgenden werden mit Hilfe von Templates verschiedene Varianten von Selektionsautomaten für unsere Beispielanwendung „myApplication“ erstellt. Dokumentiert werden die jeweils benötigten Definitionen und Parameter. Weitere Parameter sind innerhalb der jeweiligen Template-Dateien beschrieben.

### 5.1. Standard-Selektionsautomaten

In unserem ersten einfachen Selektionsautomaten wollen wir jeweils alle Länder und alle Städte, die in unserer Datenbank DemoDB enthalten sind, anzeigen.

#### 5.1.1. Die Templatedatei *Def-Files\Tpl\_Frm\_SAM.cdd*

Aus dem Def-Files-Verzeichnis des Templates kopieren wir dazu die Datei *Tpl\_Frm\_SAM.cdd* in das Def-Files-Verzeichnis der Anwendung und benennen es in *Frm\_SAM\_Countries\_Cities.cdd* um. Innerhalb der Datei werden danach die Zeichen `_##` durch `_Countries_Cities` ersetzt, wodurch folgende Sektionen entstehen:

<b>[Frm_SAM_Countries_Cities]</b> Left=50 Width=500 Height=600 SizeConstraints=True	Formulardefinition: Formulargröße und Positionierung innerhalb des Anwendungsfensters
<b>[Frm_SAM_Countries_Cities.Link]</b> cySAM_Countries_Cities=TcySAM	die definierte Formulargröße kann nicht unterschritten werden Formular enthält einen Link auf einen Selektionsautomaten
<b>[cySAM_Countries_Cities]</b> ActiveStyleSheetName=reppredef_Standard Grid=Grid_Countries_Cities	Definition Selektionsautomat: initialer Standard-Style aus <i>_STYLES.cdd</i> Name des Grids

Neben Größe und Positionierung wurde also definiert, dass das Formular einen Link auf einen Selektionsautomaten enthält, der wiederum aus dem Grid `Grid_Countries_Cities` besteht. Die weiteren Definitionen zum Inhalt des Grids erfolgen nun rollenspezifisch im nächsten Schritt.

### 5.1.2. Die Templatedatei *Roles\Admin\Tpl\_Frm\_SAM.cdd*

Aus dem Roles\Admin-Verzeichnis des Templates wird ebenfalls die Datei *Tpl\_Frm\_SAM.cdd* in das Roles\Admin-Verzeichnis der Anwendung kopiert und in *Frm\_SAM\_Countries\_Cities.cdd* umbenannt. Die Grid Sektion erhält den oben festgelegten Namen *Grid\_Countries\_Cities* und alle Bezeichner *View\_01* bzw. *View\_02* werden in *View\_Countries* bzw. *View\_Cities* umbenannt:

```
[Grid_Countries_Cities]  
RootLevelOptions.DetailTabsPosition=dtptop  
Views=View_Countries|View_Cities
```

Die Definition des Grids bestimmt Anzahl und Namen der Views, sowie die Positionierung der Tabreiter, die den Wechsel von einer View zur anderen ermöglichen. Im Beispiel werden dem Grid *Grid\_Countries\_Cities* die Views *View\_Countries* und *View\_Cities* zugeordnet.

Stellt ein Grid mehr als eine View zur Verfügung, so ist zu berücksichtigen, dass die Anzeige der aktiven View erst dann erfolgt, wenn die Beschaffung der Daten aller Views abgeschlossen ist. Für den Fall, dass der damit verbundene Performance-Verlust zu groß wird, sollte der Selektionsautomat mit Hilfe von Containern (siehe Kapitel 5.3) definiert werden.

```
[View_Countries]  
Provider=DemoDB  
IndexName=COU_IDX  
DateFormat=DDMMYYYY  
TimeFormat=HHNNSS  
VisibleFields=COU_CODE|COU_COUNTRY|FOG_FOG|COU_NATIONALITY|  
COU_VALID_FROM|COU_VALID_UP_TO  
HiddenFields=COU_IDX|FOG_IDX  
Filters=Filter0|Filter1  
Bands=View_Countries_Band1|View_Countries_Band2
```

Zwingend erforderlich sind die Angabe des Daten-Providers, über den der cyProvider die Daten bereitstellen soll, und die Benennung des Index-Feldes. Setzt sich der Index (eindeutiger Schlüssel einer Tabelle) aus mehreren Feldern zusammen, erhält *IndexName* den Wert *cyRowId* und *cyRowId* als Wert die Aufzählung der Felder, die zusammen den eindeutigen Schlüssel ergeben.

Die Parameter *DateFormat* und *TimeFormat* werden für den Fall benötigt, dass Datums- bzw. Zeitfelder als String-Felder in der Datenbank definiert wurden.

---

Unter VisibleFields werden die Felder der View aufgezählt, die beim ersten Formularaufruf im Grid sichtbar sein sollen. Mit Hilfe von HiddenFields werden Felder, die zwar Teil der SQL-Abfrage sind, aber nicht im Grid erscheinen sollen, wieder unsichtbar gemacht.

Unter Filters und Bands werden die Namen der benötigten Filter und Bänder einer View angegeben, die in jeweils einer eigenen Sektion weiter definiert werden (s.u.).

**[View\_Countries.SQL]**

```
select COU.COU_IDX,  
       FOG.FOG_IDX,  
       COU.COU_CODE,  
       COU.COU_COUNTRY_%LanguageID% as COU_COUNTRY,  
       FOG.FOG_FOG_%LanguageID% as FOG_FOG,  
       COU.COU_NATIONALITY_%LanguageID% as COU_NATIONALITY,  
       COU.COU_VALID_FROM,  
       COU.COU_VALID_UP_TO  
FROM COU left join FOG on COU.FOG_IDX = FOG.FOG_IDX  
order by COU.COU_COUNTRY_%LanguageID%
```

Diese Sektion enthält das SQL-Statement zur View. Die Reihenfolge der Datenbankfelder im select-Teil entspricht der Reihenfolge ihrer Anzeige im Grid.

Der Placeholder (Variable) %LanguageID% enthält die ID der jeweils aktiven Sprache der Anwendung. In unserem Beispiel die Ziffer 0 für deutsch oder 1 für englisch. Die möglichen IDs werden in der *cyGuiApl.ini* definiert. Damit erhält das hinter as definierte alias-Feld den Wert des sprachabhängigen Datenbankfeldes.

**[View\_Countries.COU\_VALID\_FROM]**

```
Format=fsd  
;Decimals=
```

Eine Formatierung einzelner Datenbankfelder wie im obigen Beispiel für das Feld COU\_VALID\_FROM ist an dieser Stelle nicht erforderlich. Sie wird in erster Linie bei numerischen Feldern mit Gleitkomma-Arithmetik (Darstellung, Nachkommastellen) und Datums- bzw. Zeitfeldern, die im Charakterformat in der Datenbank gespeichert sind, benötigt. Die Formatierung von numerischen Feldern ist zudem Voraussetzung für die Aktivierung der Grid-Funktionen (Summe, Durchschnitt...).

---

Für die Realisierung der beiden Ansichten „Liste aller Länder“ und „Liste aller gültigen Länder zum Stichtag“ werden die beiden Filter Filter0 und Filter1 verwendet.

```
[View_countries.Filter0]
```

```
[View_countries.Filter1]
```

```
((COU_VALID_FROM is null) or (COU_VALID_FROM <= :D)) and  
((COU_VALID_UP_TO is null) or (COU_VALID_UP_TO >= :D))
```

Filter0 übernimmt die selektierte Datenmenge gemäß der SQL-Definition. Er dient nur zur Zuordnung der Überschrift „Liste aller Länder“. Im Gegensatz dazu ist Filter1 ein echter Filter, der die Datenmenge mit Hilfe einer Bedingung weiter einschränkt. Die gültig-von- und gültig-bis-Felder werden mit dem selektierten Datum **:D** (**:SD**, falls das Datum im Stringformat auf der Datenbank angelegt wurde) im Selektionsautomat verglichen.

Mit Hilfe von Bändern kann man Felder eines Grids in einem Bereich (Band) mit einer zusätzlichen gemeinsamen Bereichs-Überschrift anordnen. Alle Felder mit gleichem Wert werden nebeneinander in der gleichen Zeile (gleiche Hierarchiestufe) angezeigt. Andernfalls erscheinen Felder mit höherem Wert unterhalb des Feldes mit niedrigerem Wert.

```
[View_countries_Band1]
```

```
COU_CODE=0  
COU_COUNTRY=0  
FOG_FOG=0  
COU_NATIONALITY=0
```

```
[View_countries_Band2]
```

```
COU_VALID_FROM=0  
COU_VALID_UP_TO=0
```

In der folgenden Sektion wird dem Ereignis Doppelklick der Aufruf eines Pflegeautomaten (s.u.) mit Namen Frm\_PAM\_Countries zugeordnet. Damit öffnet sich bei einem Doppelklick in einer Zeile des Selektionsautomaten der Pflegeautomat zur Pflege des selektierten Landes. Erfolgt ein Doppelklick genau auf dem Feld FOG\_FOG (Staatsform) wird der Pflegeautomat Frm\_PAM\_FOG aufgerufen. Beide Pflegeautomaten sind natürlich noch zu definieren. Um dem Anwender zu signalisieren, dass einzelnen Feldern oder einer Zeile eines Selektionsautomaten Aktionen (Klick, Doppelklick) zugeordnet sind, ändert sich die Darstellung des Mauszeigers (Hand-Symbol anstelle des Pfeil-Symbols). Zusätzlich können in der Language-Datei noch Hinweise (ActionHints) hinterlegt werden (siehe Kapitel 8.2).

```
[View_countries.ACTION]
```

```
OnDBLClick=Form.Frm_PAM_Countries  
OnDBLClick.FOG_FOG=Form.Frm_PAM_FOG
```

Abhängig vom Dateninhalt kann einer Zelle eines Grids ein anderer Style zugeordnet werden. Im Beispiel erhält das Feld FOG\_FOG (Staatsform) eine Stylecondition.

```
[View_Countries.FOG_FOG]
StyleCondition=FOG_UNKNOWN_STYLE
```

Die Definition der Stylecondition FOG\_UNKNOWN\_STYLE liegt in der *\_DESIGN.cdd* im Def-Files-Verzeichnis und kann damit von allen Selektionsautomaten benutzt werden.

```
[FOG_UNKNOWN_STYLE]      alle unbekanntenen Staatsformen
pinkblack=(FOG_IDX = 0)  (Index = 0) werden in Schwarz auf
                          Pink dargestellt

[pinkblack]
color=$00C1C1FF
font.name=Arial
font.color=$00000000
```

Es folgen die Sektionen zur Definition der zweiten View View\_Cities:

```
[View_Cities]
Provider=DemoDB
IndexName=CIT_IDX
DateFormat=DDMMYYYY
TimeFormat=HHNNSS
VisibleFields=CIT_CITY|CIT_VALID_FROM|CIT_VALID_UP_TO
HiddenFields=CIT_IDX
Filters=Filter0|Filter1
```

```
[View_Cities.SQL]
select
  CIT.CIT_IDX,
  CIT.CIT_CITY_%LanguageID% as CIT_CITY,
  CIT.CIT_CITY_LOCAL,
  CIT.CIT_CITY_INT,
  CIT.CIT_VALID_FROM,
  CIT.CIT_VALID_UP_TO
from CIT
order by CIT.CIT_CITY_%LanguageID%
```

```
[View_Cities.Filter0]
```

```
[View_Cities.Filter1]
(((CIT_VALID_FROM is null) or (CIT_VALID_FROM <= :D)) and
((CIT_VALID_UP_TO is null) or (CIT_VALID_UP_TO >= :D)))
```

---

```
[View_Cities.ACTION]  
OnDBLClick=Form.Frm_PAM_Cities
```

Abschließend müssen in den Sprachdateien *de.cdd* und *en.cdd* noch die Überschriften für das Formular und für die Views, inkl. der Filter, der Bänder und der Datenbankfelder hinterlegt werden.

```
[Frm_SAM_Countries_Cities]  
Caption=Länder / Städte
```

```
[View_Countries]  
Caption=Länder  
Filter0=Liste aller Länder  
Filter1=Liste aller zum Stichtag gültigen Länder  
View_Countries_Band1.Caption=Stammdaten  
View_Countries_Band2.Caption=Gültigkeitszeitraum  
COU_IDX.Caption=Länder-Id.  
COU_CODE.Caption=Kennung  
COU_COUNTRY.Caption=Ländernamen  
COU_NATIONALITY.Caption=Staatsangehörigkeit  
COU_VALID_FROM.Caption=gültig ab  
COU_VALID_UP_TO.Caption=gültig bis  
FOG_FOG.Caption=Staatsform
```

```
[View_Cities]  
Caption=Städte  
Filter0=Liste aller Städte  
Filter1=Liste aller zum Stichtag gültigen Städte  
CIT_IDX.Caption=Stadt-Id.  
CIT_CITY.Caption=Stadt  
CIT_CITY_LOCAL.Caption=Stadt (national)  
CIT_CITY_INT.Caption=Stadt (international)  
CIT_VALID_FROM.Caption=gültig ab  
CIT_VALID_UP_TO.Caption=gültig bis
```

Das Ergebnis dieser Definitionen, interpretiert und dargestellt durch den cyAgent, sind in der folgenden Abbildung Abb. 13 zu sehen:

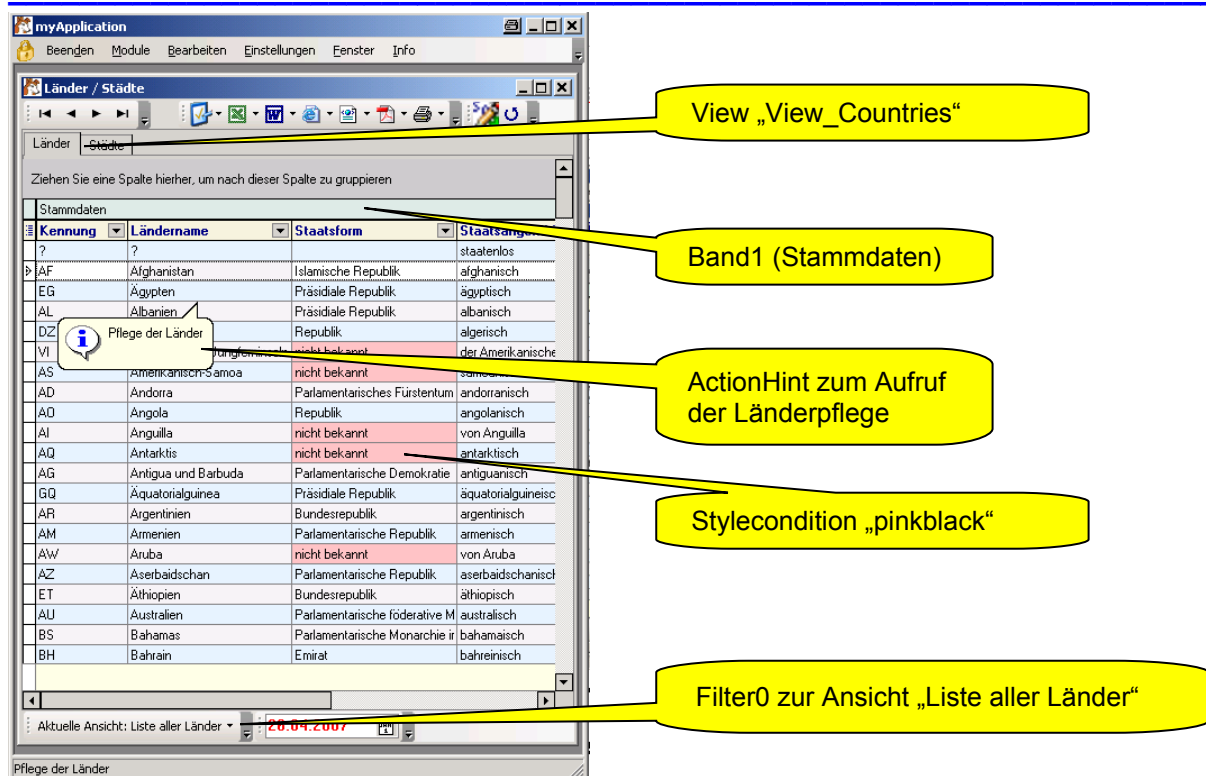


Abb. 13

## 5.2. Selektionsautomaten mit Master-Detail-Beziehung

**Achtung:** Bitte diese Form von Selektionsautomaten nicht mehr benutzen! Diese Technik wird in Zukunft nicht mehr unterstützt. Selektionsautomaten mit Containern sind die bessere Alternative.

Im letzten Kapitel wurden in einem Selektionsautomaten zwei voneinander unabhängige Views, die mit Hilfe von Tabreibern im Grid ausgewählt werden konnten, präsentiert. Nun wollen wir diese beiden Views in eine Master-Detail-Beziehung zueinander setzen. Das heißt, in der View View\_Cities sollen jeweils nur die Städte des zuvor in der View View\_Countries selektierten Landes angezeigt werden.

### 5.2.1. Die Templatedatei Def-Files\Tpl\_Frm\_SAM\_Master\_Detail.cdd

Analog zum Beispiel in Kapitel 5.1.1 kopieren wir zunächst die Datei *Tpl\_Frm\_SAM\_Master\_Detail.cdd* aus dem Def-Files-Verzeichnis des Templates in das gleichnamige Verzeichnis unserer Anwendung und benennen es in *Frm\_SAM\_MD\_Countries\_Cities.cdd* um. Innerhalb der Datei werden dann noch die Zeichen `##_` durch `_MD_Countries_Cities` ersetzt.

---

### 5.2.2. Die Templatedatei *Roles\Admin\Tpl\_Frm\_SAM\_Master\_Detail.cdd*

Mit dem Template *Tpl\_Frm\_SAM\_Master\_Detail.cdd* verfahren wir ebenso wie unter 5.1.2 beschrieben oder kopieren gleich das obige Ergebnis nach *Frm\_SAM\_MD\_Countries\_Cities.cdd*. In beiden Fällen ist darauf zu achten, dass der Name der Grid-Sektion `GRID_MD_Countries_Cities` lautet. Für die Master-Detail-Beziehung müssen wir nun folgende Änderungen vornehmen:

In der `Grid_Countries`-Sektion entfernen wir die View `View_Cities` aus der View-Liste.

```
[Grid_MD_Countries_Cities]  
RootLevelOptions.DetailTabsPosition=dtptop  
Views=View_Countries
```

Dafür fügen wir in der `View_Countries`-Sektion eine neue View-Liste mit der View `View_Cities` ein.

```
[View_Countries]  
Provider=DemoDB  
IndexName=COU_IDX  
DateFormat=DDMMYYYY  
TimeFormat=HHNNSS  
VisibleFields=COU_CODE|COU_COUNTRY|FOG_FOG|COU_NATIONALITY|  
COU_VALID_FROM| COU_VALID_UP_TO  
HiddenFields=COU_IDX|FOG_IDX  
Views=View_Cities  
Filters=Filter0|Filter1  
Bands=View_Countries_Band1|View_Countries_Band2
```

Damit stehen die beiden Views nicht mehr unabhängig nebeneinander, sondern sind miteinander verknüpft. In der Sektion `View_Cities` müssen wir nun noch den Namen des Schlüsselfeldes (im Beispiel der Länderschlüssel `COU_IDX`) bekannt geben, über den die Master-Detail-Verknüpfung realisiert werden soll (`Master-.u. DetailKeyFieldNames`). Da dieses Feld in der Master- bzw. Detail-View anders heißen kann, müssen an dieser Stelle beide Namen angegeben werden.

```
[View_Cities]  
Provider=DemoDB  
IndexName=CCM_IDX  
DataController.MasterKeyFieldNames=COU_IDX  
DataController.DetailKeyFieldNames=COU_IDX  
DateFormat=DDMMYYYY  
TimeFormat=HHNNSS  
VisibleFields=CIT_CITY|CIT_VALID_FROM|CIT_VALID_UP_TO  
HiddenFields=CIT_IDX
```

---

Filters=Filter0|Filter1

Abschließend müssen wir noch das SQL-Statement für unsere View View\_Cities austauschen. Wir benötigen zusätzlich zur Tabelle CIT noch die Zuordnungstabelle CCM, die die n:m-Beziehung von Ländern zu Städten jeweils durch eine 1:n-Beziehung auflöst. Damit existiert der Schlüssel COU\_IDX auch in unserer View View\_Cities und die Master-Detail-Beziehung kann hergestellt werden.

**[View\_Cities.SQL]**

```
select
  CCM.CCM_IDX,
  CCM.CIT_IDX,
  CCM.COU_IDX,
  CIT.CIT_CITY_ %LanguageID% as CIT_CITY,
  CIT.CIT_CITY_LOCAL,
  CIT.CIT_CITY_INT,
  CCM.CCM_VALID_FROM,
  CCM.CCM_VALID_UP_TO
from CCM, CIT
where CCM.CIT_IDX = CIT.CIT_IDX
order by CCM.COU_IDX, CIT.CIT_CITY_ %LanguageID%
```

Zu beachten ist, dass bei der Detail-View immer der gesamte Datenbestand (hier alle Städte) gelesen wird und dieser nach dem Schlüsselfeld DetailKeyField (hier COU\_IDX) sortiert sein muss. Wird in der Master-View dann ein Datensatz selektiert und die Detail-View angezeigt, werden alle nicht zum Schlüsselfeld gehörenden Sätze weggefiltert.

Dieses Verfahren sollte demnach nur angewendet werden, wenn die Detail-View nicht zu große Datenmengen bereitstellen muss. Ansonsten sollten auch in diesem Fall aus Performance-Gründen besser Selektionsautomaten mit Containern eingesetzt werden.

Das Ergebnis unserer Definitionen zeigt Abb. 14:

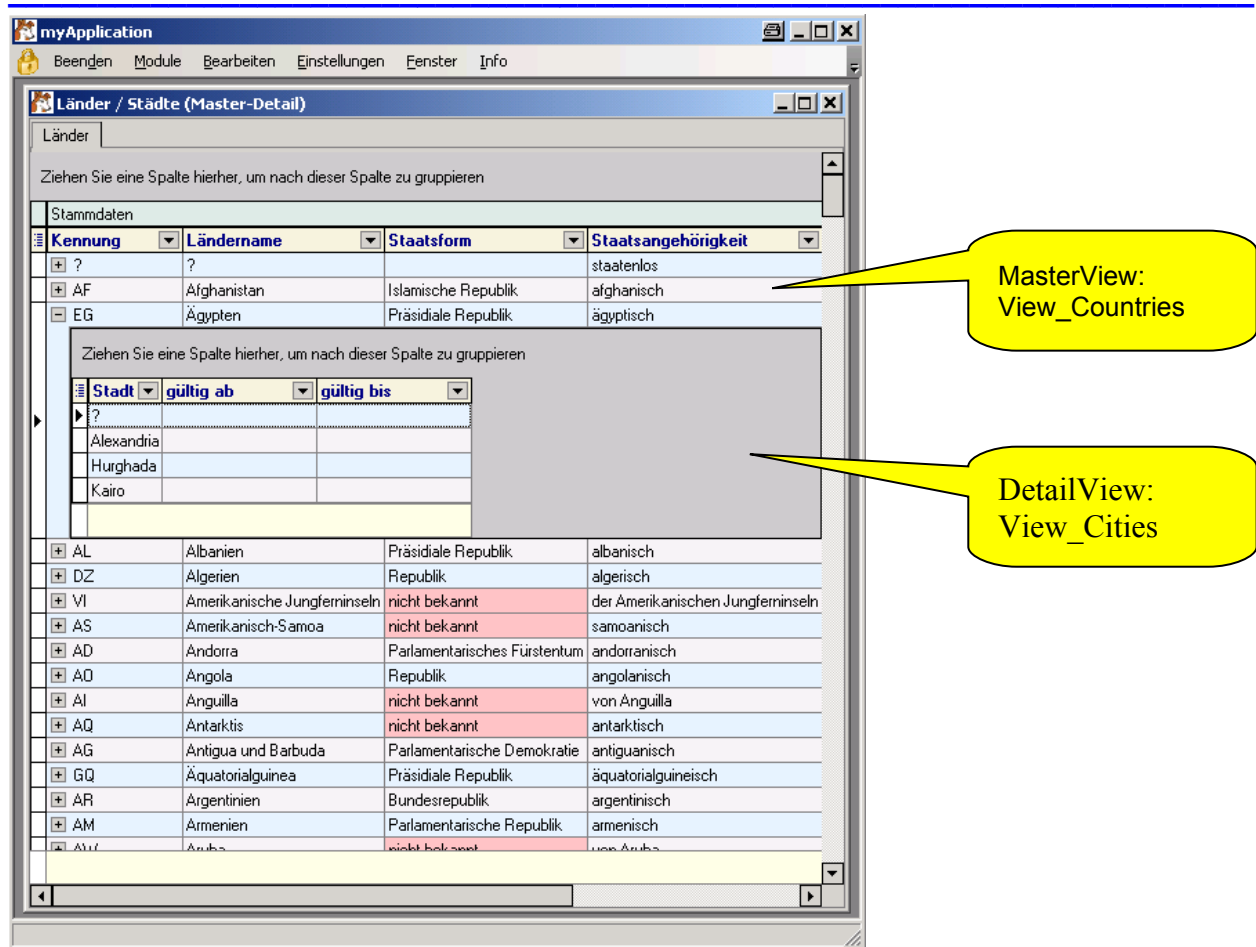


Abb. 14

### 5.3. Selektionsautomaten und Container

Container bieten die Möglichkeit, ein Formular in Bereiche aufzuteilen, in denen Selektionsautomaten und/oder Pflegeautomaten untergebracht werden können. Diese können untereinander abhängig (verlinkt), aber auch völlig unabhängig voneinander sein.

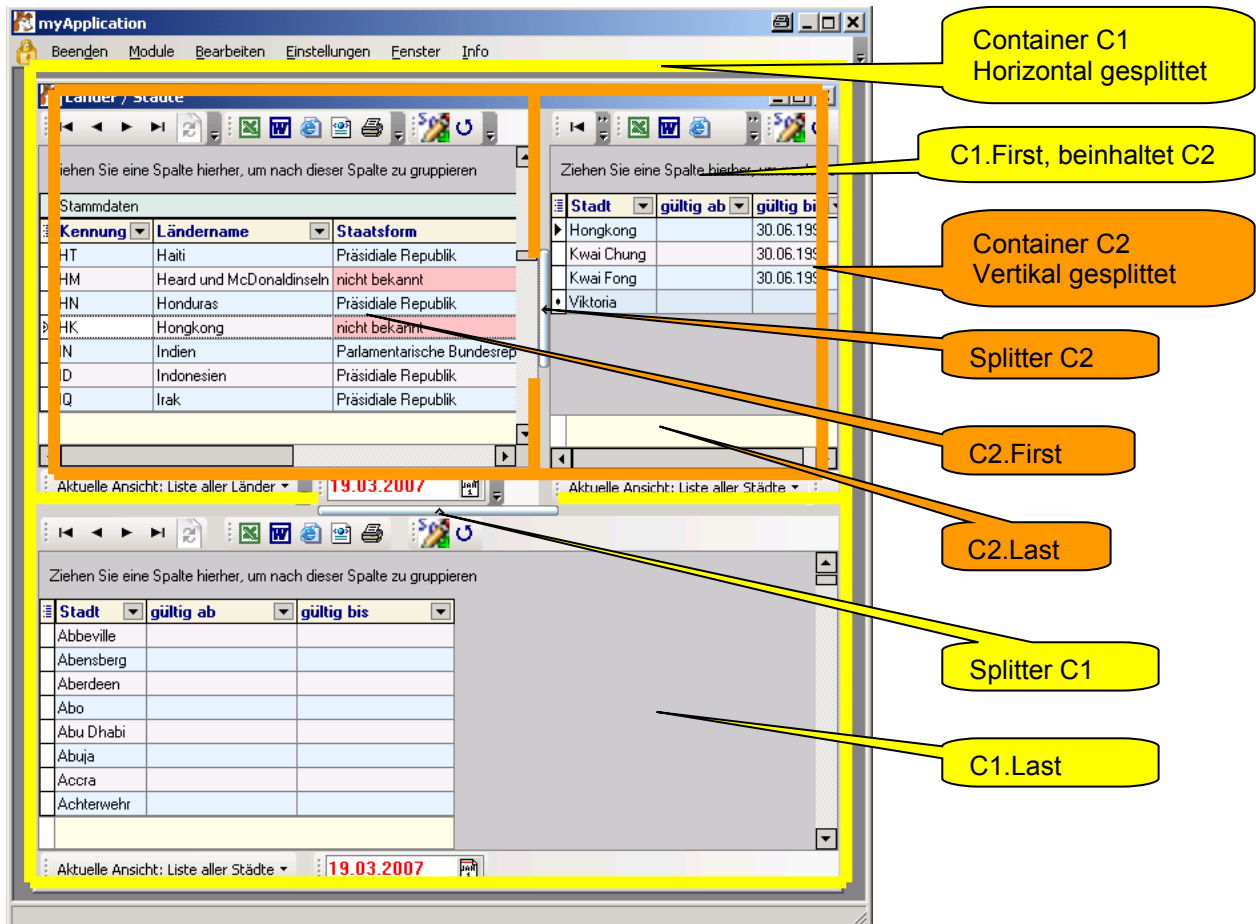


Abb. 15

Ein Container besteht aus maximal zwei Bereichen, die über <ContainerName>.First und <ContainerName>.Last adressiert werden. Das obige Beispiel zeigt die beiden Container C1 und C2. C1 ist horizontal durch einen sogenannten Splitter in die beiden Bereiche C1.First und C1.Last unterteilt. Der Container C2 ist vertikal unterteilt und belegt den gesamten Bereich von C1.First. Dazu sind die beiden Selektionsautomaten in C2 noch verlinkt, um eine Master-Detail-Beziehung abzubilden. Im Beispiel zeigt der Selektionsautomat in C2.Last Detail-Daten zum gerade selektierten Datensatz des Selektionsautomaten in C2.First.

Wie bereits das Beispiel mit zwei Containern zeigt, lässt sich ein Formular durch Verschachtelung von Containern beliebig unterteilen.

Ein zusätzliches Gestaltungsmittel, die beiden Bereiche von Containern zu untergliedern, bilden die Tabreiter. Tabreiter bieten die Möglichkeit, den Bereich eines Containers mit einem anderen Inhalt zu füllen. Natürlich können hinter einem Tabreiter wiederum weitere Container liegen. Dabei ist zu berücksichtigen, dass immer nur die Daten des aktivierten Tabreiter beschafft werden.

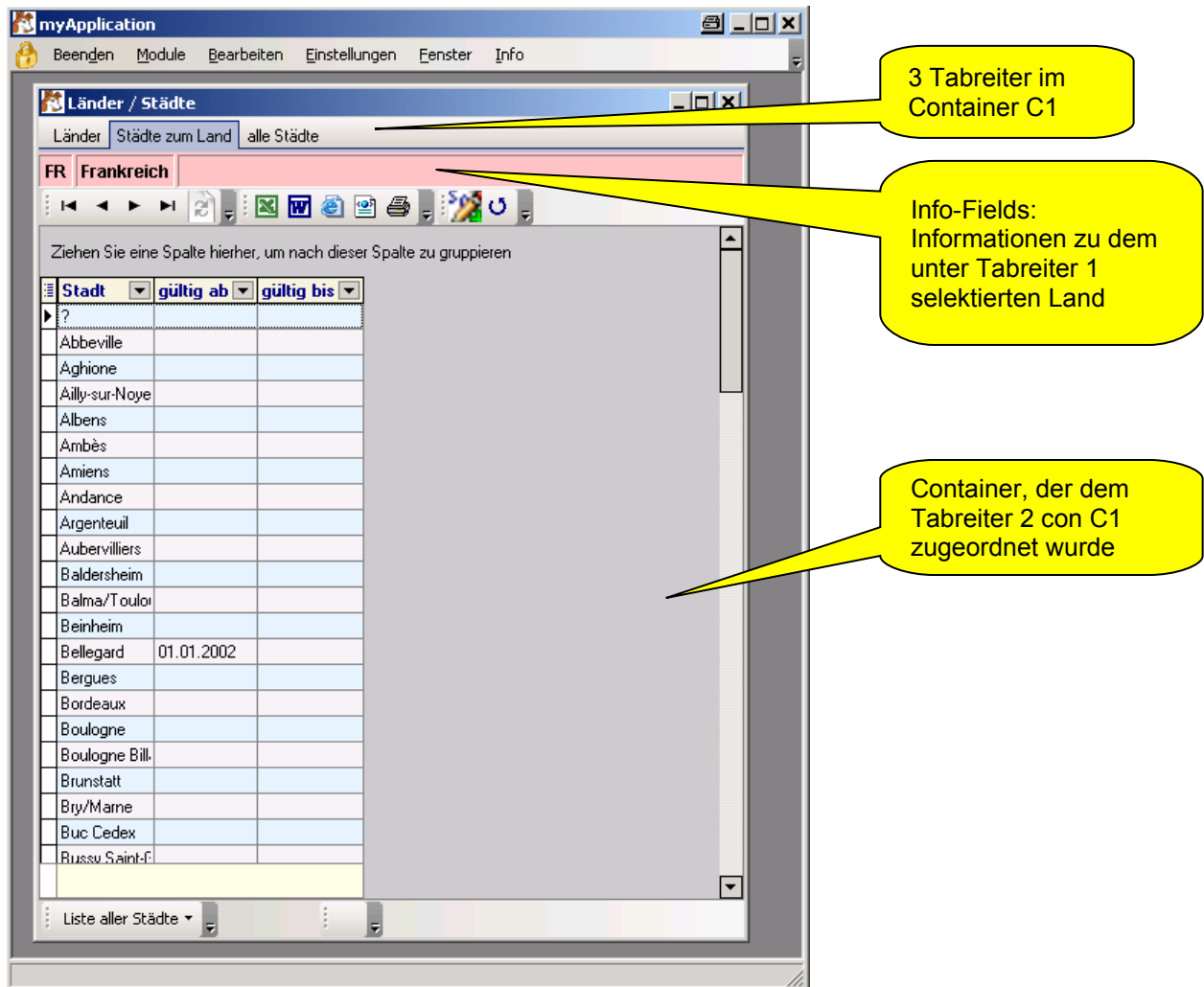


Abb. 16

Unter Performance-Gesichtspunkten sind Container mit Tabreitern die bessere Alternative im Vergleich zu den zuvor beschriebenen Lösungen, da mit der Präsentation der Daten nicht gewartet werden muss, bis die Daten für alle Views innerhalb des Formulars beschafft worden sind.

Die Reihenfolge der Tabreiter wird über eine eigene TabOrder-Sektion vorgegeben:

**[<formname>.TabOrder]**

```
<containername1>.FIRST=<tabname1>|<tabname2>|<tabname3>...
<containername1>.LAST=<tabname1>|<tabname2>|<tabname3>...
<containername2>.FIRST=<tabname1> ...<
```

### 5.3.1. Die Templatedatei *Def-Files\Tpl\_Frm\_SAM\_Container.cdd*

Wie zuvor schon angedeutet, bieten Container vielfältige Möglichkeiten mehrere Selektionsautomaten auf einem Formular anzuordnen. Im Template *Tpl\_Frm\_SAM\_Container.cdd* ist eine Reihe von Beispielen aufgeführt. Anhand von Beispiel 2 werden wir nun das obige Beispiel mit den zwei Containern und drei Selektionsautomaten definieren.

Nachdem wir das Template ins Def-Files-Verzeichnis kopiert und in *Frm\_SAM\_Countries\_Cities.cdd* umbenannt haben, löschen wir alle Definitionen, die nicht zum Beispiel 2 gehören. Anschließend sind die einzelnen Sektionen folgendermaßen anzupassen:

**[Frm\_SAM\_CON\_Countries\_Cities]**

```
Top=50
Left=50
Width=500
Height=600
SizeConstraints=True
Container=C1 | C2
```

**[Frm\_SAM\_CON\_Countries\_Cities.C1]**

```
;der Container C1 wird horizontal im Verhältnis 50:50
;gesplittet und dem Formular direkt zugeordnet (Home=Form)
Home=Form
Split=Horizontal
Dimension=50
```

**[Frm\_SAM\_CON\_Countries\_Cities.C2]**

```
;der Container C2 wird vertikal im Verhältnis 60:40
;gesplittet und dem oberen Bereich von Container C1
;zugeordnet (Home=C1.First)
Home=C1.First
Split=Vertical
Dimension=60
```

**[Frm\_SAM\_CON\_Countries\_Cities.Link]**

```
;die beiden unabhängigen SAM's
```

---

```
cySAM_Countries=TcySAM
cySAM_Cities_all=TcySAM
```

**[cySAM\_Countries]**

```
;SAM Countries wird dem 1. Bereich von Container C2
;zugeordnet
Container=C2.First
ActiveStyleSheetName=reppredef_Standard
Grid=Grid_Countries
SyncSelDateChange=cySAM_Cities|cySAM_Cities_all
;durch SyncSelDateChange wird bei Änderung des
Selektionsdatum auch das Selektionsdatum der anderen SAM's
entsprechend geändert
```

**[cySAM\_Countries.Link]**

```
;SAM Countries erhält einen zusätzlichen Link auf den
;abhängigen SAM Cities
cySAM_Cities=TcySAM
```

**[cySAM\_Cities]**

```
;SAM Cities wird dem 2. Bereich von Container C2 zugeordnet
;und erhält als Master View, die View des SAM's Countries
Container=C2.Last
ViewMaster=View_countries
ActiveStyleSheetName=reppredef_Standard
Grid=Grid_Cities
```

**[cySAM\_Cities\_all]**

```
;der unabhängige SAM Cities_all wird dem 1. Bereich von
;Container C1 zugeordnet
Container=C1.Last
ActiveStyleSheetName=reppredef_Standard
Grid=Grid_Cities_all
```

**5.3.2. Die Templatedatei Roles\Admin\Tpl\_Frm\_SAM\_Container.cdd**

Mit dem Template *Tpl\_Frm\_SAM\_Container.cdd* des Rollenverzeichnisses wird entsprechend verfahren, d. h. Template kopieren, umbenennen und die nicht benötigten Beispiele löschen. Es verbleiben die folgenden drei bereits angepassten Grid-Definitionen in der Datei:

**[Grid\_Countries]**

```
RootLevelOptions.DetailTabsPosition=dtNone
Views=View_Countries
```

**[View\_countries]**

```
Provider=DemoDB
IndexName=COU_IDX
DateFormat=DDMMYYYY
TimeFormat=HHNNSS
VisibleFields=COU_CODE|COU_COUNTRY|FOG_FOG|COU_NATIONALITY
HiddenFields=COU_IDX|FOG_IDX
Filters=Filter0|Filter1
Bands=View_countries_Band1|View_countries_Band2
```

**[View\_countries\_Band1]**

```
COU_CODE=0
COU_COUNTRY=0
FOG_FOG=0
COU_NATIONALITY=0
```

**[View\_countries\_Band2]**

```
COU_VALID_FROM=0
COU_VALID_UP_TO=0
```

**[View\_countries.SQL]**

```
select COU.COU_IDX,
       FOG.FOG_IDX,
       COU.COU_CODE,
       COU.COU_COUNTRY_ %LanguageID% as COU_COUNTRY,
       FOG.FOG_FOG_ %LanguageID% as FOG_FOG,
       COU.COU_NATIONALITY_ %LanguageID% as COU_NATIONALITY,
       COU.COU_VALID_FROM,
       COU.COU_VALID_UP_TO
FROM COU left join FOG on COU.FOG_IDX = FOG.FOG_IDX
order by COU.COU_COUNTRY_ %LanguageID%
```

**[View\_countries.Filter0]****[View\_countries.Filter1]**

```
((COU_VALID_FROM is null) or (COU_VALID_FROM <= :D)) and
((COU_VALID_UP_TO is null) or (COU_VALID_UP_TO >= :D))
```

**[View\_countries.ACTION]**

```
OnDBLClick=Form.Frm_PAM_Countries
```

**[View\_Countries.FOG\_FOG]**

```
StyleCondition=FOG_UNKNOWN_STYLE
```

---

**[Grid\_Cities]**

RootLevelOptions.DetailTabsPosition=dtbNone  
Views=View\_Cities

**[View\_Cities]**

Provider=DemoDB  
IndexName=CCM\_IDX  
DateFormat=DDMMYYYY  
TimeFormat=HHNNSS  
VisibleFields=CIT\_CITY|CIT\_VALID\_FROM|CIT\_VALID\_UP\_TO  
HiddenFields=CIT\_IDX  
Filters=Filter0|Filter1

**[View\_Cities.SQL]**

```
select
  CCM.CCM_IDX,
  CCM.CIT_IDX,
  CIT.CIT_CITY_%LanguageID% as CIT_CITY,
  CIT.CIT_CITY_LOCAL,
  CIT.CIT_CITY_INT,
  CCM.CCM_VALID_FROM,
  CCM.CCM_VALID_UP_TO
from CCM, CIT
where CCM.CIT_IDX = CIT.CIT_IDX and
      CCM.COU_IDX = :COU_IDX
order by CIT.CIT_CITY_%LanguageID%
```

**[View\_Cities.Filter0]****[View\_Cities.Filter1]**

```
((CIT_VALID_FROM is null) or (CIT_VALID_FROM <= :D)) and
((CIT_VALID_UP_TO is null) or (CIT_VALID_UP_TO >= :D))
```

**[View\_Cities.ACTION]**

OnDBLClick=Form.Frm\_PAM\_Cities

**[Grid\_Cities\_all]**

RootLevelOptions.DetailTabsPosition=dtbNone  
Views=View\_Cities\_all

**[View\_Cities\_all]**

Provider=DemoDB  
IndexName=CCM\_IDX  
DateFormat=DDMMYYYY

---

```
TimeFormat=HHNNSS  
VisibleFields=CIT_CITY|CIT_VALID_FROM|CIT_VALID_UP_TO  
HiddenFields=CIT_IDX  
Filters=Filter0|Filter1
```

---

**[View\_Cities\_all.SQL]**

```
select
  CIT.CIT_IDX,
  CIT.CIT_CITY_ %LanguageID% as CIT_CITY,
  CIT.CIT_CITY_LOCAL,
  CIT.CIT_CITY_INT,
  CIT.CIT_VALID_FROM,
  CIT.CIT_VALID_UP_TO
from CIT
order by CIT.CIT_CITY_ %LanguageID%
```

**[View\_Cities\_all.Filter0]****[View\_Cities\_all.Filter1]**

```
((CIT_VALID_FROM is null) or (CIT_VALID_FROM <= :D)) and
((CIT_VALID_UP_TO is null) or (CIT_VALID_UP_TO >= :D))
```

**[View\_Cities\_all.ACTION]**

```
OnDBLClick=Form.Frm_PAM_Cities
```

Zu beachten ist im SQL-Statement zur View VIEW\_Cities die Eingrenzung der Datenmenge durch den Parameter :COU\_IDX, der den aktuell selektierten Länderschlüssel aus der Master-View View\_Countries enthält. Eine derartige Parameterübergabe ist nur zwischen verlinkten Selektionsautomaten gegeben und ermöglicht damit die Darstellung einer Master-Detail-Beziehung. In unserem Beispiel werden also im Selektionsautomaten Cities nur die Städte des im Selektionsautomaten Countries selektierten Landes angezeigt.

## 5.4. Selektionsautomaten und FastReport®

Alle Selektionsautomaten bieten mit der Toolbox-Bar die Möglichkeit, die Daten eines Grids in ein Word-Dokument, in eine Excel-Tabelle, in ein HTML- oder in ein XML-Dokument zu überführen. Die Darstellung der Daten wird dabei eins zu eins übernommen, wobei im Grid gebildete Knoten aufgelöst werden.

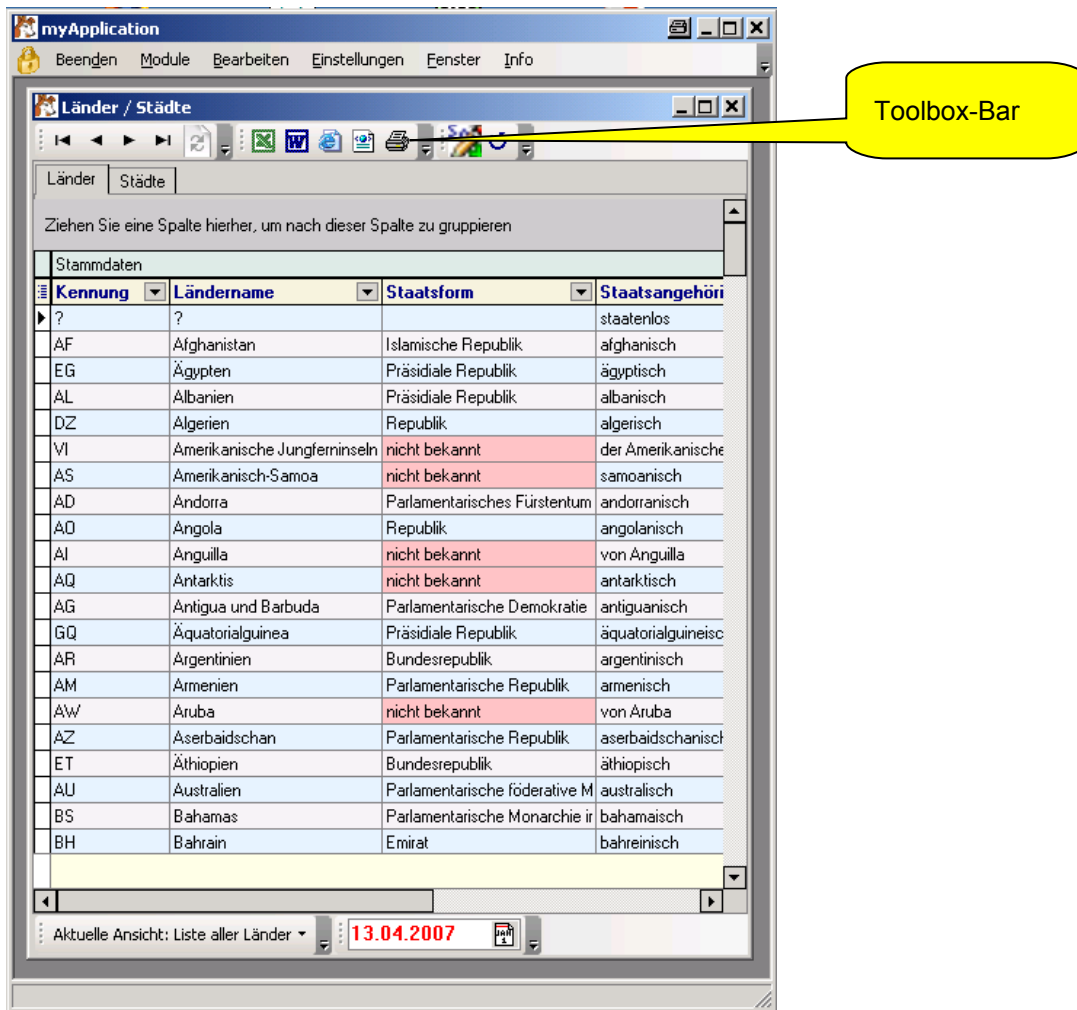


Abb. 17

Mit der Integration von FastReport® in cySystem wird die Toolbox nicht nur um eine zusätzliche PDF-Schnittstelle, sondern um eine vollständige, sehr flexible Report-Komponente erweitert.

---

Im folgenden Beispiel werden wir unseren Länder-Selektionsautomaten *Frm\_SAM\_Countries\_Cities.cdd* aus Kapitel 5.1.2 um den Report „Städteverzeichnis“ erweitern.

In der View-Countries-Sektion ordnen wir über den Reports-Parameter dem Selektionsautomaten seine Reports zu. Im Beispiel beschränken wir uns auf einen Report mit Namen Rep\_Cities.

```
[View_Countries]  
...  
Reports=Rep_Cities
```

Abschließend müssen wir noch die Datasets (Datenmengen) definieren, die wir dem Report zur Auswertung zur Verfügung stellen.

```
[View_Countries.Rep_Cities]  
Datasets=DS1  
  
[View_Countries.Rep_Cities.DS1]  
Bands=Band1  
Source=SQL  
Provider=DemoDB  
  
[View_Countries.Rep_Cities.DS1.SQL]  
select  
    CCM.CCM_IDX,  
    CCM.CIT_IDX,  
    CIT.CIT_CITY_%LanguageID% as CIT_CITY,  
    CIT.CIT_CITY_LOCAL,  
    CIT.CIT_CITY_INT,  
    CCM.CCM_VALID_FROM,  
    CCM.CCM_VALID_UP_TO  
from CCM, CIT  
where CCM.COU_IDX = :COU_IDX and  
       CCM.CIT_IDX = CIT.CIT_IDX and  
       CIT.CIT_CITY_%LanguageID% <> '?'  
order by CIT.CIT_CITY_%LanguageID%
```

In den genannten Sektionen wird ein Dataset DS1 definiert, das dem Report-Bereich Band1 zugeordnet wird und dessen Daten vom Provider DemoDB gemäß dem SQL-Statement bereitgestellt werden. Im Beispiel enthält DS1 alle Städte der selektierten Länder (:COU\_IDX).

---

Es können selbstverständlich mehrere Datasets für einen Report definiert werden. Nicht definiert werden muss das Dataset SELECTION. Dieses steht dem Report standardmäßig zur Verfügung und enthält die Selektionsdatenmenge.

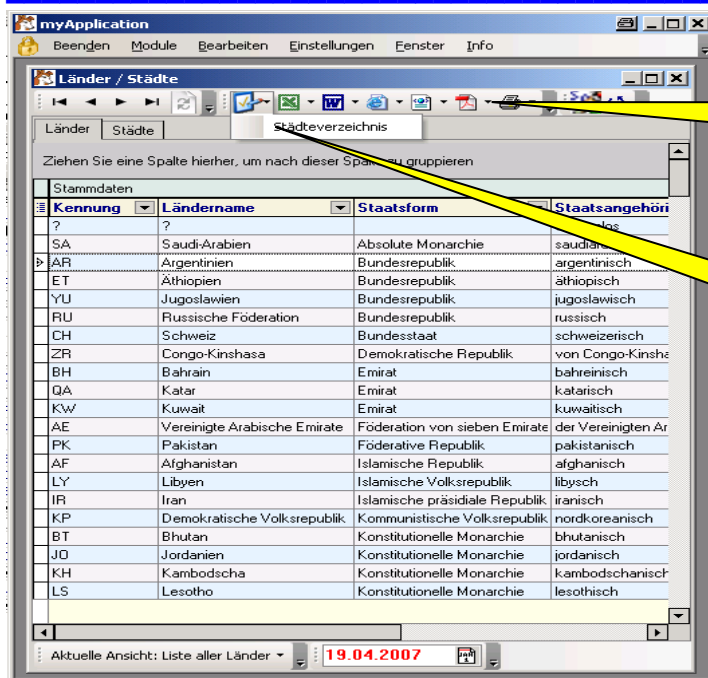
Bevor wir nun die Anwendung neu starten um den FastReport®-Designer aufzurufen, müssen wir noch die Datei *myReport1.fr3* aus dem Template-Verzeichnis Roles\Admin\Reports in das gleichnamige Verzeichnis unserer Anwendung kopieren und in *Rep\_Cities.fr3* umbenennen. *Rep\_Cities.fr3* enthält die Report-Definitionen im XML-Format, die wir anschließend mit dem FastReport®-Designer bearbeiten werden.

Nicht zu vergessen sind natürlich auch die Sprachdefinitionen für unseren Report *Rep\_Cities*, z. B. in der Sprachdatei *de.cdd*.

**[Rep\_Cities]**

```
Caption=Städteverzeichnis  
Header.Caption=Städteverzeichnis  
Var_Date.Caption=Auswertung vom  
city.caption=Stadt  
valid_from.Caption=gültig ab  
valid_up_to.Caption=gültig bis  
city_local.Caption=lokaler Name  
city_int.Caption=internationaler Name
```

Nach dem Aufruf des FastReport®-Designers (Abb. 18) für unseren Report Städteverzeichnis können wir dann unseren Report gestalten (Abb. 19).



Zusätzliches PDF-Icon  
Die ▼-Taste deutet an, dass Reports zur Verfügung stehen

FastReport-Designer für den Report Städteverzeichnis aufrufen.

Abb. 18

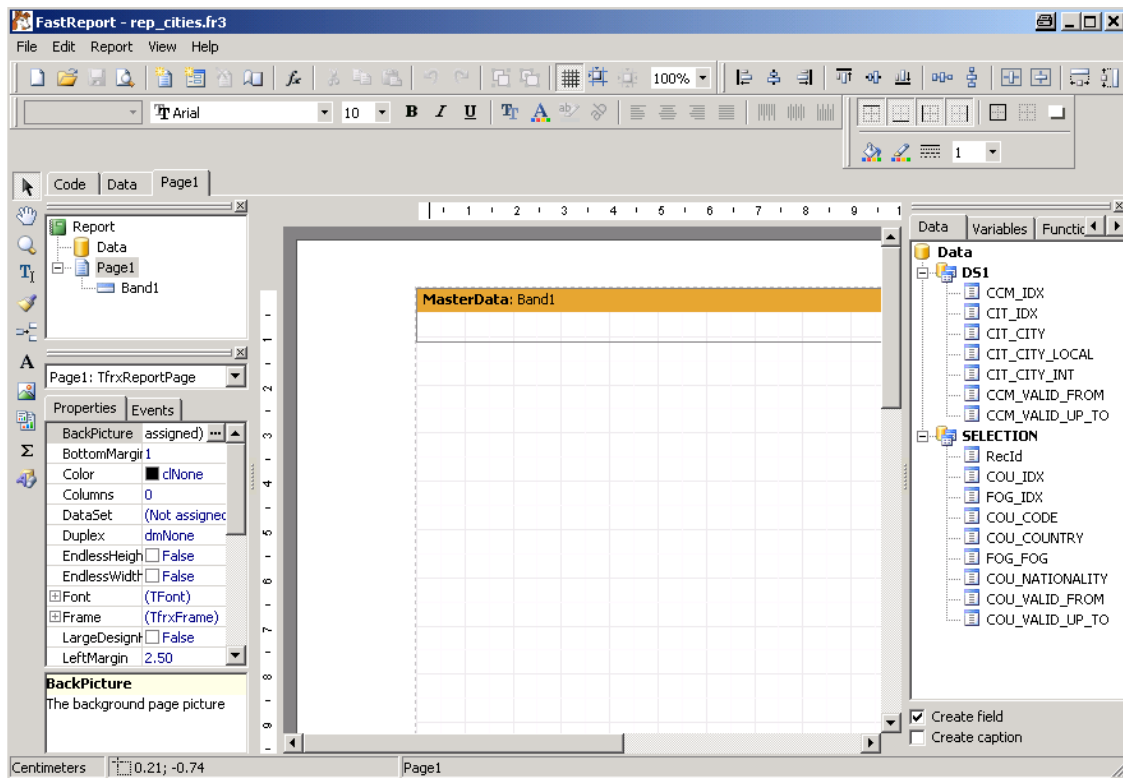


Abb. 19

Alle Möglichkeiten, die im FastReport®-Designer zur Verfügung stehen, sind in der FastReport®-Dokumentation (<http://fast-report.com/>) aufgeführt. Für unser Beispiel legen wir noch drei weitere Bereiche (PageHeader, GroupHeader, PageFooter) an, ziehen die gewünschten Datenfelder der beiden Datasets in die jeweiligen Bereiche und fügen ein Logo und einige System-Variablen (Date, Page#) hinzu (Abb. 20).

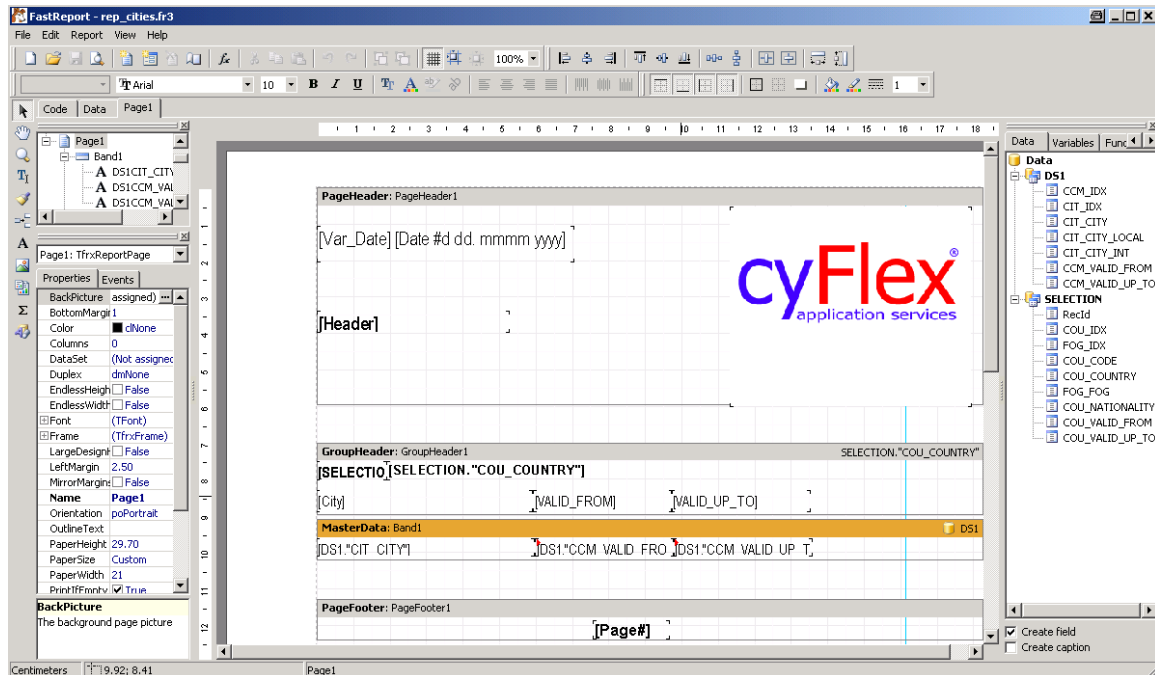


Abb. 20

Mit der Preview-Funktion können wir das Resultat unserer Einstellungen sofort kontrollieren (Abb. 21) und abschließend in die oben bereits erwähnte Datei *Rep\_Cities.fr3* sichern.



Abb. 21

Nach dem Neustart der Anwendung steht der Report dann für die Dokumenttypen zur Verfügung.

**Hinweis:** Für die Nutzung des FastReport®-Designer in cySystem ist eine entsprechende Lizenz erforderlich (FASTREPORT® 4.0 VCL - report generator for Delphi)!

## 5.5. Selektionsautomaten und SpecialBands

In Kapitel 5.1 haben wir bereits Bänder kennengelernt. Mit ihrer Hilfe können wir mehrere Felder in einem Selektionsautomaten unter einer zusätzlichen gemeinsamen Überschrift gruppieren und ihre Anordnung mehrzeilig gestalten. Beim Einsatz von Bändern gibt es nun noch die Besonderheit, dass eines der Bänder vom Typ SpecialBand sein kann.

SpecialBands sind speziell vordefinierte Bänder zur Darstellung von Kalendern. Die folgende Abbildung zeigt einen Ausschnitt aus einem Schichtplan, in dem jeder Tag eines Monats in einer Spalte dargestellt wird. Während den Überschriften der Felder die Werte 01 bis maximal 31 zugeordnet wurden, enthält das SpecialBand pro Tag die Tageskurzbezeichnung.

	Mi.	Do.	Fr.	Sa.	So.	Mo.	Di.	Mi.	Do.	Fr.	Sa.	So.	Mo.
	01	02	03	04	05	06	07	08	09	10	11	12	13
	wFBFC	wFBFC	wFBFC			wFBFC	wFBFC	wFBFC	wFBFC	wFBFC			wFBFC
	wFBFC	wFBFC	wFBFC			wFBFC	wFBFC	wFBFC	wFBFC	wFBFC			wFBFC
	RA306		RF305	RA305	RA306	RF324	RF508		RF324	RF284	KaAU		RF324
	RP245	RP245	RP245	RP245		RP245	RP245	RP245	RP245	RP245		RP882	RP882
	GB246		GB224			GB609	GB528	GB609	GB689	GB609	GB689		GB688
	GF286	GF810	GF760	GF590	F	GF366	GF286	GF286		GF639	GF599	GF810	GB528
	RP245	RP245	RP245			RP245	RP245	RP245	RP245	RP245			RP245
	FB346	FB367				FB366	FB488	FB266	FB507		FB366		FB386
	GF366	GF366	GF639	GF590		GF366	GF286	GF286		GF639	GF810	GF599	GB708
	FB507		FB224	FB204	FB366	FB548		FB204	FB204	FB366			FB326
	RP245	RP245	RP245			RP882	RP882	RP882	RP882	RP882			RP245

Abb. 22

Für dieses Beispiel sind in der Definition der View VIEW\_Schichtplan zwei Bänder aufzunehmen, die wie folgt zu definieren sind:

### [View\_Schichtplan]

Bands=View\_Schichtplan\_Band1|View\_Schichtplan\_Band2

...

### [View\_Schichtplan\_Band1]

HeaderAlignmentHorz=taLeftJustify

Feld1=0

Feld2=0

```
...  
[View_Schichtplan_Band2]  
SpecialBand=fbSDOM  
Enum=SCH_**  
HeaderAlignmentHorz=taCenter
```

In der Definition für das SpecialBand steht der Typ fbSDOM für "Short Days of Month". Alternativ wäre auch fbLDOM für „Long Days of Month“ möglich. Weitere (Kurz-) Bezeichnungen für Wochen und Monate sind in Planung.

Den Tagen oder Monaten können Styles zugeordnet werden. Dies geschieht in der Design Datei *\_Design.cdd* (s.u.) in den Sektionen [DaysOfWeek] oder [MonthsOfYear].

Die Funktion Enum ermöglicht eine automatische Zuordnung von Feldern zum SpecialBand. Sie ist aber nur bei Feldern anwendbar, die in ihrem Namen eine fortlaufende Nummer führen (im Beispiel SCH\_01, SCH\_02...SCH\_31). Andernfalls sind die Felder wie im ersten Band einzeln zuzuordnen.

## 5.6. Referenz-Selektionsautomaten

Referenz-Selektionsautomaten werden in Pflegeautomaten zur Auswahl von gültigen Feldinhalten bei den sogenannten Referenzfeldern benötigt (siehe Kapitel 6.1.2). Ein Eingabefeld bezieht seinen Wert also durch eine Referenz auf ein bestimmtes Feld einer anderen Tabelle, im Folgenden auch Referenztabelle genannt.

Obwohl erst im nächsten Kapitel benötigt, werden wir den Referenz-Selektionsautomaten bereits an dieser Stelle behandeln, da es sich um eine besondere Form eines Selektionsautomaten handelt. Bezüglich seiner Definition entspricht der Referenz-Selektionsautomat dem oben bereits vorgestellten Standard-Selektionsautomaten. Er enthält in der Regel nur eine View und wird im Pflegeautomaten in der Action-Sektion seines Referenzfeldes als RefForm aufgerufen. Das folgende Beispiel zeigt den Aufruf der Städteliste, der beim Klick auf das Referenzfeld Stadt erfolgen soll.

```
[cyPAM_Cities.Action]  
OnClick.CIT_CITIES=RefForm.Ref_SAM_Cities
```

Durch den RefForm-Aufruf, anstelle von Form, wird der Selektionsautomat in einem modalen Fenster geöffnet und erhält automatisch die beiden zusätzlichen Buttons (OK und Abbrechen) zur Bestätigung der getroffenen Auswahl.

Zur Anlage eines Referenz-Selektionsautomaten befinden sich in unserem Template die beiden Dateien *Tpl\_Ref\_SAM.cdd*. Diese sind wie bereits in Kapitel 5.1 beschrieben zu kopieren und zu überarbeiten. Die Definitionsdateien für die Anzeige der Städteliste sehen wie folgt aus:

- 
- *Ref\_SAM\_Cities.cdd* im Def-Files-Verzeichnis

**[Ref\_SAM\_Cities]**

```
Top=50
Left=100
Width=350
Height=400
SizeConstraints=True
```

**[Ref\_SAM\_Cities.Link]**

```
cySAM_Cities=TcySAM
```

**[cySAM\_Cities]**

```
ActiveStyleSheetName=reppredef_Referenz
Grid=Grid_Cities
```

Um den Referenz-Selektionsautomaten farblich abzuheben, verwenden wir ein anderes Design (\_Referenz).

- *Ref\_SAM\_Cities.cdd* im Roles-Verzeichnis

**[Grid\_Cities]**

```
RootLevelOptions.DetailTabsPosition=dtbNone
Views=View_Cities
```

**[View\_Cities]**

```
Provider=DemoDB
IndexName=CIT_IDX
DateFormat=YYYYMMDD
TimeFormat=HHNNSS
VisibleFields=CIT_CITY
HiddenFields=CIT_IDX
Filters=Filter0
```

**[View\_Cities.SQL]**

```
select
  CIT.CIT_IDX,
  CIT.CIT_CITY_%LanguageID% as CIT_CITY,
  CIT.CIT_CITY_LOCAL,
  CIT.CIT_VALID_FROM,
  CIT.CIT_VALID_UP_TO
```

```
from CIT
order by CIT.CITY_ %LanguageID%
```

**[View\_Cities.Filter0]**

**[View\_Cities.ACTION]**

OnDBLClick=Form.Frm\_PAM\_Cities

Die Action-Sektion enthält den Aufruf eines Pflegeautomaten, um die Städteliste direkt aus dem Referenz-Selektionsautomaten heraus bearbeiten zu können.

Das Ergebnis zeigt der folgende Screenshot in Abb. 23:

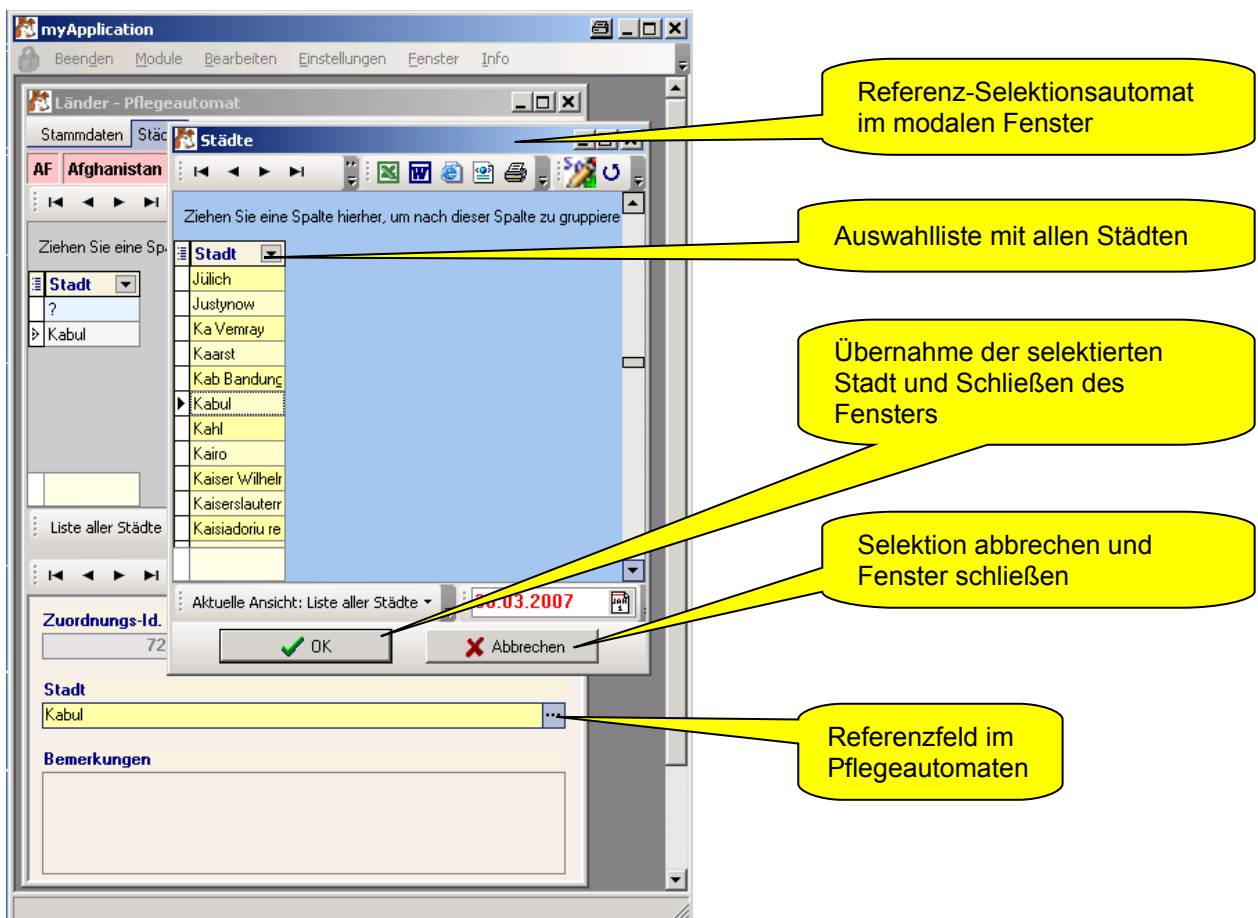


Abb. 23

---

## 6. Pflegeautomaten

Nachdem nun ausführlich die verschiedenen Formen von Selektionsautomaten vorgestellt wurden, wollen wir uns den Pflegeautomaten zuwenden. Damit wir endlich Daten in unserer Beispielanwendung pflegen können, werden wir verschiedene Pflegeautomaten wiederum mit Hilfe von Templates entwickeln.


### 6.1. Standard-Pflegeautomaten

In unserem Selektionsautomaten *Frm\_SAM\_Countries\_Cities.cdd* haben wir bereits den Aufruf eines Pflegeautomaten zur Pflege der Länderdaten vorgesehen:

```
[View_countries.ACTION]
OnDBLClick=Form.Frm_PAM_Countries
```

Es fehlt also noch das Formular *Frm\_PAM\_Countries.cdd* mit einem entsprechenden Pflegeautomaten zur Pflege der Länderdaten. Der folgende Screenshot in Abb. 24 zeigt vorab, wie der Länder-Pflegeautomat aussehen wird und aus welchen Komponenten er sich zusammensetzt.

Die Navigationsbar enthält neben den Navigationsfunktionen (z. B. Wechseln zum nächsten Datensatz) auch die Update-Funktionen (Ändern, Löschen, Einfügen, Kopieren).

Bei den Datenfeldern unterscheiden wir neben den unterschiedlichen Feldtypen (z. B. Datums-, Text-, numerische Felder) noch, ob die Felder eingabebereit sind, ob sie nur angezeigt werden (grauer Hintergrund) oder ob Eingaben erforderlich sind (Mussfeld, gelber Hintergrund). Referenzfelder sind nicht direkt editierbar, sondern erlauben nur die Selektion eines Wertes aus einer sogenannten Referenztabelle. Nach Anklicken des Icons  öffnet sich ein Referenz-Selektionsautomat, der die für dieses Feld zulässigen Werte zur Auswahl anbietet.

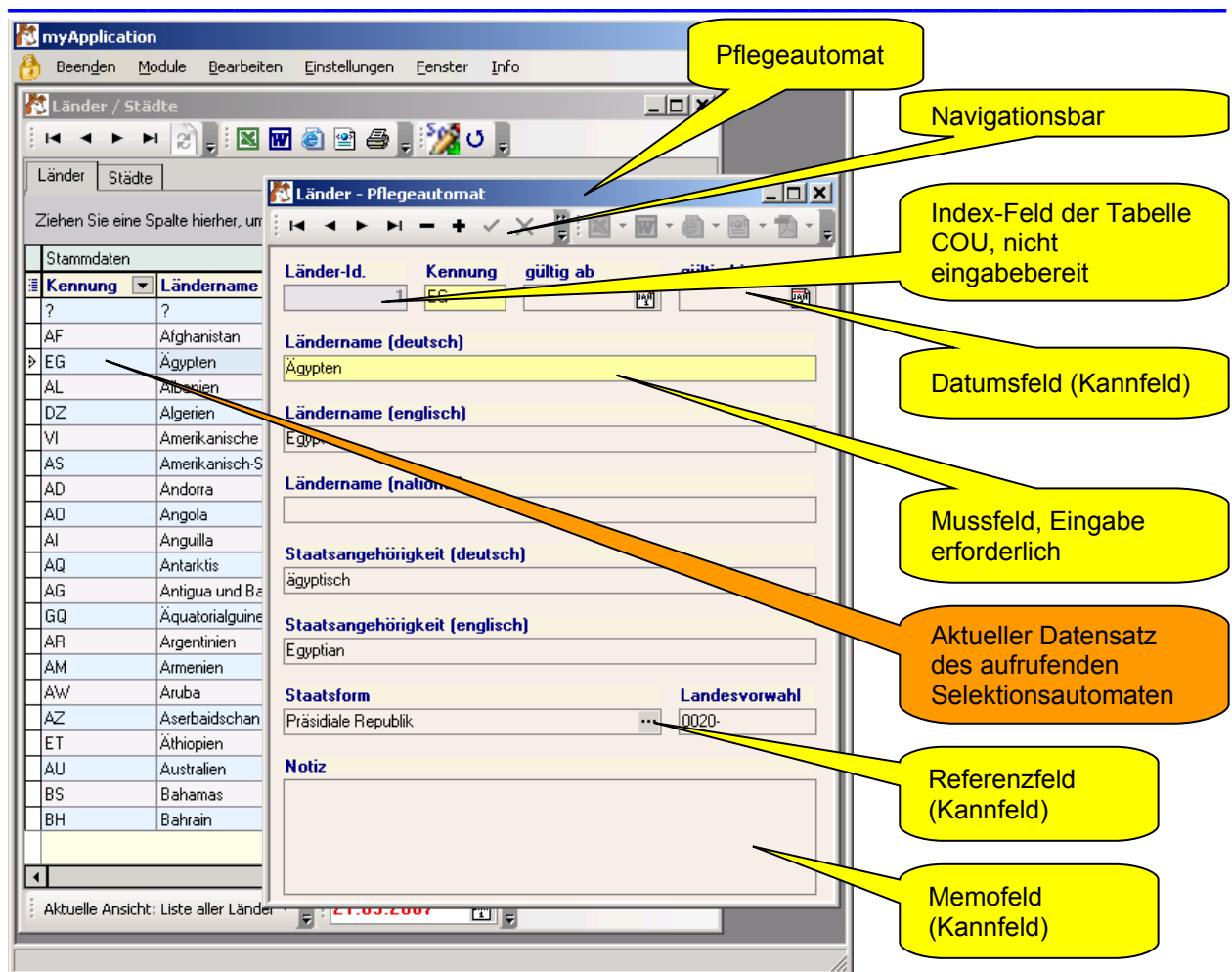


Abb. 24

### 6.1.1. Die Templatedatei Def-Files\Tpl\_Frm\_PAM.cdd

Zur Definition des oben abgebildeten Pflegeautomaten, kopieren wir nun aus dem Def-Files-Verzeichnis von unserem Template die Datei *Tpl\_Frm\_PAM.cdd* in das Def-Files-Verzeichnis der Anwendung und benennen es in *Frm\_PAM\_Countries.cdd* um. Innerhalb der Datei werden danach die Zeichen `##_` durch `_Countries` ersetzt, wodurch sich folgende Sektionen ergeben:

```
[Frm_PAM_Countries]
Top=70
Left=100
Width=410
Height=520
SizeConstraints=True
```

---

```
[Frm_PAM_Countries.Link]
```

```
cyPAM_Countries=TcyPAM
```

```
[cyPAM_Countries]
```

```
;Deny=Update|Delete|Insert
```

```
RefreshMasterView=True
```

```
Provider=DemoDB
```

```
DateFormat=YYYYMMDD
```

```
TimeFormat=HHNNSS
```

```
ApplyTable=COU
```

```
IndexName=COU_IDX
```

Zusätzlich wurde in der cyPAM-Countries-Sektion der Provider gesetzt, die Apply Table und der Name des Indexfeldes dieser Tabelle definiert. Unter Apply Table wird der Tabellename der zu pflegenden Datenbanktabelle angegeben.

Der auskommentierte Parameter Deny kann dazu verwandt werden, einzelne Änderungsfunktionen des Pflegeautomaten zu unterdrücken. Soll die Funktion Deny rollenabhängig eingesetzt werden, wird eine Sektion cyPAM-Countries mit dem Deny-Parameter in den namensgleichen Definitionsdateien der Rollen-Verzeichnisse aufgenommen (siehe 5.1.2).

Standardmäßig wird nach einer Datenbankänderung des Pflegeautomaten auch seine Master-View aktualisiert. Dabei verstehen wir unter Master-View die View des Selektionsautomaten, aus der der Pflegeautomat aufgerufen wurde. Werden mit einem Pflegeautomaten viele Datenbankänderungen hintereinander durchgeführt, kann es aus Performance-Gründen sinnvoll sein den Refresh der View mit Hilfe des Parameters RefreshMasterView auszuschalten. Der Anwender muß dann den Refresh nach Verlassen des Pflegeautomaten manuell durchführen. Über einen Button in der Toolbox des Pflegeautomaten kann der Anwender den automatischen Refresh der Master-View auch eigenständig ein- bzw. ausschalten.

### 6.1.2. Die Templatedatei Roles\Admin\Tpl\_Frm\_PAM.cdd

Aus dem Roles\Admin-Verzeichnis des Templates kopieren nun die Datei *Tpl\_Frm\_PAM.cdd* in das Roles\Admin-Verzeichnis der Anwendung und benennen es in *Frm\_PAM\_Countries.cdd* um. Wie gehabt ersetzen wir *##\_* in der Datei durch *\_Countries*. Die erste, auskommentierte Sektion cyPAM-Countries wird nur benötigt, wenn mit Hilfe des Deny-Parameters (s.o.) Änderungsfunktionen rollenbezogen deaktiviert werden sollen.

```
; [cyPAM_Countries]
```

```
;Deny=Update|Delete|Insert
```

Die nächste Sektion dient zur Bestimmung der Selektionsdatenmenge, also der Daten, die zu unserem Pflegeautomaten gehören.

**[cyPAM\_Countries.SQL]**

```
Select COU.COU_IDX,  
       COU.COU_CODE,  
       COU.COU_VALID_FROM,  
       COU.COU_VALID_UP_TO,  
       COU.COU_COUNTRY_0,  
       COU.COU_COUNTRY_1,  
       COU.COU_COUNTRY_LOCAL,  
       COU.COU_NATIONALITY_0,  
       COU.COU_NATIONALITY_1,  
       COU.FOG_IDX,  
       FOG.FOG_FOG_%LanguageID% as FOG_FOG,  
       COU.COU_DIALCODE,  
       COU.COU_REMARK,  
       COU.COU_MASTER,  
       COU.COU_MASTERTIMESTAMP,  
       COU.COU_USER,  
       COU.COU_TIMESTAMP  
from COU left join FOG on COU.FOG_IDX = FOG.FOG_IDX  
where COU.COU_IDX = :COU_IDX
```

Diese SQL-Abfrage enthält alle Felder, die im Pflegeautomaten dargestellt werden sollen, sowie alle Felder, die für den Datenbank-Update (alle Felder der Apply Table COU) benötigt werden, insbesondere die Schlüssel von Referenzfeldern (im Beispiel FOG\_IDX). Die Where-Clause schränkt die Selektion mit Hilfe des Parameters :COU\_IDX auf genau den Datensatz ein, der zum zuvor im Selektionsautomaten selektierten Land gehört. Die Reihenfolge der Felder im Select-Statement bestimmt auch die Tabulatorreihenfolge im Pflegeautomaten (die Ansteuerung des nächsten Feldes per Tabulator).

Den letzten vier Feldern COU\_Master bis COU\_Timestamp kommt eine besondere Bedeutung zu. Sind diese Felder wie im Beispiel in der Apply Table vorhanden, besteht die Möglichkeit, mit ihrer Hilfe Datensatzänderungen zu protokollieren (Details siehe unten).

Nun folgt etwas Fleißarbeit, da wir jedes Feld unserer Selektionsdatenmenge in einer eigenen Sektion definieren müssen. Beginnen wir mit dem Indexfeld der Ländertabelle:

**[cyPAM\_Countries.COU\_IDX]**

```
Enabled=False  
LabelAlign=Left  
VAlign=Top  
FieldTop=10  
FieldLeft=10  
FieldWidth=90  
Fieldtypes=isIndexField, isNotNullField
```

---

```
PanelAnchors=akLeft, akTop
```

Die meisten der genannten Parameter sind selbsterklärend. Da COU\_IDX nicht enabled ist, handelt es sich um ein Anzeigefeld, dessen Überschrift links ausgerichtet (LabelAlign) und dessen Feldinhalt (VAlign) vertikal nach oben (Top, alternativ Bottom oder Center) ausgerichtet ist. Eine horizontale Ausrichtung geschieht automatisch in Abhängigkeit vom Feldinhalt (Right bei numerischen-, Left bei alphanumerischen Inhalten). Die Parameter FieldTop, FieldLeft, FieldWidth und FieldHeight bestimmen die Größe und Positionierung des Feldes auf dem Formular. Der Parameter FieldTypes charakterisiert das Feld als Index- und Mussfeld. Unser Schlüsselfeld der Ländertabelle darf natürlich nicht leer sein, wobei sein eindeutiger Inhalt vom Datenbanksystem automatisch gesetzt wird und nicht vom Anwender einzugeben ist.

Die PanelAnchors akLeft, akRight, akTop und akBottom verankern ein Feld mit dem übergeordneten Element, in unserem Fall dem Formular. Verändert der Anwender die Formulargröße, wirkt sich die Änderung je nach Wahl des PanelAnchors auch auf die Feldgröße aus. Bei Datumsfeldern (z. B. COU\_VALID\_FROM) sollte auf akRight verzichtet werden, da es hier schnell zu Überlagerungen kommen kann, wenn das Formular vergrößert und anschließend wieder verkleinert wird.

Bei den folgenden Feldern taucht Format als zusätzlicher Parameter auf. Er wird in erster Linie benötigt bei numerischen Feldern mit Gleitkomma-Arithmetik (Darstellung, Nachkommastellen) und Datums- bzw. Zeitfeldern, die im Charakterformat in der Datenbank gespeichert sind. Für die Definition von Nachkommastellen ist zusätzlich der Parameter Decimals nötig.

```
[cyPAM_Countries.COU_CODE]
```

```
Enabled=True  
LabelAlign=Left  
VAlign=Top  
FieldTop=10  
FieldLeft=110  
FieldWidth=60  
FieldTypes=isNotNullField  
PanelAnchors=akLeft, akTop
```

```
[cyPAM_Countries.COU_VALID_FROM]
```

```
Format=fsd  
Enabled=True  
LabelAlign=Left  
VAlign=Top  
FieldTop=10  
FieldLeft=180  
FieldWidth=100
```

---

PanelAnchors=akLeft, akTop

**[cyPAM\_Countries.COU\_VALID\_UP\_TO]**

format=fsd  
Enabled=True  
LabelAlign=Left  
VAlign=top  
FieldTop=10  
FieldLeft=290  
FieldWidth=100  
PanelAnchors=akTop, akLeft

**[cyPAM\_Countries.COU\_COUNTRY\_0]**

Enabled=True  
LabelAlign=Left  
VAlign=Top  
FieldTop=60  
FieldLeft=10  
FieldWidth=380  
Fieldtypes=isNotNullField  
PanelAnchors=akLeft, akTop, akRight

**[cyPAM\_Countries.COU\_COUNTRY\_1]**

Enabled=True  
LabelAlign=Left  
VAlign=Top  
FieldTop=110  
FieldLeft=10  
FieldWidth=380  
PanelAnchors=akLeft, akTop, akRight

**[cyPAM\_Countries.COU\_COUNTRY\_LOCAL]**

Enabled=True  
LabelAlign=Left  
VAlign=Top  
FieldTop=160  
FieldLeft=10  
FieldWidth=380  
PanelAnchors=akLeft, akTop, akRight

**[cyPAM\_Countries.COU\_NATIONALITY\_0]**

Enabled=True  
LabelAlign=Left  
VAlign=Top

---

```
FieldTop=210
FieldLeft=10
FieldWidth=380
PanelAnchors=akLeft, akTop, akRight
```

```
[cyPAM_Countries.COU_NATIONALITY_1]
Enabled=True
LabelAlign=Left
VAlign=Top
FieldTop=260
FieldLeft=10
FieldWidth=380
PanelAnchors=akLeft, akTop, akRight
```

Unser nächstes Feld, das Referenzfeld Staatsform (FOG\_FOG) erfordert die Definition von drei zusätzlichen Parametern.

```
[cyPAM_Countries.FOG_FOG]
Enabled=True
LabelAlign=Left
VAlign=Top
FieldTop=310
FieldLeft=10
FieldWidth=270
ApplyTarget=FOG_IDX
ReferenceSelection=FOG_FOG
ReferenceTarget=FOG_IDX
PanelAnchors=akLeft, akTop
```

Ein Referenzfeld ist ein Feld eines Pflegeautomaten, das seinen Wert aus einer Auswahltabelle (Referenztabelle) erhält. Für unser Beispiel bedeutet das, dass wir in der Ländertabelle nicht die Staatsform selbst speichern, sondern den eindeutigen Fremdschlüssel (Referenzschlüssel) der Staatsform (FOG\_IDX) aus der Tabelle FOG. Im Pflegeautomat anzeigen wollen wir natürlich nicht den Fremdschlüssel, sondern die Bezeichnung der Staatsform (FOG\_FOG). Diese Technik stellt sicher, dass bei derartigen Attributen keine inhaltlichen Abweichungen (durch unterschiedliche Schreibweisen, Tippfehler) auftreten können und sie damit auswertbar bleiben.


Zum Verständnis der drei neuen Parameter gilt es zunächst zu berücksichtigen, dass wir es mit drei unterschiedlichen Datenmengen zu tun haben:

- Die Applydatenmenge enthält **alle** Felder der zu pflegenden Tabelle (COU).

- 
- Die Referenzdatenmenge enthält die Felder der Referenztable (FOG).
  - Der Selektionsdatenmenge enthält die Felder, die im Pflegeautomat definiert werden (die unter cyPAM\_Countries.SQL definierte View).

Zu beachten ist dabei, dass die Felder in der jeweiligen Datenmenge unter einem anderen Namen geführt werden können (im Beispiel sind die Namen gleich). Um nun die Felder der drei Datenmengen richtig zu verknüpfen, benötigen wir die oben bereits angesprochenen Parameter:

- ApplyTarget bestimmt den (Alias-) Namen des Datenbankfeldes aus der Selektionsdatenmenge, das den Referenzschlüssel (FOG\_IDX) speichert.
- ReferenceSelection gibt den (Alias-) Namen des Feldes der Referenzdatenmenge an, dessen Inhalt im Pflegeautomat angezeigt werden soll, also unser aktuelles Feld (FOG\_FOG).
- ReferenceTarget bestimmt den (Alias-) Namen des Feldes der Referenzdatenmenge, das in das zugehörige Feld der Applydatenmenge übertragen werden muss.

Damit hat der cyAgent nun die nötigen Informationen zur Frage, welche Felder angezeigt und welche im Hintergrund gespeichert werden sollen. Es fehlt noch die Angabe des Referenz-Selektionsautomaten (siehe Kapitel 5.6), der beim Anklicken des Icons  die Referenztable anzeigen soll. Dies geschieht in der folgenden Sektion:

**[cyPAM\_FOG\_FOG.Action]**

```
OnClick.fieldname2=RefForm.Ref_SAM_FOG
```

Ist die Referenzdatenmenge nicht zu groß, können wir die Fremdschlüssel auch direkt im Pflegeautomaten in einer Selektion-Box zur Verfügung stellen. Mit Hilfe der Parameter Format=fSelect und Data=SQL sowie der folgenden SQL-Anweisung füllen wir die Werte der Selektion-Box. Den über Union hinzugefügten leeren Eintrag benötigen wir nur bei Kann-Feldern.

**[cyPAM\_Countries.FOG\_IDX]**

```
Enabled=True  
LabelAlign=Left  
VAlign=Top  
FieldTop=310  
FieldLeft=10  
FieldWidth=270
```

---

```
PanelAnchors=akLeft, akTop  
Format=fSelect  
Data=SQL
```

```
[cyPAM_Countries.FOG_IDX.SQL]  
select  
    FOG.FOG_IDX as Target,  
    FOG.FOG_FOG_%LanguageID% as Selection  
from FOG  
union  
select  
Null as Target,  
Null as Selection  
from FOG  
where FOG.FOG_IDX = 1  
order by Selection
```

Beim Einsatz einer Selektion-Box anstelle eines Referenz-Selektionsautomaten können wir natürlich auch auf den Join mit der Tabelle FOG in unserer SQL-Abfrage für unseren Pflegeautomaten verzichten. Die folgende Abbildung zeigt den Pflegeautomaten für Länder mit einer Selektion-Box für die Staatsform.

The screenshot shows a software window titled 'Länder - Pflegeautomat'. It has a menu bar with 'Stammdaten', 'Städte', 'Grafiken', 'Währungen', and 'Sprachen'. Below the menu, there is a tab for 'EG Ägypten'. A toolbar with various icons is visible. The main area contains several input fields and a dropdown menu:

- Länder-Id.:** 1
- Kennung:** EG
- gültig ab:** [calendar icon]
- gültig bis:** [calendar icon]
- Ländername (deutsch):** Ägypten
- Ländername (englisch):** Egypt
- Ländername (national):** [empty field]
- Staatsangehörigkeit (deutsch):** ägyptisch
- Staatsangehörigkeit (englisch):** Egyptian
- Staatsform:** A dropdown menu with 'Präsidentiale Republik' selected. Other options include 'Parlamentarische Monarchie im Commonwealth', 'Parlamentarische Republik', 'Parlamentarische Republik im Commonwealth', 'Parlamentarisches Fürstentum', 'Präsidentiale Bundesrepublik', 'Präsidentiale föderative Republik', and 'Präsidentiale Republik im Commonwealth'.
- Landesvorwahl:** 0020-

At the bottom, there are small icons for 'CAPS', 'NUM', 'SCRL', 'INS', and a status indicator 'Status: Normal'.

Abb. 25

Beim letzten Feld, dem Bemerkungsfeld, erscheint noch einmal der Format-Parameter, um das Feld als Memo-Feld zu definieren.

```
[cyPAM_Countries.COU_DIALCODE]
Enabled=True
LabelAlign=Left
VAlign=Top
FieldTop=310
FieldLeft=290
```

---

```
FieldWidth=100
PanelAnchors=akLeft, akTop, akRight

[cyPAM_Countries.COU_REMARK]
Format=fMemo
Enabled=True
LabelAlign=Left
VAlign=Top
FieldTop=360
FieldLeft=10
FieldWidth=380
PanelAnchors=akLeft, akTop, akRight, akBottom
```

Den Parametern Visible, InsertFromMaster und EnableOnInsert, die im obigen Beispiel nicht benötigt wurden, werden wir uns erst im nächsten Kapitel widmen.

Ebenso wurden bisher keine Checkboxes eingesetzt. Checkboxes werden Hilfe der folgenden Parameter definiert:

- Checked=<Wert in der Datenbank z. B. „J“ >
- Unchecked=<Wert in der Datenbank z. B. „N „>
- Undefined=null (oder ein weiterer Datenbank-Wert)

Die zur Checkbox gehörenden Texte werden in den Sprachdateien unter der Sektion des Pflegeautomaten gepflegt:

- <Feldname>.CAPTIONCHECKED=Ja
- <Feldname>.CAPTIONUNCHECKED=Nein
- <Feldname>.CAPTIONUNDEFINED=nicht definiert

Abschließen wollen wir unseren ersten Pflegeautomaten mit den folgenden vier internen Feldern:

```
[cyPAM_Countries.COU_MASTER]
FieldTypes=isModifierField
ModifierType=mtCreator

[cyPAM_Countries.COU_MASTERTIMESTAMP]
FieldTypes=isModifierField
ModifierType=mtCreatedOn
```

---

```
Format=fsdst  
[cyPAM_Countries.COU_USER]  
FieldTypes=isModifierField  
ModifierType=mtModifier
```

```
[cyPAM_Countries.COU_TIMESTAMP]  
FieldTypes=isModifierField  
ModifierType=mtModifiedOn  
Format=fsdst
```

Diese Felder werden nicht in unserem Pflegeautomaten angezeigt. In ihrer Definition finden wir den neuen Parameter ModifierType. Die beiden Felder vom Typ mtCreator und mtCreatorOn halten fest, welcher Anwender einen Datensatz neu angelegt hat und wann dieser Datensatz zum erstenmal angelegt wurde. Die letzten beiden Felder geben Auskunft darüber, wer und wann einen Datensatz zuletzt geändert hat. Zu beachten ist, dass alle vier Felder als String-Felder (Typ Text) auf der Datenbank angelegt sein müssen.

## 6.2. Pflegeautomaten und Container

Das Container-Konzept wurde bereits im Zusammenhang mit Selektionsautomaten ausführlich vorgestellt. Container können natürlich nicht nur Selektionsautomaten, sondern auch Pflegeautomaten oder häufig auch beide Typen in unterschiedlichen Kombinationen aufnehmen.

Wir werden nun unseren Länder-Pflegeautomaten so erweitern, dass wir zusätzlich zur Pflege der Länder-Stammdaten auch noch Städte und Grafiken zuordnen können (siehe Abb. 26).

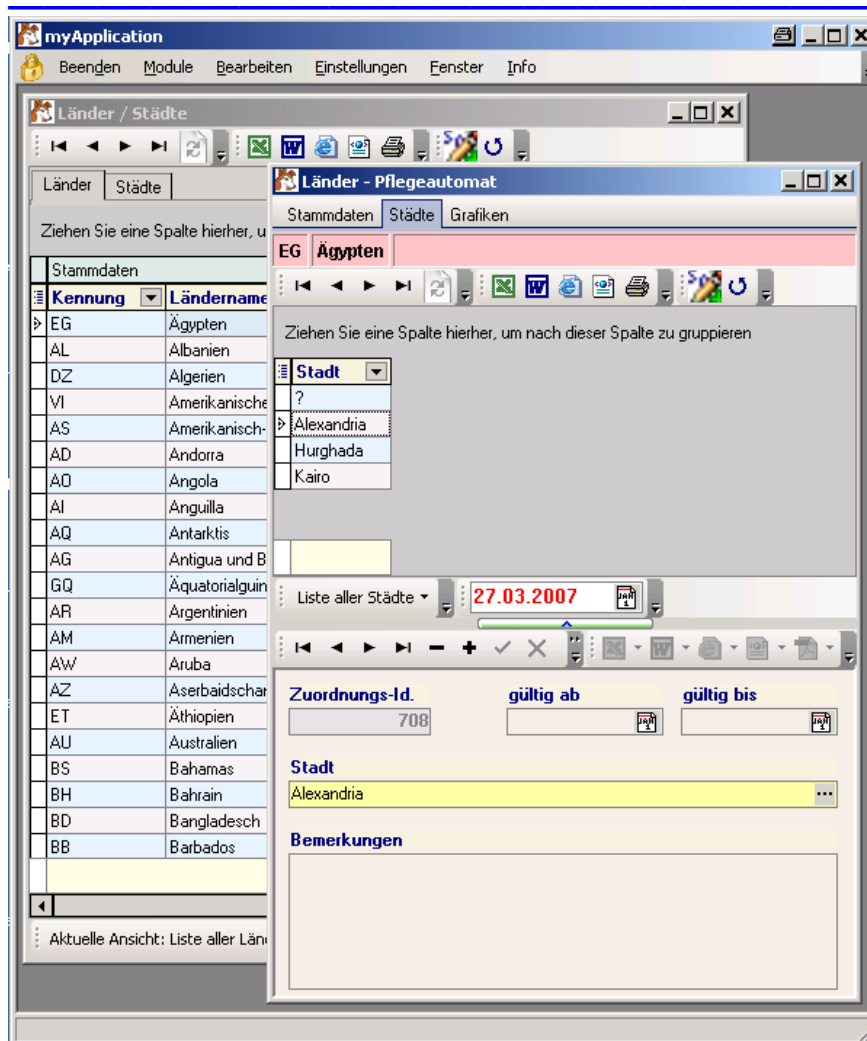


Abb. 26

### 6.2.1. Die Templatedatei *Def-Files\Tpl\_Frm\_PAM\_Container.cdd*

Dazu kopieren wir aus dem Def-Files-Verzeichnis von unserem Template die Datei *Tpl\_Frm\_PAM\_Container.cdd* in das Def-Files-Verzeichnis der Anwendung und nennen sie in *Frm\_PAM\_Countries.cdd* um bzw. ersetzen die bestehende Datei aus dem Beispiel 5.1.1. Innerhalb der Datei löschen wir alle Definitionen zu Beispiel 1. Dann ersetzen wir die Zeichenketten:

- `_##` durch `_Countries`
- `_#1` durch `_Countries`
- `_#2` durch `_Cities`

- `_#3` durch `_Graphics`
- `_#4` durch `_Cities`
- `_#5` durch `_Graphics`
- `Provider=??` durch `Provider=DemoDB`

In den cyPAM-Sektionen müssen wir nun noch die jeweiligen Indexnamen und Applytabellen definieren:

- cyPAM\_Countries: IndexName=COU\_IDX , ApplyTable=COU
- cyPAM\_Cities: IndexName=CCM\_IDX , ApplyTable=CCM
- cyPAM\_Graphics: IndexName=GRA\_IDX , ApplyTable=GRA

Abschließend setzen wir noch die Info-Fields in den Sektionen cyPAM\_Countries, cySAM\_Cities und cySAM\_Graphics auf COU\_CODE|COU\_COUNTRY. Damit werden in unserem Pflegeautomaten oberhalb der Tabreiter der aktuelle Länderschlüssel und das aktuelle Land angezeigt.

Unsere Definitionsdatei enthält nun folgende Sektionen:

```
[Frm_PAM_Countries]
Top=50
Left=50
Width=410
Height=580
SizeConstraints=True
Container=C1 |C2 | C3 | C4           Insgesamt vier Container

[Frm_PAM_Countries.C1]             Container C1 enthält die
Home=Form                          Tabreiter
Dimension=100

[Frm_PAM_Countries.C2]             Container C2 wird Tabreiter
Home=C1.First.Tab1                 1 zugeordnet
Dimension=100

[Frm_PAM_Countries.C3]             Container C3 wird Tabreiter
Home=C1.First.Tab2                 2 zugeordnet
Split=Horizontal
Dimension=50
```

---

<b>[Frm_PAM_Countries.C4]</b> Home=C1.First.Tab3 Split=Horizontal Dimension=50	Container C4 wird Tabreiter 3 zugeordnet
<b>[Frm_PAM_Countries.Link]</b>  cyPAM_Countries=TcyPAM  cySAM_Cities=TcySAM  cySAM_Graphics=TcySAM	Der Pflegeautomat enthält 3 direkte Links: PAM zur Pflege der Länderdaten SAM zur Anzeige der zugeordneten Städte SAM zur Anzeige der zugeordneten Grafiken PAM zur Pflege der Länderdaten wird im Container C2 angezeigt (Tab1)
<b>[cyPAM_Countries]</b> Container=C2.First Provider=DemoDB DateFormat=YYYYMMDD TimeFormat=HHNNSS InfoFields=COU_CODE COU_COUNTRY IndexName=COU_IDX ApplyTable=COU	
<b>[cySAM_Cities=TcySAM]</b> Container=C3.First InfoFields=COU_CODE COU_COUNTRY Grid=Grid_Cities	SAM zur Anzeige der zugeordneten Städte wird im ersten Bereich von Container C3 angezeigt (Tab2)
<b>[cySAM_Cities.Link]</b> cyPAM_Cities=TcyPAM	SAM enthält einen weiteren Link zur Pflege der Städtezuordnung
<b>[cyPAM_Cities]</b> Container=C3.Last Provider=DemoDB  ViewMaster=View_Cities DateFormat=YYYYMMDD TimeFormat=HHNNSS IndexName=CCM_IDX ApplyTable=CCM	PAM zur Pflege der Städte- zuordnung wird im letzten Bereich von Container C3 angezeigt (Tab3) zu welcher übergeordneten MasterView (SAM) gehört der zu pflegende Datensatz?
<b>[cySAM_Graphics=TcySAM]</b> Container=C4.First ActiveStyleSheetName=reppredef_S	SAM zur Anzeige der zugeordneten Grafiken wird im Container C4

---

---

tandard InfoFields=COU_CODE COU_COUNTRY Grid=Grid_Graphics	angezeigt (Tab3)
<b>[cySAM_Graphics.Link]</b> cyPAM_Graphics=TcyPAM	SAM enthält einen weiteren Link zur Zuordnung der Grafiken
<b>[cyPAM_Graphics]</b> Container=C4.Last Provider=DemoDB	PAM zur Pflege der Grafiken-Zuordnung wird im letzten Bereich von Container C4 angezeigt (Tab3)
ViewMaster=View_Graphics DateFormat=YYYYMMDD TimeFormat=HHNNSS IndexName=GRA_IDX ApplyTable=GRA	zu welcher übergeordneten MasterView (SAM) gehört der zu pflegende Datensatz?

### 6.2.2. Die Templatedatei *Roles\Admin\Tpl\_Frm\_PAM\_Container.cdd*

Für die rollenspezifische Definitionsdatei können wir die Datei aus dem Beispiel 5.1.1. übernehmen und um die Definitionen für die zusätzlichen Selektions- und Pflegeautomaten ergänzen oder wie gewohnt das Template *Tpl\_Frm\_PAM\_Container.cdd* kopieren und entsprechend überarbeiten.

Auf die vollständige Definition der drei Pflege- und zwei Selektionsautomaten wird an dieser Stelle verzichtet, da es sich größtenteils um Wiederholungen von bereits beschriebenen Definitionen handelt. Stattdessen werden wir uns im Folgenden nur noch mit den noch nicht erläuterten Definitionen beschäftigen.

Im Pflegeautomaten für die Zuordnung der Städte benötigen wir u. a. folgende Definition:

```
[cyPAM_Cities.COU_IDX]
InsertFromMaster=True
VISIBLE=False
Fieldtypes=isNotNullField
```

Der Länderschlüssel (COU\_IDX) ist ein Mussfeld in der Applydatenmenge, aber nicht unbedingt in der Selektionsdatenmenge unseres Pflegeautomaten enthalten (z. B. beim Insert). In dem übergeordneten Selektionsautomaten unseres Pflegeautomaten ist der Länderschlüssel aber sehr wohl enthalten. InsertFromMaster sorgt dann dafür, dass der benötigte Länderschlüssel aus dem Master-Selektionsautomaten in die Applydatenmenge übernommen wird. Durch VISIBLE=False wird die Anzeige des Feldes im Pflegeautomaten unterdrückt.

---

Für den Fall, dass ein Mussfeld beim Insert-Vorgang nicht automatisch von der Masterview bereitgestellt werden kann und es trotzdem bei Anzeige- /Update-Vorgängen verborgen bleiben soll, gibt es den Parameter EnableOnInsert=True. Damit wird dieses Feld nur beim Insert sichtbar und eingabebereit.

Der ebenfalls für die Länder-Städte-Zuordnung benötigte Städteschlüssel CIT\_IDX wird über die Definition des Referenzfeldes CIT\_CITY der Applydatenmenge zugewiesen:

```
[cyPAM_Countries_Cities.CIT_CITY]
Enabled=True
LabelAlign=Left
VAlign=Top
FieldTop=60
FieldLeft=10
FieldWidth=380
Fieldtypes=isNotNullField
ApplyTarget=CIT_IDX
ReferenceSelection=CIT_CITY
ReferenceTarget=CIT_IDX
PanelAnchors=akLeft, akTop, akRight
```

Die übrigen Definitionen können auch in der gleichnamigen Definitionsdatei der Beispielanwendung „Digitaler Atlas“ nachgeschaut werden. Zu den noch fehlenden Formularen für die Referenz-Selektionsautomaten sei auf Kapitel 5.6 verwiesen. Auf die fehlenden Einträge in den Sprachdateien werden wir erst weiter unten eingehen.

### 6.3. Pflegeautomaten und Clones

Mit Hilfe von Clones können Daten einer Applydatenmenge in mehreren Containern gepflegt werden. Wenn zum Beispiel der Platz in einem Container nicht für alle Daten ausreicht, sollen weitere Felder unter einem neuen Tabreiter gepflegt werden.

Um die Anwendung von Clones an einem Beispiel zu demonstrieren, erweitern wir das Beispiel aus Kapitel 6.2 dahingehend, dass wir das Feld Notiz der Ländertabelle (COU\_REMARK) unter einem eigenen Tabreiter Bemerkungen zur Bearbeitung zur Verfügung stellen.

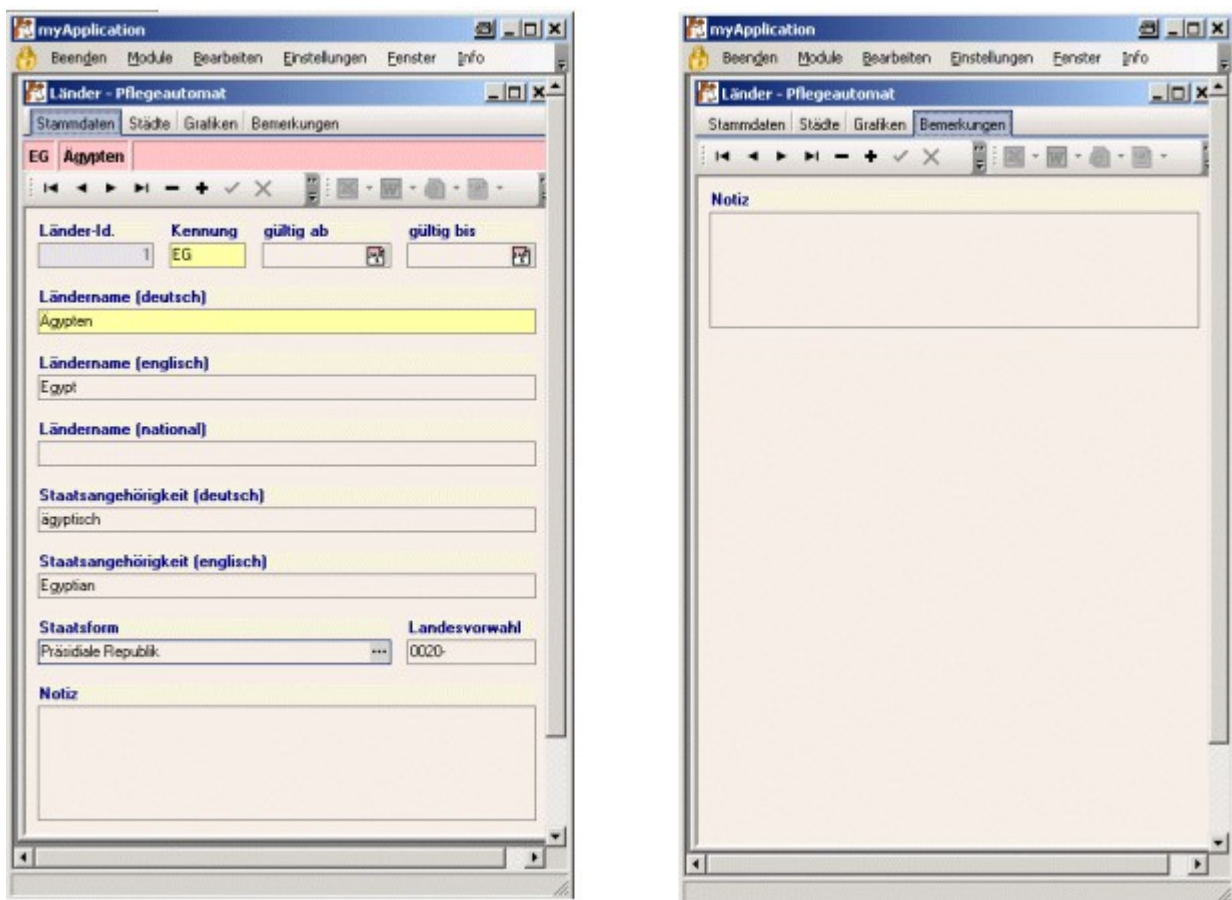


Abb. 27

In der Datei *Frm\_PAM\_Countries.cdd* des Def-Files-Verzeichnisses müssen wir dazu einen weiteren Container (C5) aufnehmen und den Pflegeautomaten *cyPAM\_Countries* mit dem neuen Clone verlinken, welcher wiederum dem neuen Container zugeordnet wird.

---

**[cyPAM\_Countries.Link]**

cyPAM\_Countries\_PAMClone=TcyPAM

**[cyPAM\_Countries\_PAMClone]**

Container=C5.First

Cloned=True

In der Datei *Frm\_PAM\_Countries.cdd* des Roles-Verzeichnisses werden dann unter dem Clone-Namen die Felder definiert. In unserem Beispiel ist es nur das eine Feld COU\_REMARK. Ein separates SQL-Statement existiert für den Clone nicht. Er bezieht sich immer auf die Selektion des verlinkten Pflegeautomaten (hier cyPAM\_Countries), d. h. in dessen View müssen auch alle Clone-Felder selektiert werden.

**[cyPAM\_Countries\_PAMClone.COU\_REMARK]**

format=fmemo

Enabled=True

LabelAlign=Left

VAlign=Top

FieldTop=10

FieldLeft=10

FieldWidth=380

PanelAnchors=akLeft, akTop, akRight, akBottom

---

## 6.4. Weitere Parameter für Pflegeautomaten

### 6.4.1. InsertOnEmpty

Öffnet man einen leeren Pflegeautomaten wird sein Status automatisch auf Insert gesetzt. Da die selektierte Applydatenmenge leer ist, weiss der cyAgent, dass er aus den Eingabedaten ein Insert-Befehl an die Datenbank absetzen muß.

Es gibt aber auch Fälle, in denen man einen Teil der benötigten Eingabedaten per SQL bereits initial vorgeben kann. Damit ist aber die selektierte Applydatenmenge nicht mehr leer und der Status des Pflegeautomaten wechselt auf Update, was natürlich anschließend zu einem Fehler führt. Um diesen Fehler zu vermeiden, setzen wir den Parameter InsertOnEmpty in der Sektion des Pflegeautomaten auf true.

InsertonEmpty sorgt dafür, dass anstelle eines Update-Befehls ein Insert-Befehl durchgeführt wird. Dabei werden alle PAM-Felder übergeben, auch die Felder die not enabled und not visible definiert sind, mit Ausnahme der Felder, für die explizit InsertOnEmptyField=False in der Felddefinition gesetzt wurde.

In der Beispielanwendung „Digitaler Atlas“ wird der Parameter im Pflegeautomaten Frm\_PAM\_Mail benutzt.

### 6.4.2. Die Formate fRichMemo und fListView

Für Textfelder gibt es zusätzlich zum Format fMemo noch das Format fRichMemo. fRichMemo-Felder stellen im Pflegeautomaten zur Bearbeitung des Feldinhaltes einen kleinen Texteditor zur Verfügung. Der Feldinhalt wird dann in einem komprimierten rtf-Format in der Datenbank gespeichert. Datenbankseitig sollten ausreichend große Character-Felder (z.B. CLOB, LONGTEXT) angelegt werden. Binäre Feldtypen (BLOB) dürfen nicht verwendet werden. Soll das Einfügen von Bildern in den Text verhindert werden, so muss zusätzlich der Parameter AllowImageInsert auf False gesetzt werden.

Für den Fall, dass ganze Dateien in ein Feld importiert werden sollen, gibt es noch das Format fListView. Auch für dieses Format wird datenbankseitig ein Character-Feldtyp (z.B. CLOB, LONGTEXT) benötigt.

### 6.4.3. Parameter für den Email-Versand

Ist in unserem cySystem der cyMessenger aktiv, besteht die Möglichkeit, dass aus einem Pflegeautomaten heraus direkt eine Email versendet werden kann. Dazu müssen wir zunächst in der Sektion des Pflegeautomaten den Parameter MailSupport auf true setzen:

```
[cyPAM_Mail]
MailSupport=True
```

**[cyPAM\_MAIL.SQL]**

```
Select MAIL.MAIL_IDX,  
       MAIL.MAIL_ADDR,  
       MAIL.MAIL_CC,  
       MAIL.MAIL_BCC,  
       MAIL.MAIL_SUBJECT,  
       MAIL.MAIL_ATTACHMENT,  
       MAIL.MAIL_TEXT,  
       MAIL.MAIL_MASTER,  
       MAIL.MAIL_MASTERTIMESTAMP,  
       MAIL.MAIL_USER,  
       MAIL.MAIL_TIMESTAMP,  
       USR.USR_EMAIL  
from MAIL LEFT JOIN USR ON MAIL.MAIL_MASTER = USR.USR_LOGON  
where MAIL.MAIL_IDX = :MAIL_IDX and MAIL.MAIL_MASTER = :USER
```

Nach der Definition der Felder unseres Pflegeautomaten bestimmen wir anschließend in einer zusätzlichen Mailing-Sektion, mit welchen Werten die Variablen unsere Email gefüllt werden sollen. Dabei sind die Variablen Address, Subject und Text Mussfelder. Zusätzlich können die Variablen CC (Carbon Copy), BCC (Blind CC), Attach und MailFrom benutzt werden. Können wir alle Werte aus dem Selektionsdataset wie im obigen Beispiel verwenden, sieht die Mailing-Sektion wie folgt aus:

**[cyPAM\_MAIL.MAIL]**

```
Subject=:MAIL_SUBJECT  
Address=:MAIL_ADDR  
CC=:MAIL_CC  
BCC=:MAIL_BCC  
Text=:MAIL_TEXT  
Attach=:MAIL_ATTACHMENT  
MailFrom=:USR_EMAIL
```

Dabei dürfen die Adressfelder auch mehrere Adressen enthalten, die durch Semikolon getrennt sein müssen.

Neben der Zuweisung eines Parameters aus dem Selektionsdataset (<:name>) besteht bei allen Variablen auch die Möglichkeit einen festen Wert zuzuweisen. Darüber hinaus lassen sich Subject, Address, CC, BCC und Text über ein separates SQL-Statement mit Werten versorgen. Diese SQL-Statements werden in einer zusätzlichen Sektion hinterlegt. Zum Beispiel für die Variable Address:

**[cyPAM\_MAIL.MAIL.Address.SQL]**

```
Select email_address from ...
```

---

Dabei ist zu beachten, dass Angaben in der Mailing-Sektion Vorrang vor Angaben in einer SQL-Sektion haben.

#### **6.4.4. Modifyconflict-Prüfung einschränken**

Bevor im PAM geänderte Daten in die Datenbank zurückgeschrieben werden, wird standardmäßig überprüft, ob diese nicht in der Zwischenzeit von einem anderen Anwender verändert wurden. Ist das der Fall, so wird der Anwender auf diese Änderung hingewiesen und kann entscheiden, ob er die Daten trotzdem zurückschreiben oder die Änderung zurücknehmen will.

Enthält der zu ändernde Datensatz sehr große Felder (z.B. Felder vom Format fListView) kann diese Modifyconflict-Prüfung zu Performance-Problemen führen. Für diesen Fall gibt es den PAM-Parameter OriginFields=fast, der das Verfahren der Prüfung stark einschränkt. Zusätzlich sollte bei den Feldern mit den sehr großen Inhalten mit Hilfe des Parameters CheckConflict=False diese Prüfung generell ausgeschaltet werden.

#### **6.4.5. Lokale Dokumente direkt aus dem PAM öffnen**

Der PAM-Parameter ShellOpen=<SHELL> ermöglicht den Aufruf eines externen Programmes zur Anzeige einer lokalen Datei oder der Anzeige einer URL in einem Browser. Ist der Parameter definiert so erscheint in der Toolbox des PAM's ein Drucker Icon zum Öffnen des Dokumentes. Der zugewiesene Wert <SHELL> verweist auf eine gleichnamige Sektion in der Datei *\_PROGRAMS.cdd*.

In der Sektion wird über den OpenFile-Parameter definiert, was geöffnet werden soll. Es kann eine Datei fest zugeordnet (OpenFile=c:\temp\xyz.doc) oder über einen Parameter aus dem Selektionsdataset des PAM's, mit dem vollständigen Dateinamen, bereitgestellt werden (Openfile=:Filename). Ebenso können URL's eines Webserverns zugeordnet werden (OpenFile=http://www.dokument.server/xyz.html). Bei lokalen Dateien ist auch eine Prüfung auf das Vorhandensein (validate=true) der Dateien vor dem Aufruf des externen Programms möglich.

---

## 7. Zusätzliche Features

### 7.1. Clickvalue und Clickmap

Clickvalue und Clickmap dienen im Selektionsautomaten der Parameterübergabe an einen aufzurufenden Selektions- oder Pflegeautomaten. Bisher erfolgte die Parameterübergabe immer über Parameter der Form :<feldname>. Der aufgerufene Automat nutzt diesen Parameter mit dem aktuellen Wert aus der Selektionsdatenmenge seines Masters (Masterview) zur Einschränkung seiner zu selektierenden Datenmenge.

Nun gibt es aber auch Anwendungsbeispiele, in denen in der Selektionsdatenmenge Parameter vom gleichen Typ (z.B. Fremdschlüssel) mehrfach vorkommen (denormalisierte View).

So enthält zum Beispiel die Monatssicht eines Schichtplanes pro Tag eine eigene Spalte mit der jeweiligen Schicht des Mitarbeiters. Klickt der Anwender nun eine bestimmte Schicht an, um diese zu bearbeiten, bekommt der aufgerufene Pflegeautomat eine Selektionsdatenmenge mit bis zu 31 Schichtplan-Schlüsseln übergeben. Welchen dieser Schichtpläne soll er nun darstellen?

Genau für diesen Fall gibt es den Parameter :Clickvalue. Er enthält genau den angeklickten Wert. Wichtig ist, dass dem Aufruf dieses Pflegeautomaten in der Action-Sektion der View alle Felder, die einen Schichtplan-Schlüssel enthalten, zugeordnet werden.

Jetzt kann es aber auch noch vorkommen, dass der angeklickte Wert in unserem Schichtplan gar nicht den eindeutigen Schlüssel der Schicht sondern die Bezeichnung der Schicht enthält.

In diesem Fall benötigen wir noch zusätzlich eine Clickmap-Sektion, in der wir jedem anzuklickenden Feld (z. B. SCH\_01) den als Clickvalue zu übergebenden Schlüssel (z. B. SCH\_IDX\_01) zuordnen, zum Beispiel:

```
[View_Schichtplan.Clickmap]
```

```
SCH_01=SCH_IDX_01
```

```
...
```

```
SCH_31=SCH_IDX_31
```

```
[View_Schichtplan.Action]
```

```
OnDBLClick.SCH_01=FORM.Frm_PAM_Schicht
```

```
.....
```

```
OnDBLClick.SCH_31=FORM.Frm_PAM_Schicht
```

---

Im SQL-Statement des Pflegeautomaten können wir anschließend durch die Where-Bedingung SCH\_IDX = :Clickvalue auf die angeklickte Schicht zugreifen.

Für Felder, die nicht in der Clickmap zugeordnet werden, muss in ihrer Definitions-Sektion der Parameter Focusing auf False (default ist True) gesetzt werden.

## 7.2. Drag & Drop

Wie bereits erwähnt, lassen sich Selektionsautomaten mit Hilfe von Drag & Drop auch zur Datenpflege nutzen. Dies ist insbesondere dann nützlich, wenn größere Datenmengen zueinander in Beziehung gesetzt werden müssen, wie zum Beispiel bei der Zuordnung von Personen zu Gruppen. Mit Hilfe von Pflegeautomaten kann jeweils immer nur eine Zuordnung bearbeitet werden, was bei großen Mengen recht mühsam werden kann. Im Gegensatz dazu zieht man einfach die selektierten Daten eines Selektionsautomaten in einen anderen und über die hinterlegten Definitionen werden die gewünschten Zuordnungen angelegt beziehungsweise aufgehoben. Selbstverständlich sind alle Datenbankoperationen möglich, vorausgesetzt alle benötigten Attribute können mit Werten des „gedragten“, bzw. des „gedropten“ Datensatzes versorgt werden.

Welche Einstellungen für Drag & Drop erforderlich sind, werden wir uns am Beispiel der Zuordnung von Städten zu Ländern anschauen. Dazu greifen wir auf unser Beispiel aus Kapitel 5.3 zurück. In diesem Beispiel sind bereits alle Selektionsautomaten für eine Städte-Länder-Zuordnung per Drag & Drop vorhanden. Zur besseren Übersicht ändern wir zunächst die Anordnung der Selektionsautomaten wie in der Abbildung (Abb. 28) zu sehen und erweitern den Selektionsautomaten mit den zugeordneten Städten um die beiden InfoFields Ländercode und Land:

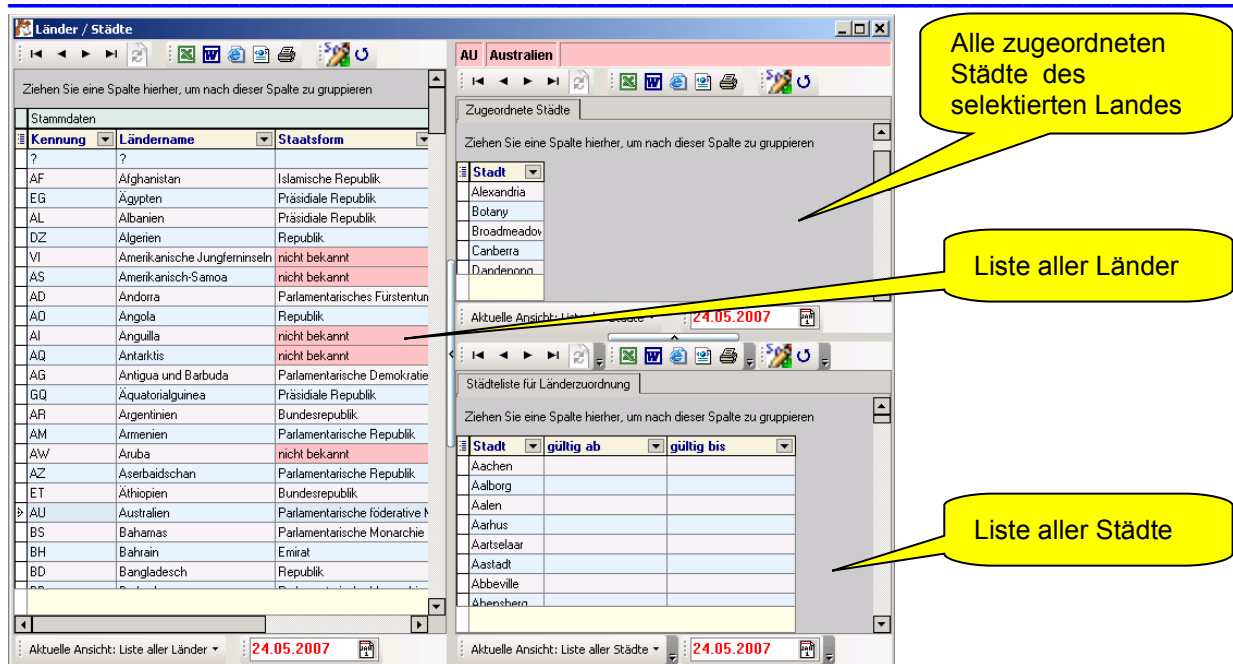


Abb. 28

Zur Aktivierung von Drag & Drop fügen wir in der Datei *Frm\_SAM\_CON\_Countries\_Cities* des ROLES-Verzeichnisse folgende Zeilen hinzu:

```

;Städte zuordnen
[View_Countries.DropAllowedList]
FRM_SAM_CON_countries_cities.VIEW_Cities_all=Cities_uCIT_TO_COU
[View_Countries.Cities_uCIT_TO_COU]
Provider=DemoDB
ApplyTable=CCM
IndexName=CCM_IDX
;cyRowId=
;ScriptProject=
;ExecBefore=
Operation=Insert
ModifierDate=CCM_MASTERTIMESTAMP
ModifierUser=CCM_MASTER
;WhereCondition=CCM_IDX=:CCM_IDX
AbortOnError=False
[View_Countries.Cities_uCIT_TO_COU.Fields]
COU_IDX=:DROP_COU_IDX
CIT_IDX=:DRAG_CIT_IDX

[View_Cities.DropAllowedList]
FRM_SAM_CON_countries_cities.VIEW_Cities_all=uCIT_to_aCIT

```

---

```

[View_Cities.uCIT_TO_aCIT]
Provider=DemoDB
ApplyTable=CCM
IndexName=CCM_IDX
Operation=Insert
ModifierDate=CCM_MASTERTIMESTAMP
ModifierUser=CCM_MASTER
AbortOnError=False
[View_Cities.uCIT_TO_aCIT.Fields]
COU_IDX=:DROP_COU_IDX
CIT_IDX=:DRAG_CIT_IDX

;Zuordnung der Städte aufheben
[View_Cities_all.DropAllowedList]
FRM_SAM_CON_countries_cities.VIEW_Cities=aCIT_to_uCIT
[View_Cities_all.aCIT_to_uCIT]
Provider=DemoDB
ApplyTable=CCM
IndexName=CCM_IDX
Operation=Delete
WhereCondition=CCM_IDX=:CCM_IDX
AbortOnError=False
[View_Cities_all.aCIT_to_uCIT.Fields]
CCM_IDX=:Drag_CCM_IDX

```

In der DropAllowedList einer View definieren wir aus welcher „Drag-View“ Werte in diese „gedrop“ werden dürfen und wie die dazugehörige Updateregul heißt:

```
<formname>.<drag-view>=<Updateregul>
```

Die Updateregul wird in einer zusätzlichen Sektion definiert und enthält die für die Datenbankänderung erforderlichen Informationen. In unserem Beispiel wird ein Insert in die Tabelle CCM mit dem Index CCM\_IDX durchgeführt. Weitere Datenbankoperationen wären Update und Delete. In diesen Fällen wird auch der Parameter WhereCondition benötigt, um den zu ändernden Datensatz eindeutig zu selektieren. AbortOnError wird bei Massenänderungen benötigt, um gegebenenfalls die Verarbeitung abzubrechen. Existieren in der Tabelle Felder die Änderungszeit und Änderungsuser dokumentieren, müssen diese über die Parameter ModifierDate und ModifierUser benannt werden.

Damit der cyAgent nun das SQL-Statement für unsere Insert-Operation aufbauen kann, fehlen ihm noch die Wertzuweisungen der betroffenen Datenbankfelder. Dies geschieht mit Hilfe der folgenden Fields-Sektion. Während cyAgent den nächsten freien Schlüssel CCM\_IDX der Tabelle CCM selbst ermittelt, erhält der Fremdschlüssel der Tabelle COU den Wert des gleichnamigen Feldes des Datensatz auf den „gedrop“ wurde und der

Fremdschlüssel der Tabelle CIT den Wert des gleichnamigen Feldes des Datensatzes aus dem „gedragt“ wurde.

Folgende Wertzuweisungen sind in einer Fields-Sektion möglich:

<DB-Feldname>=:Drag\_<Feldname der „drag-view“>

<DB-Feldname>=:Drop\_<Feldname der „drop-view“>

<DB-Feldname>=:<Parameter>

<DB-Feldname>=:<Konstante>

<DB-Feldname>=:

Im letzten Fall wird das Datenbankfeld auf Null gesetzt.

Damit können wir nun mehrere Städte in einem Arbeitsschritt Ländern zuordnen, indem wir sie in der Liste aller Städte (View\_Cities\_ALL) markieren und anschließend auf ein Land der Länderliste oder auf die Städteliste des aktuell selektierten Landes (View\_Cities) ziehen. Zum Löschen dieser Zuordnungen gilt der umgekehrte Weg. Hier ziehen wir aus der Liste der zugeordneten Städte einfach in die Liste aller Städte.

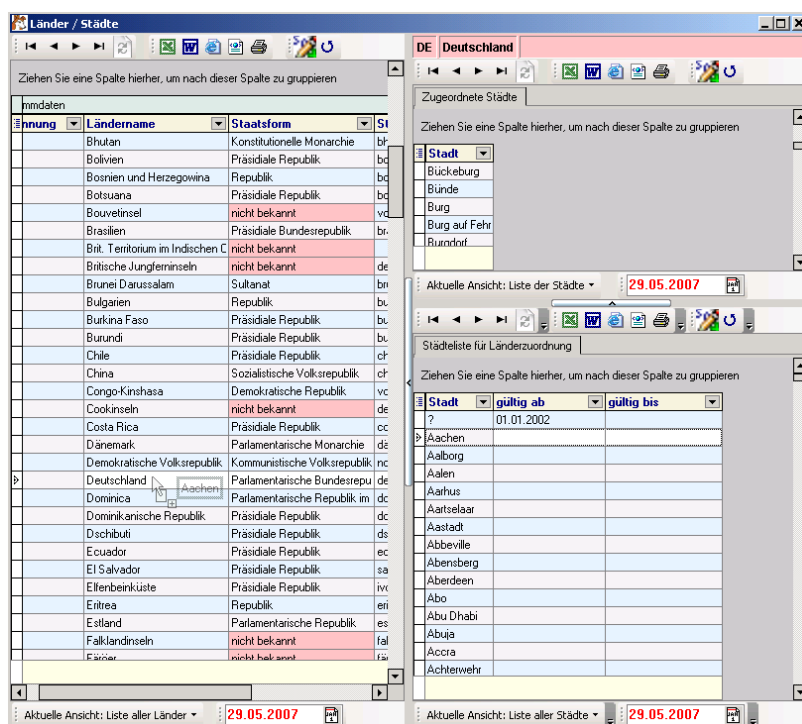


Abb. 29

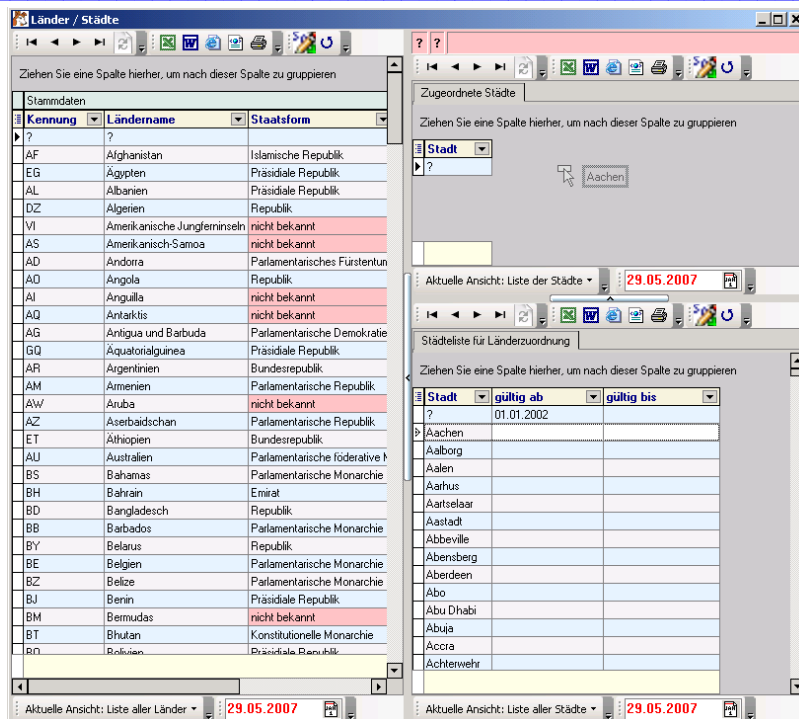


Abb. 30

Im obigen Beispiel haben wir immer ganze Zeile „gedragt“. Es gibt aber auch Anwendungsfälle (z.B. Schichtpläne), in denen nur bestimmte Zellen „gedragt“ werden sollen. Dazu müssen wir in den View-Sektionen der betroffenen Views den Parameter `AllowFieldSelect` aufnehmen. Über diesen Parameter wird definiert, welche Felder überhaupt „gedragt“ werden dürfen. Die Zuordnung kann durch Aufzählung der Felder oder generisch erfolgen.

```
AllowFieldSelect=<feldname1>|<feldname2>|...
AllowFieldSelect=*<teilstelfeldname>
```

Um in der Anwendung mehrere Zellen zu selektieren, wird beim Mausklick die Steuerungstaste gedrückt. Ist diese Zelle bereits selektiert, so wird die Selektion aufgehoben. Wird ohne Steuerung in eine nicht selektierte Zelle geklickt, werden alle Selektionen aufgehoben. Selektierte Zellen sind mit einem Rahmen umgeben. Die fokussierte Zelle ist zusätzlich mit einem Pfeil in der Zelle markiert.

Abschließend muss für das oben beschriebene Drag & Drop-Verfahren noch auf folgende Einschränkungen hingewiesen werden:

- 
- Es wird nur von Views der Selektionsautomaten unterstützt.
  - Die Datenbank-Provider müssen vom Typ ADO sein.
  - Drag & Drop aus oder in eine Detail View ist nicht möglich.

### 7.3. Parameter

In den oben beschriebenen Definitionsdateien haben wir das ein oder andere Mal Parameter eingesetzt. Der Vollständigkeit halber folgt nun eine Übersicht aller zur Verfügung stehender Parameter:

:APPNAME	Name der Anwendung
:USER	Name des angemeldeten Anwenders
:USERROLE	Name der Rolle des Anwenders
:SD	Selektionsdatum im DateFormat der View
:SDATETIME	aktuelles Datum plus Uhrzeit im Date/TimeFormat der View
:SDATE	aktuelles Datum im DateFormat der View
:STIME	aktuelle Uhrzeit im TimeFormat der View
:D	aktuelles Datum in aufbereiteter Form
:DATETIME	aktuelles Datum und Uhrzeit in aufbereiteter Form
:YEAR	das Jahr von :SD
:MONTH	der Monat von :SD
:DAY	der Tag von :SD
:<Feldname>	Inhalt des Feldes <Feldname> des selektierten Datensatzes der Masterview
:CLICKVALUE	Inhalt der angeklickten Zelle der Masterview
:TICKET	das aktuelle Ticket

---

## 7.4. Storefile

Der cyAgent unterstützt selbstverständlich auch die Personalisierung einer Anwendung. Dazu gehören alle vom Anwender vorgenommenen Einstellungen wie das Verschieben oder Vergrößern von Fenstern, die Zuordnung von Styles, alle möglichen Veränderungen an einem Selektions- oder Pflegeautomaten, die gewählte Sprache und so weiter. Beim Beenden der Anwendung schreibt der cyAgent alle individuellen Anpassungen in ein lokales Storefile.

Standardmäßig wird dieses Storefile unter dem Namen <applicationname>\_storefile.cdd in das Verzeichnis \Dokumente und Einstellungen\<user>\cyGui geschrieben. Über den Parameter StoreFileDir in der ini-Datei der jeweiligen Anwendung (siehe Kapitel 3.1.4) kann auch ein beliebiges Verzeichnis zur Ablage des Storefiles gewählt werden. Wichtig ist natürlich, dass der cyAgent den vollen Zugriff auf dieses Verzeichnis hat.

Ein Update der Anwendung, wie z.B. zusätzliche Felder in einem Selektionsautomaten, führt in der Regel dazu, dass ein bereits existierendes Storefile nicht mehr gültig ist. Sind die Namen der geänderten Selektions- oder Pflegeautomaten bekannt, reicht es die entsprechenden Sektionen in der Storefile zu löschen. Natürlich kann der Anwender auch das komplette Storefile löschen und nach dem Neustart der Anwendung wieder neu aufbauen.

Für den Fall, dass ein Storefile vom Anwender gelöscht wurde, kann der Entwickler dem Anwender ein initiales, von ihm bereits erstelltes Storefile zur Verfügung stellen. Dieses muß im cyAgent-Unterverzeichnis InitialStoreFiles abgelegt werden. Immer, wenn der cyAgent beim Starten im cyGui-Verzeichnis kein Storefile findet, prüft er, ob ein entsprechendes Storefile im InitialStoreFiles-Verzeichnis existiert.

---

## 8. Sprachdateien

Für jede in der *cyGuiApl.ini* definierten Sprache existieren zwei Sprachdateien mit sämtlichen sprachabhängigen Texten. Wechselt der Anwender die Sprache fordert der cyAgent diese beiden Sprachdateien vom cyProvider an und die Anwendung wird unter Verwendung der aktuellen Sprachdateien neu aufgebaut.

Wir unterscheiden globale, anwendungsübergreifende und die anwendungsabhängigen Sprachdefinitionen, die sich pro Sprache in einer separaten Sprachdatei befinden.

### 8.1. Anwendungsübergreifende Sprachdateien

Im Verzeichnis Sources\Globals\Language befinden sich die anwendungsübergreifenden Sprachdefinitionen, die pro Sprache in Dateien mit dem Namen *<countrycode>\_Globals.cdd* abgelegt sind. Neben System-/Fehlermeldungen, Beschriftungen der Buttons, Hinweistexten zu den Icons, bis zu den Beschriftungen des Stylesheet-Editors oder der Darstellung von Kalenderobjekten (Tage, Monate) sind alle globalen Sprachdefinitionen in diesen Dateien enthalten.

Im Folgenden werden die einzelnen Sektionen einer englischen Sprachdatei kurz vorgestellt:

#### **[COUNTRYINFO]**

```
CountryCode=en
LCID=2057
CountryNumber=1
CountryDescription=english
CountryFlag=9906
```

#### **[MESSAGES]**

Die Messages-Sektion enthält sämtliche Fehlermeldungen.

#### **[Express]**

In der Express-Sektion sind alle Texte der vom cyAgent benutzten Developer Express-Komponenten untergebracht. Da die Standard-Sprache dieser Komponenten Englisch ist, wird diese Sektion nur in den nicht-englischen Sprachdateien benötigt.

---

Die folgenden vier Sektionen werden für die Darstellung von Kalendern (siehe Kapitel 5.5) benötigt.

**[DaysOfWeekShort]**

1.Caption=Su.

...

7.Caption=Sa.

**[DaysOfWeekLong]**

1.Caption=Sunday

...

7.Caption=Saturday

**[MonthsShort]**

1.Caption=Jan

...

12.Caption=Dec

**[MonthsLong]**

1.Caption=January

...

12.Caption=December

**[BARMANAGER]**

In dieser Sektion befinden sich alle Hinweise und Texte der Schaltflächen sämtlicher Bars.

**[TCYFORM]**

Diese Sektion enthält zur Zeit nur die Texte des OK- und Abbrechen-Buttons eines Referenz-Selektionsautomaten.

**[TCYSAM]**

Die TCYSAM-Sektion enthält aktuell alle Texte des Stylesheet-Editors.

**[TCYPAM]**

In dieser Sektion sind die Texte für zusätzliche allgemeine Schaltflächen von Pflegeautomaten enthalten.

## 8.2. Anwendungsabhängige Sprachdateien

In Kapitel 5.1 haben wir für unser erstes Formular bereits einige Einträge in der deutschen Sprachdatei *de.cdd* vorgenommen, die sich im Language-Verzeichnis der Anwendung befindet. Anwendungsabhängige Sprachdateien bestehen aus den Sektionen AliasMap, einer Sektion für die Anwendung selbst (z.B. myApplication) sowie aus den Sektionen, die die Übersetzungen der Formulare und Views enthalten.

### [AliasMap]

```
cyPAM_Countries=View_Countries
cyPAM_Countries_PAMClone=View_Countries
cyPAM_Cities=View_Cities
Ref_SAM_Cities=View_Cities
cyPAM_Graphics=View_Graphics
```

Die AliasMap-Sektion bietet die Möglichkeit der Übernahme von Sprachdefinitionen aus einer anderen Sektion. Benötigt ein Pflegeautomat zum Beispiel viele Felder, deren Übersetzungen bereits unter einer View-Sektion definiert wurden, reicht es, die fehlenden Übersetzungen unter der View-Sektion aufzunehmen und dann eine Referenz unter der AliasMap-Sektion einzutragen.

### [myApplication]

```
Caption=myApplication
mnuFinish.Caption=Beenden&den
mnuFinish.Hint=Anwendung beenden / Formulare schließen
...
mnuModules.Caption=&Module
mnuModules.Hint=Module der Anwendung
mnuModules1.Caption=&Stammdaten
mnuModules1.Hint=Stammdaten
mnuModules11.Caption=&Länder / Städte
mnuModules11.Hint=Länder / Städte
mnuModules12.Caption=&Länder / Städte (Master-Detail)
mnuModules12.Hint=Länder / Städte
mnuModules13.Caption=&Länder / Städte mit Container
mnuModules13.Hint=Länder / Städte
...
```

Die myApplication-Sektion enthält neben dem Namen der Anwendung die Bezeichnungen für die einzelnen Menüpunkte.

Es folgen nun die Sektionen für alle benötigten Formulare und Views der Selektions-, Pflege- oder Referenz-Selektionsautomaten unter ihrem jeweiligen Namen. Die Sprachdefinitionen für die Felder werden unter dem Namen der jeweiligen View oder des

---

Pflegeautomaten vorgenommen, in der ebenfalls die Bezeichnungen der Tabulatoren, Bändern, Filter und ActionHints hinterlegt werden. Für die oben behandelten Beispiele sollten in der deutschen Sprachdatei *de.cdd* folgende Einträge vorhanden sein:

**[Frm\_SAM\_Countries\_Cities]**

Caption=Länder / Städte

**[Frm\_SAM\_CON\_Countries\_Cities]**

Caption=Länder / Städte

Tab1.Caption=Länder

Tab2.Caption=Städte zum Land

Tab3.Caption=alle Städte

**[Frm\_SAM\_MD\_Countries\_Cities]**

Caption=Länder / Städte (Master-Detail)

**[Frm\_PAM\_Countries]**

Caption=Länder - Pflegeautomat

Tab1.Caption=Stammdaten

Tab2.Caption=Städte

Tab3.Caption=Grafiken

Tab4.Caption=Bemerkungen

**[View\_Countries]**

Caption=Länder

Filter0=Liste aller Länder

Filter1=Liste aller zum Stichtag gültigen Länder

View\_Countries\_Band1.Caption=Stammdaten

View\_Countries\_Band2.Caption=Gültigkeitszeitraum

OnDBLClick.FOG\_FOG.ActionHint=Pflege der Staatsformen

OnDBLClick.ActionHint=Pflege der Länder

COU\_IDX.Caption=Länder-Id.

COU\_CODE.Caption=Kennung

COU\_COUNTRY.Caption=Ländernamen

COU\_COUNTRY\_0.Caption=Ländernamen (deutsch)

COU\_COUNTRY\_1.Caption=Ländernamen (englisch)

COU\_COUNTRY\_LOCAL.Caption=Ländernamen (national)

COU\_NATIONALITY.Caption=Staatsangehörigkeit

COU\_NATIONALITY\_0.Caption=Staatsangehörigkeit (deutsch)

COU\_NATIONALITY\_1.Caption=Staatsangehörigkeit (englisch)

COU\_VALID\_FROM.Caption=gültig ab

COU\_VALID\_UP\_TO.Caption=gültig bis

COU\_REMARK.Caption=Notiz

COU\_DIALCODE.Caption=Landesvorwahl

---

FOG\_FOG.Caption=Staatsform  
FOG\_IDX.Caption=Staatsform-Id.

**[View\_Cities]**

Caption=Städte  
Filter0=Liste aller Städte  
Filter1=Liste aller zum Stichtag gültigen Städte  
CIT\_IDX.Caption=Stadt-Id.  
CCM\_IDX.Caption=Zuordnungs-Id.  
CIT\_CITY.Caption=Stadt  
CIT\_CITY\_LOCAL.Caption=Stadt (national)  
CIT\_CITY\_INT.Caption=Stadt (international)  
CCM\_VALID\_FROM.Caption=gültig ab  
CCM\_VALID\_UP\_TO.Caption=gültig bis  
CCM\_REMARK.Caption=Bemerkungen

**[View\_Cities\_all]**

Caption=alle Städte  
Filter0=Liste aller Städte  
Filter1=Liste aller zum Stichtag gültigen Städte  
OnDBLClick.ActionHint=Pflege der Städte  
CIT\_IDX.Caption=Stadt-Id.  
CIT\_CITY.Caption=Stadt  
CIT\_CITY\_LOCAL.Caption=Stadt (national)  
CIT\_CITY\_INT.Caption=Stadt (international)  
CIT\_VALID\_FROM.Caption=gültig ab  
CIT\_VALID\_UP\_TO.Caption=gültig bis

**[Ref\_SAM\_Cities]**

Caption=Städte

**[View\_Graphics]**

Caption=Grafiken  
Filter0=Liste aller Grafiken  
Filter1=Liste aller zum Stichtag gültigen Grafiken  
OnDBLClick.ActionHint=Pflege der Grafiken  
GRA\_IDX.Caption=Grafik-Id.  
GRA\_GRAPHIC\_NO.Caption=Bildnr.  
GRA\_VALID\_FROM.Caption=gültig ab  
GRA\_VALID\_UP\_TO.Caption=gültig bis  
GRA\_GRAPHIC.Caption=Bild  
GRA\_REMARK.Caption=Notiz  
ICC\_IMAGECONTENTSCATEGORY.Caption=Bildkategorie  
ICC\_IMAGECONTENTSCATEGORY\_0.Caption=Bildkategorie (deutsch)

---

ICC\_IMAGECONTENTSCATEGORY\_1.Caption=Bildkategorie (englisch)  
GRF\_GRAPHICFORMAT.Caption=Grafikformat  
GRF\_GRAPHICFORMAT\_0.Caption=Grafikformat (deutsch)  
GRF\_GRAPHICFORMAT\_1.Caption=Grafikformat (englisch)  
GRF\_CODE.Caption=Grafikformat Kennung

## 9. Styles

Jeder (Referenz-) Selektionsautomat bietet dem Anwender die Möglichkeit, über das Styles-Icon den Stylesheet-Editor aufzurufen und diesem Selektionsautomaten ein anderes vordefiniertes Layout zuzuordnen bzw. eigene Designs zu entwerfen und diese zu verwenden. Die vordefinierten Styles sind in der Datei `_Styles.cdd` im Def-Files-Verzeichnis zentral für die Anwendung abgelegt. In den Definitionen der Selektionsautomaten haben wir bereits die Styles „Standard“ bzw. „Referenz“ (Parameter `ActiveSheetName`) als initiale Styles unseren Selektionsautomaten zugewiesen. Die vom Anwender definierten Styles werden in der Datei `myApplication_StyleFile.cdd` abgelegt, die sich im cyGui-Verzeichnis befindet, welches wiederum im Ordner „Dokumente und Einstellungen“ zu finden ist, wenn nicht über den `StorefileDir`-Parameter ein anderer Ablageort festgelegt wurde. Ordnet der Anwender einen neuen Style einem Selektionsautomaten zu, so wird dieser auch von den von diesem Automaten abhängigen Selektions- oder Pflegeautomaten übernommen.

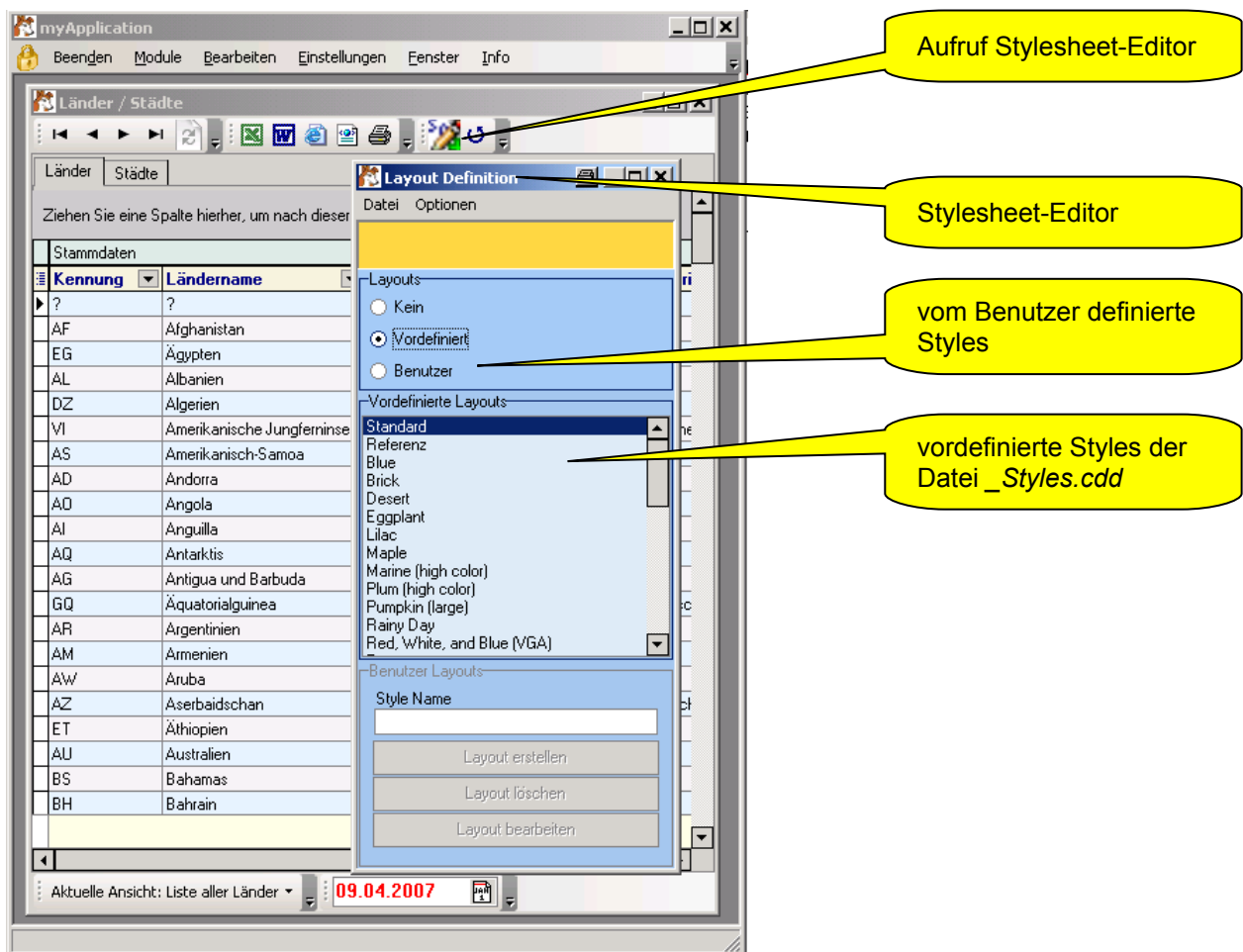


Abb. 31

Wie einzelne Style-Elemente von einem abhängigen Pflegeautomaten übernommen werden, wird in der *\_Design.cdd* (siehe nächstes Kapitel) festgelegt.

Das folgende Beispiel enthält die Definitionen des Styles „Standard“. Dem jeweiligen Style-Element werden die dezimalen RGB-Farben für den Hintergrund, die Schriftfarbe, die Schriftart, der Schriftgrad, der Schriftschnitt (Standard, Kursiv, Fett, Fett Kursiv) und die Darstellungsart (unterstrichen, durchgestrichen) zugeordnet.

#### [Standard]

Background=13158600,0,MS Sans Serif,8,[]	Grid Hintergrund
BandBackground=15451300,0,MS Sans Serif,8,[]	Band Hintergrund
BandHeader=14871257,0,MS Sans Serif,8,[]	Band Header
Content=14872561,0,MS Sans Serif,8,[]	Grid-Zeileneinhalt
ContentEven=16445670,0,MS Sans Serif,8,[]	Grid even
ContentOdd=15790320,0,MS Sans Serif,8,[]	Grid odd
FilterBox=14811135,0,MS Sans Serif,8,[]	Filter Box
Footer=14811135,8388608,MS Sans Serif,8,[B]	Grid Fußzeile
Group=15787730,0,MS Sans Serif,8,[]	Gruppierungsbereich innerhalb des Grids
GroupByBox=13158600,0,MS Sans Serif,8,[]	Gruppierungsbereich oberhalb Grid Header
Header=14479600,8388608,MS Sans Serif,8,[B]	Grid Header
Inactive=16247513,0,MS Sans Serif,8,[]	selektierte, aber inaktive Grid Zeile
IncSearch=12698111,0,MS Arial,9,[B]	Positionierung in einer Spalte
Indicator=16777215,0,MS Sans Serif,8,[]	Grid-Indikatorspalte
Preview=16777215,0,MS Sans Serif,8,[]	Vorschau im Stylesheet Editor
Selection=16777215,0,MS Sans Serif,8,[]	Selektierte Grid-Zeile

Mit dem Stylesheet-Editor definierte Styles können natürlich aus der lokalen *myApplication\_StyleFile.cdd* in die *\_Styles.cdd* übernommen und als zusätzliche vordefinierte Styles zur Verfügung gestellt werden.

## 10. Design

Während alle Style-Definitionen im vorherigen Kapitel im Zusammenhang mit dem in Selektionsautomaten integrierten Stylesheet-Editor standen, werden im Folgenden weitere anwendungsspezifische Designmöglichkeiten vorgestellt. In der Datei `_Design.cdd` im Def-Files-Verzeichnis stehen folgende Sektionen für weitere Definitionen zur Verfügung:

### [TcyForm]

(Referenzen auf Elemente des Stylesheet-Editors)

### [TcySAM]

InfoStyle=IncSearch

Zusätzliche Style-Elemente von Selektionsautomaten: Style der Infofelder

### [TcyPAM]

InfoStyle=IncSearch

Style-Elemente von Pflegeautomaten

NormalStyle=ContentEven

Style der Infofelder

FocusedStyle=GroupByBox

Style der Eingabefelder

LabelStyle=Header

Style des fokussierten Feldes

Style der Label

### [CriticalColors]

Farben für spezielle Felder von Pflegeautomaten:

ColorDisabled=\$E0E0E0

Hintergrundfarbe für Anzeigefelder

ColorNotNull=\$EBC4A4

Hintergrundfarbe für Mussfelder

ColorConflict=\$FF0000

Hintergrundfarbe bei unzulässigen Eingaben

### [Styles]

zusätzliche Styles für Style-

pinkblack

Conditions

### [pinkblack]

color=\$00C1C1FF

font.name=Arial

font.color=\$00000000

### [FOG\_UNKNOWN\_STYLE]

pinkblack=(FOG\_IDX = 0)

Style-Condition: Der Style pinkblack wird gesetzt, wenn das Feld FOG\_IDX den Wert 0 hat.

---

## 11. Zugriff auf SAP® Systeme

Der in den bisherigen Beispielen benutzte Provider DemoDB, über den der cyProvider der cySystem-Anwendung die gewünschten Daten bereitstellt, ist vom Typ ADO (Microsoft ActiveX Data Objects). Mit diesem Providertyp ist ein Zugriff auf alle ole- oder odbc-fähigen Datenbanken möglich. Natürlich könnte man diesen auch nutzen, um lesend auf die Datenbank eines SAP® Systems zu zugreifen. Bezüglich der SAP® Berechtigungen würde man aber die Kontrolle des SAP® Systems umgehen. Deshalb wurde für SAP® Systeme von cyFlex ein eigener SAP®-Provider entwickelt, der unter Berücksichtigung des SAP® Berechtigungssystems lesenden und schreibenden Zugriff ermöglicht. Im Folgenden werden Anforderungen und Arbeitsweise des SAP®-Provider beschrieben.

Der SAP®-Provider nutzt die SAP® eigene RFC-Schnittstelle (Remote Function Call), um mit Hilfe von speziellen Funktionsbausteinen die SAP®-Daten zu lesen bzw. zu ändern.

Dies bedeutet für unsere cySystem-Anwendungen, dass wir dem SAP®-Provider neben der gewünschten SAP® Tabelle (oder View) auch den Namen des zugehörigen Funktionsbausteins mitteilen müssen. Dazu erweitern wir die SQL-Befehle in unseren Definitionsdateien derart, dass wir vor die gewünschte Tabelle den Namen des Funktionsbausteins angeben, der die Tabelle liefern bzw. ändern soll. Um z.B. die SAP® Tabelle SCARR aus dem Mandanten 000 mit Hilfe des Funktionsbausteins Z\_FLIGHT\_MODEL\_DATA in einem Selektionsautomaten anzuzeigen, benötigen wir folgende SQL-Sektion:

```
[DemoView.Sql]
Select * from Z_FLIGHT_MODEL_DATA.SCARR where (MANDT='000')
```

Im SAP® System müssen Funktionsbausteine entwickelt werden, die den eigentlichen Datenzugriff durchführen. Voraussetzung für den fehlerfreien Aufruf dieser Funktionsbausteine durch den SAP®-Provider ist die Existenz der folgenden Schnittstellenparameter:

- Importparameter
  - QUERY\_TABLE (Name der Tabelle bzw. der View, Char 30)
  - ACTION (Datenbank-Aktion, Char 1, S = Select, U = Update, I = Insert, D = Delete, wird vom SAP®-Provider automatisch gesetzt)
  - LOGGING (optional, Char 1, falls der Funktionsbaustein ein Logging implementieren soll)
- Exportparameter

- 
- SYSTEM (Abap-Returncode, Integer)
  - Tabellen
    - FIELDS (Exporttabelle mit der Beschreibung der Tabellenfelder in der SAP® Struktur RFC\_DB\_FLD)
    - OPTIONS (Importtabelle mit der Where Clause der Select-Anweisung in der SAP® Struktur RFC\_DB\_OPT, bei größeren Where-Bedingungen muss die Länge der Struktur von 72 Stellen erweitert werden)
    - alle Exporttabellen, die über den Importparameter QUERY\_TABLE angefordert werden können (z.B. SCARR)
  - EXCEPTIONS

Optional definierbare Ausnahmen, die zum vorzeitigen Ende des Funktionsbausteins führen.

Der Quelltext eines Funktionsbausteins beinhaltet in Abhängigkeit von den möglichen Importparametern (QUERY\_TABLE, ACTION, OPTIONS) das Coding für die Datenbeschaffung bzw. Datenänderung inkl. der Berechtigungsprüfung. Der Funktionsbaustein Z\_FLIGHT\_MODEL\_DATA steht unter [www.cyflex.de](http://www.cyflex.de) als Beispiel zur Verfügung.

Der SAP®-Provider setzt den vom cyAgent empfangenen SQL-Befehl in einen Aufruf des jeweiligen Funktionsbausteins um. War der Aufruf erfolgreich, werden die Daten der Ergebnistabelle mit Hilfe der Informationen der ebenfalls exportierten FIELDS Tabelle konvertiert und dem cyAgent zur Verfügung gestellt.

Zur Zeit können immer nur alle Felder (Select \*) genau einer SAP®-Tabelle vom cyAgent angefordert werden. Werden die Daten mehrerer Tabellen benötigt, müssen im SAP® System entsprechende Views angelegt werden.

Die folgende Abbildung (Abb. 32) zeigt die SAP®-Tabelle SCARR aus dem obigen Beispiel in einem Selektionsautomaten:

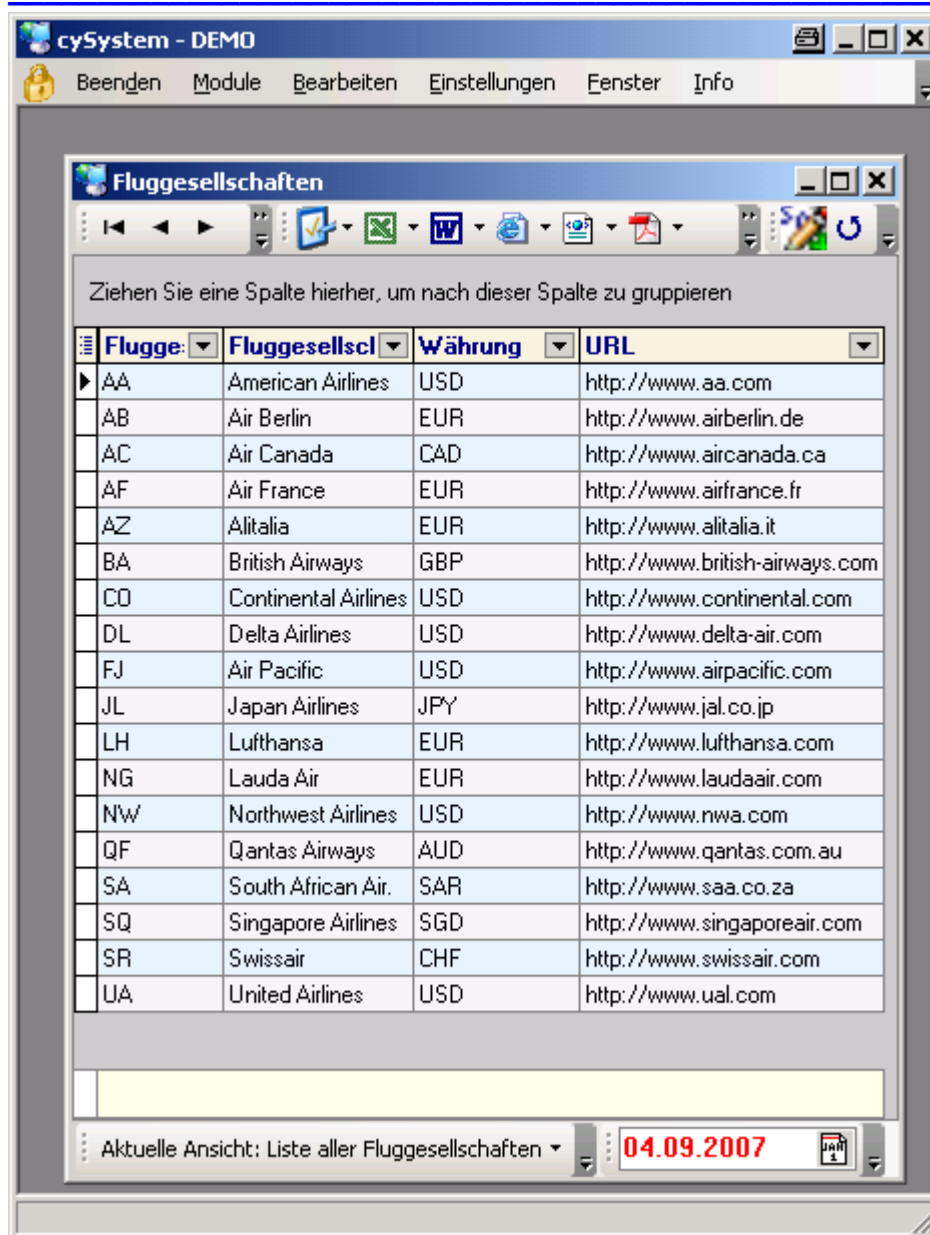


Abb. 32

Das abschließende Beispiel in Abb. 33 zeigt einen Selektionsautomaten mit Master-Detail-Beziehungen, in dem die Daten abwechselnd aus einem SAP® System (via SAP®-Provider) und einer Datenbank (via ADO-Provider) stammen.

SAP / ORACLE - Fluggesellschaften und Detailinformationen

Ziehen Sie eine Spalte hierher, um nach dieser Spalte zu gruppieren

Flugges.-Id.	Fluggesellschaft	Währung	URL
FJ	Air Pacific	USD	http://www.airpacific.com
JL	Japan Airlines	JPY	http://www.jal.co.jp
LH	Lufthansa	EUR	http://www.lufthansa.com

Ziehen Sie eine Spalte hierher, um nach dieser Spalte zu gruppieren

Global Key	Bezeichnung	Passagierzahl	Länge	Spannweite	Höhe	Reichweite	Hersteller	Fluggesellschaft
LH_737_300	Boeing 737-300	123	33,40	28,88	11,13	2.500	Boeing	Lufthansa
LH_737_500	Boeing 737-500	103	31,01	28,88	11,13	2.500	Boeing	Lufthansa
LH_747_400	Boeing 747-400	390	70,66	64,44	19,40	12.200	Boeing	Lufthansa
LH_A300_600	Airbus A 300-600	280	54,08	44,84	16,53	3.900	Airbus	Lufthansa
LH_A319	Airbus A 319	126	33,84	34,10	12,00	3.400	Airbus	Lufthansa
LH_A320	Airbus A 320	150	37,57	34,10	12,00	3.200	Airbus	Lufthansa
LH_A321	Airbus A 321	182	44,51	34,10	12,00	4.200	Airbus	Lufthansa

Ziehen Sie eine Spalte hierher, um nach dieser Spalte zu gruppieren

Flugges.-Id.	Verbindungs-Id.	Abflugdatum	Preis	Währung	Flugzeugtyp	Anzahl Plätze	Belegte Plätze	Gesamtsumme
LH	2402	18.09.2007	485,00	EUR	A321	220	218	99.458
LH	2402	27.11.2007	485,00	EUR	A321	220	92	41.472
LH	2402	22.01.2008	485,00	EUR	A321	220	84	38.339
LH	2402	04.03.2008	485,00	EUR	A321	220	1	412
LH	2402	01.04.2008	485,00	EUR	A321	220	0	0
LH	2402	15.04.2008	485,00	EUR	A321	220	8	3.787
LH	2407	16.09.2007	485,00	EUR	A321	220	210	94.599
LH	2407	25.11.2007	485,00	EUR	A321	220	56	26.980
LH	2407	20.01.2008	485,00	EUR	A321	220	0	0
LH	2407	02.03.2008	485,00	EUR	A321	220	0	0
LH	2407	30.03.2008	485,00	EUR	A321	220	0	0
LH	2407	13.04.2008	485,00	EUR	A321	220	13	5.868

Ziehen Sie eine Spalte hierher, um nach dieser Spalte zu gruppieren

Global Key	Bezeichnung	Passagierzahl	Länge	Spannweite	Höhe	Reichweite	Hersteller	Fluggesellschaft
LH_A330_300	Airbus A 330-300	221	63,66	60,30	16,83	10.000	Airbus	Lufthansa
LH_A340_300	Airbus A 340-300	247	63,66	60,30	16,91	11.500	Airbus	Lufthansa
LH_A340_600	Airbus A 340-600	345	75,30	63,45	17,30	14.260	Airbus	Lufthansa
LH_ATR42_500	Aerospatiale ATR 42-500	44	22,67	24,57	7,65	900	Aerospatiale	Lufthansa

Aktuelle Ansicht: Liste aller Fluggesellschaften | 04.09.2007

Airline-Stammdaten aus einem SAP-System

Detaildaten von Flugzeugen der selektierten Airline aus einer Datenbank (kein SAP)

Flugdaten zu dem jeweiligen Flugzeug aus einem SAP-System

Abb. 33

## Index:

ActionHints.....	26, 85	InsertFromMaster.....	62, 67
AllowFieldSelect.....	79	InsertOnEmpty.....	71
Anwendungsfenster.....	6	InsertOnEmptyField.....	71
Apply Table.....	54	Language.....	13, 28
Applydatenmenge.....	58, 67	Link.....	13, 34, 41
ApplyTarget.....	59	LogonProvider.....	16
AutoLogon.....	14	MailSupport.....	71
Band.....	26, 43	Master-Detail.....	29
Barglyph.....	13	MDI-Form.....	6
Checkbox.....	62	Menü.....	21
CheckConflict.....	73	ModifierType.....	63
Clickmap.....	74	Modifyconflict.....	73
Clickvalue.....	74	Mussfeld.....	52
Clone.....	69	myApplication.ini.....	17
Connect.....	13	Navigationsbar.....	52
Container.....	34, 63	Navigationsfunktion.....	52
cyGuiApl.ini.....	12, 14	OriginFields.....	73
cyMessenger.....	71	PanelAnchors.....	56
cyProvider.ini.....	16	Parameter.....	80
cyRowId.....	24	Performance.....	35
cyScripter.....	5	Pflegeautomat.....	8
Def-Files.....	12	Provider.....	14
DefaultLanguage.....	17	ReferenceSelection.....	59
Deny.....	54	ReferenceTarget.....	59
Design.....	13, 90	Referenz-Selektionsautomaten.....	9
DetailKeyField.....	32	Referenzdaten.....	8
Drag & Drop.....	7, 75	Referenzdatenmenge.....	59
DropAllowedList.....	77	Referenzfeld.....	49, 52, 58
EnableOnInsert.....	62, 68	ReferenzSelektionsautomaten.....	49
FastReport®.....	14, 42	Referenztabelle.....	49
Filter.....	26	RefForm.....	49
fMemo.....	61	RefreshMasterView.....	54
Focusing.....	75	Roles.....	13
Format.....	25, 56	Selektion-Box.....	59
fRichMemo.....	71	Selektionsautomat.....	7, 23
fSelect.....	59	Selektionsdatenmenge.....	54, 59, 67
Grid.....	7, 23	ShellOpen.....	73
InitialStoreFiles.....	81	ShowCDDInfo.....	15

---

SpecialBand.....	48	Stylesheet-Editor.....	88
Splitter.....	34	SyncSelDateChange.....	37
Sprachdatei.....	82	TabOrder.....	36
Sprachdefinition.....	82	Tabreiter.....	24, 35
Storefile.....	81	Tracelog.....	15
StoreFileDir.....	17, 81	Verzeichnisstruktur.....	10
Stylecondition.....	27	View.....	7, 24
Styles.....	13, 88		