Software:	**FRITZ!data**
Betriebssystem:	Windows 95/98/NT 4.0 (Workstation i386) und Windows 2000 (Professional)
Version:	ab 2.02
Revision:	ab 1.26

Dokumentation der COM-Schnittstelle von FRITZ!data

Diese Dokumentation ist eine Ergänzung des Handbuches von FRITZ!, sie gilt für das Modul FRITZ!data, das ab Version 2.02 Revision 1.26 das Component Object Model (COM) unterstützt. ***Bitte drucken Sie das Dokument auf Ihrem Drucker aus, und legen Sie es ihrem FRITZ!-Handbuch bei!***

INHALT:

---

1 Einleitung

FRITZ!data exportiert ein duales Interface zur Steuerung des Client-Modus. Mit Hilfe von 32-Bit-Windows-Anwendungen kann so die Datenübertragung automatisiert werden. Die Schnittstellen-Methoden können sowohl direkt (COM-Interface) als auch über Dispatch-Methoden (OLE-Automation-Interface) aufgerufen werden. Jeder Client, der diese Schnittstelle nutzt muss selbst ein OLE-Automation-Interface bereitstellen und bei FRITZ!data anmelden, um die Nachrichtenübermittlung an den Client zu ermöglichen.

FRITZ!data kann nicht gleichzeitig als COM-Server und interaktiv betrieben werden.

Folgende Datentransferfunktionen sind über diese Schnittstelle zu erreichen:

- Verbindungsaufbau und -abbau;
- Verzeichnisinhalt ermitteln und aktuelles Verzeichnis wechseln;
- Verzeichnisse erstellen;
- Dateien und Verzeichnisbäume kopieren;
- Dateien und Verzeichnisbäume bewegen;
- Dateien, Verzeichnisse und Verzeichnisbäume löschen.

Die einzelnen Aktionen werden durch den Aufruf von Schnittstellen-Methoden gestartet und deren Beendigung durch Nachrichten gemeldet. Sie können nur nacheinander ausgeführt werden, erfolgt der Aufruf einer entsprechenden Methode während einer laufenden Aktion, wird ein Fehler zurückgeliefert.

2 Registrierung des COM-Servers

Das FRITZ!-Installationsprogramm nimmt alle erforderlichen Registrierungen von FRITZ!data als COM-Server vor. Neben dieser Dokumentation ist die Typbibliothek FriDat32.tlb im Lieferumfang enthalten, sie ist im FRITZ!-Verzeichnis zu finden.

Dokumentname:	FritzData.Document
Schnittstelle:	IDualFritzdat (Version 1.0)
COM-Server:	FriDat32.exe

3 Schnittstellen-Methoden

Die folgende Beschreibung der Schnittstellen-Methoden richtet sich nach den syntaktischen Regeln der Microsoft® Interface Definition Language (MIDL). Für jede Methode sind zwei Deklarationen angegeben, die erste für die Verwendung des OLE-Automation-Interface und die zweite für das COM-Interface. Bei der Nutzung des COM-Interface wird ein Fehlercode zurückgeliefert, der sich auf den Aufruf der Methode über COM bezieht, nicht aber auf die Funktionalität, die mit dieser Methode implementiert ist.

3.1 SetClientId

```
[id(1)] void SetClientId(IDispatch* ClientId);
[id(1)] HRESULT SetClientId([in] IDispatch* ClientId);
```

ClientId	Zeiger auf die Rückrufschnittstelle. Diese muss die Methode `PutMessage` zur Verfügung stellen.

Mit dieser Methode wird der Zeiger auf ein Interface an FRITZ!data übergeben. Über die Methode `PutMessage` dieser Automation-Schnittstelle sendet FRITZ!data Nachrichten an den COM-Client. Näheres dazu im Abschnitt Nachrichtenaustausch.

3.2 Connect

```
[id(2)] void Connect(BSTR Address, BSTR User, BSTR Passwd, short Protocol,
long* pResult);
[id(2)] HRESULT Connect([in] BSTR Address, [in] BSTR User,
[in] BSTR Passwd, [in] short Protocol, [out] long* pResult);
```

Address	Rufnummer
User	Benutzername
Passwd	Passwort
Protocol	Flags (Bitfeld) [0]..[7]: reserviert, alle Bits zurücksetzten [8]: Protokoll IDtrans [9]: Protokoll Eurofile Transfer [10]: 2-Kanal-Transfer (nur IDtrans) [11]: Kompression (nur IDtrans) [12]: Kompression V.42bis nach CAPI 2.0 zulassen (wenn vom Installierten CAPI unterstützt, sonst ignoriert) [13]..[15]: reserviert, alle Bits zurücksetzten
pResult	Zeiger auf Puffer für einen Rückgabewert 0x10000: Verbindungsaufbau eingeleitet 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Verbindungsaufbau nicht möglich (z.B. bei bestehender Verbindung), siehe auch Kapitel **5** Fehlercodes

Die Methode leitet einen Verbindungsaufbau ein. Ist Result gleich 0x10000, wird der Abschluss mit einer Nachricht gemeldet.

3.3 Disconnect

```
[id(3)] void Disconnect(long* pResult);
[id(3)] HRESULT Disconnect([out] long* pResult);
```

pResult	Zeiger auf Puffer für einen Rückgabewert 0: Verbindungsabbau erfolgreich 0x10000: Verbindungsabbau eingeleitet 0x3nnnn, 0x4nnnn: Verbindungsabbau nicht möglich (z.B. bei fehlender Verbindung) , siehe auch Kapitel **5** Fehlercodes

Abbau einer bestehenden Verbindung. Ist Result gleich 0 oder 0x10000, wird der Abschluss mit einer Nachricht gemeldet. Diese Nachricht kann (bei 0) noch vor der Rückkehr aus dieser Methode empfangen worden sein.

3.4 ChangeDirectory

```
[id(4)] void ChangeDirectory(short Select, BSTR Directory, long* pResult);
[id(4)] HRESULT ChangeDirectory([in] short Select, [in] BSTR Directory,
[out] long* pResult);
```

Select	Verzeichnisauswahl (Bitfeld) [0]: lokales Verzeichnis wechseln [1]: entferntes Verzeichnis wechseln [2]..[15]: reserviert, alle Bits zurücksetzten
Directory	neues Verzeichnis als relativer oder absoluter Pfad [<Laufwerk>:][<Pfad>]<Name>
pResult	Zeiger auf Puffer für einen Rückgabewert 0: Verzeichniswechsel erfolgt 0x10000: Verzeichniswechsel eingeleitet 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Verzeichniswechsel nicht möglich, siehe auch Kapitel **5** Fehlercodes 0x50000: Server ist beschäftigt (letzte Aktion noch nicht abgeschlossen)

Die Methode wechselt das aktuelle Verzeichnis und/oder Laufwerk, lokal und bei aufgebauter Verbindung lokal oder entfernt. Das Format des Parameters Directory entspricht den Konventionen des Betriebssystems. Um festzustellen, ob der angerufene Server lange Dateinamen zulässt, ist die Methode `GetInfo` zu verwenden. Ist Result gleich 0 oder 0x10000, wird der Abschluss mit einer Nachricht gemeldet. Diese Nachricht kann (bei 0) noch vor der Rückkehr aus dieser Methode empfangen worden sein.

3.5 MakeDirectory

```
[id(5)] void MakeDirectory(short Select, BSTR Directory, long* pResult);
[id(5)] HRESULT MakeDirectory([in] short Select, [in] BSTR Directory,
[out] long* pResult);
```

Select	Verzeichnisauswahl (Bitfeld) [0]: lokales Verzeichnis [1]: entferntes Verzeichnis [2]..[15]: reserviert, alle Bits zurücksetzten
Directory	Name des neuen Unterverzeichnisses
pResult	Zeiger auf Puffer für einen Rückgabewert 0: Verzeichnis Erstellen erfolgreich 0x10000: Verzeichnis Erstellen eingeleitet 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Verzeichnis Erstellen nicht möglich, siehe auch Kapitel **5** Fehlercodes 0x50000: Server ist beschäftigt (letzte Aktion noch nicht abgeschlossen)

Die Methode erstellt ein neues Unterverzeichnis im aktuellen Verzeichnis, lokal und bei aufgebauter Verbindung lokal oder entfernt. Das Format des Parameters Directory entspricht den Konventionen des Betriebssystems. Um festzustellen, ob der angerufene Server lange Dateinamen zulässt, ist die Methode `GetInfo` zu verwenden. Ist Result gleich 0 oder 0x10000, wird der Abschluss mit einer Nachricht gemeldet. Diese Nachricht kann (bei 0) noch vor der Rückkehr aus dieser Methode empfangen worden sein.

3.6 Copy

```
[id(6)] void Copy(short Select, BSTR Name, long* pResult)
[id(6)] HRESULT Copy([in] short Select, [in] BSTR Name,
[out] long* pResult);
```

Select	Auswahl der Quelle (Bitfeld) [0]: Kopieren vom lokalen in das entfernte Verzeichnis [1]: Kopieren vom entfernten in das lokale Verzeichnis [2]: vorhandene Date(en) im Zielverzeichnis überschreiben (keine Rückfrage) [3]..[15]: reserviert, alle Bits zurücksetzten
Name	Name der Datei / des Verzeichnisses
PResult	Zeiger auf Puffer für einen Rückgabewert 0x10000: Kopieren eingeleitet 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Kopieren nicht möglich, siehe auch Kapitel **5** Fehlercodes 0x50000: Server ist beschäftigt (letzte Aktion noch nicht abgeschlossen)

`Copy` kopiert eine Datei oder ein Verzeichnis (mit Dateien und Unterverzeichnissen) zwischen den aktuellen Verzeichnissen, lokal und entfernt. Das Format des Parameters Name entspricht den Konventionen des Betriebssystems (ohne Pfad!). Es sind die Wildcards ”*” und ”?” zugelassen. Um festzustellen, ob der angerufene Server lange Dateinamen zulässt, ist die Methode `GetInfo` zu verwenden. Ist Result gleich 0x10000, wird der Abschluss mit einer Nachricht gemeldet.

3.7 Move

```
[id(7)] void Move(short Select, BSTR Name, long* pResult);
[id(7)] HRESULT Move([in] short Select, [in] BSTR Name,
[out] long* pResult);
```

Select	Auswahl der Quelle (Bitfeld) [0]: Verschieben vom lokalen in das entfernte Verzeichnis [1]: Verschieben vom entfernten in das lokale Verzeichnis [2]: vorhandene Date(en) im Zielverzeichnis überschreiben (keine Rückfrage) [3]..[15]: reserviert, alle Bits zurücksetzten
Name	Name der Datei / des Verzeichnisses
pResult	Zeiger auf Puffer für einen Rückgabewert 0x10000: Verschieben eingeleitet 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Verschieben nicht möglich, siehe auch Kapitel **5** Fehlercodes 0x50000: Server ist beschäftigt (letzte Aktion noch nicht abgeschlossen)

`Move` verschiebt eine Datei oder ein Verzeichnis (mit Dateien und Unterverzeichnissen) zwischen den aktuellen Verzeichnissen, lokal und entfernt. Das Format des Parameters Name entspricht den Konventionen des Betriebssystems (ohne Pfad!). Es sind die Wildcards ”*” und ”?” zugelassen. Um festzustellen, ob der angerufene Server lange Dateinamen zulässt, ist die Methode `GetInfo` zu verwenden. Ist Result gleich 0x10000, wird der Abschluss mit einer Nachricht gemeldet.

3.8 Delete

```
[id(8)] void Delete(short Select, BSTR Name, long* pResult);
[id(8)] HRESULT Delete([in] short Select, [in] BSTR Name,
[out] long* pResult);
```

Select	Verzeichnisauswahl (Bitfeld) [0]: Löschen aus lokalem Verzeichnis [1]: Löschen aus entferntem Verzeichnis [2]..[15]: reserviert, alle Bits zurücksetzten
Name	Name der Datei / des Verzeichnisses
PResult	Zeiger auf Puffer für einen Rückgabewert 0: löschen erfolgreich 0x10000: Löschen eingeleitet 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Löschen nicht möglich, siehe auch Kapitel **5** Fehlercodes 0x50000: Server ist beschäftigt (letzte Aktion noch nicht abgeschlossen)

`Delete` löscht eine Datei oder ein Verzeichnis (mit Dateien und Unterverzeichnissen) im aktuellen Verzeichnis, lokal oder entfernt. Das Format des Parameters Name entspricht den Konventionen des Betriebssystems (ohne Pfad!). Es sind die Wildcards ”*” und ”?” zugelassen. Um festzustellen, ob der angerufene Server lange Dateinamen zulässt, ist die Methode `GetInfo` zu verwenden. Ist Result gleich 0 oder 0x10000, wird der Abschluss mit einer Nachricht gemeldet. Diese Nachricht kann (bei 0) noch vor der Rückkehr aus dieser Methode empfangen worden sein.

3.9 Directory

```
[id(9)] void Directory(short Select, long* pResult);
[id(9)] HRESULT Directory([in] short Select, [out] long* pResult);
```

Select	Auswahl der Quelle (Bitfeld) [0]: lokales Verzeichnis [1]: entferntes Verzeichnis [2]: mit neuem Einlesen vom Laufwerk bzw. Server [3]..[15]: reserviert, alle Bits zurücksetzten
pResult	Zeiger auf Puffer für einen Rückgabewert 0: fehlerfrei (lokales Verzeichnis) 0x10000: Ausgabe eingeleitet (entferntes Verzeichnis) 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Ausgabe nicht möglich, siehe auch Kapitel **5** Fehlercodes 0x50000: Server ist beschäftigt (letzte Aktion noch nicht abgeschlossen)

`Directory` fordert die Einträge des aktuellen Verzeichnisses – lokal und entfernt – an. Die Verzeichniseinträge werden als einzelne Nachrichten geliefert. Diese können bereits vor Rückkehr aus dieser Methode empfangen worden sein (lokal und entfernt ohne Einlesen).

3.10 GetInfo

```
[id(10)] void GetInfo(short Flag, VARIANT* pInfo, long* pResult)
[id(10)] HRESULT GetInfo([in] short Flag, [out] VARIANT* pInfo,
[out] long* pResult);
```

Flag	Auswahl der Information (Bitfeld), siehe Funktionen weiter unten
pInfo	Zeiger auf Variable für Parameter und Rückgabewert
pResult	Zeiger auf Puffer für einen Rückgabewert 0: fehlerfrei, Rückgabewert gültig 0x20000: mind. ein Parameter fehlerhaft/ungültig 0x3nnnn, 0x4nnnn: Information nicht verfügbar, siehe auch Kapitel **5** Fehlercodes 0x50000: Server ist beschäftigt (letzte Aktion noch nicht abgeschlossen)

`GetInfo` liefert aktuelle Informationen. Für den Rückgabewert wird pInfo verwendet.

3.10.1 Verbindungsstatus

Flag	[8]: Abfrage des Verbindungsstatus
pInfo	Übergabe: leer Rückgabe: 2 Byte Integer 0: keine Verbindung [0]+[2]: Server beschäftigt [8]: Protokoll IDtrans [9]: Protokoll Eurofile Transfer [10]: 2-Kanal-Transfer (nur IDtrans) [11]: Kompression (nur IDtrans) [12]: Kompression V.42bis nach CAPI 2.0

Die Abfrage liefert den aktuellen Status der Verbindung, er ist während eines Verbindungsaufbaus nicht verfügbar. Sind die Bits für ”Server beschäftigt” gesetzt, werden gerade Daten zwischen den Gegenstellen ausgetauscht – der Aufruf einer anderen Schnittstellenmethode würde eine entsprechende Fehlermeldung liefern.

3.10.2 Servername

Flag	[9]: Abfrage des Namens des Servers zu dem Verbindung besteht
pInfo	Übergabe: leer Rückgabe: BSTR

Besteht eine Verbindung kann mit dieser Funktion der Name des angerufenen Servers ermittelt werden.

3.10.3 Verzeichnis

Flag	[0]: lokal [1]: entfernt [8]+[9]: aktuelles Verzeichnis
pInfo	Übergabe: leer Rückgabe: BSTR

Der zurückgelieferte Text ist der vollständige Pfad des aktuell eingestellten Verzeichnisses in der Form <Laufwerk:><absoluter Pfad>.

3.10.4 Lange Dateinamen

Flag	[10]: Unterstützung langer Dateinamen im entfernten Verzeichnis
pInfo	Übergabe: leer Rückgabe: bool false: keine langen Dateinamen true: lange Dateinamen

Die Unterstützung von langen Dateinamen im entfernten Verzeichnis ist abhängig von der IDtrans-Protokollversion (ab Version 6.0) die im angerufenen Server implementiert ist. Im Eurofile Transfer sind keine langen Dateinamen möglich. Die Verwendung von langen Dateinamen auf der lokalen Seite hängt von Protokollversion und der Einstellung unter Fritz/Einstellungen/FRITZ!data ab. Entweder finden hier immer lange Dateinamen Anwendung, oder nur dann, wenn auch der angerufene Server diese unterstützt.

3.10.5 Laufwerke

Flag	[0]: lokal [1]: entfernt [8]+[10]: verfügbare Laufwerke
pInfo	Übergabe: leer Rückgabe: 4 Byte Integer [0]..[25]: Laufwerke A:..Z:

Diese Funktion liefert Information darüber, welche Laufwerke verfügbar sind. Für jeden gültigen Laufwerksbuchstaben von A bis Z wird das korrespondierende Bit (0 bis 25) gesetzt.

3.10.6 ISDN- und CAPI-Fehler

Flag	[9]+[10]: ISDN- und CAPI-Fehler
pInfo	Übergabe: 2 Byte Integer; Fehlernummer Rückgabe: BSTR; Fehlermeldung

Zu jedem gültigen CAPI- oder ISDN-Fehlercode wird ein Text geliefert, der dem Benutzer als Fehlermeldung angezeigt werden kann. Die Nachrichten (siehe Abschnitte Nachrichtenaustausch und CAPI- oder ISDN -Fehler) liefern ebenfalls vollständige Fehlermeldungen.

3.11 Abort

```
[id(11)] void Abort(long* pResult);
[id(11)] HRESULT Abort([out] long* pResult);
```

pResult	Zeiger auf Puffer für einen Rückgabewert 0x10000: Abbruch eingeleitet 0x40000: keine abbrechbare Aktion

`Abort` bricht die laufende Aktivität ab. Ist Result gleich 0x10000, wird der Abschluss der laufenden Aktion mit einer Nachricht gemeldet.

4 Nachrichtenaustausch (Rückrufschnittstelle)

```
void PutMessage(short Message, long LParam, VARIANT VParam, long* pResult)
```

Diese Methode wird von FRITZ!data aufgerufen, um Nachrichten an den COM-Client zu senden. Sie darf keine langen Bearbeitungen enthalten oder Benutzereingaben abwarten, da sonst ein Timeout des COM-Aufrufmechanismus erreicht wird.

4.1 Verzeichniseintrag

Message	1: Eintrag ist Unterverzeichnis 2: Eintrag ist Datei 10: Ende (keine weiteren Verzeichniseinträge)
LParam	reserviert
VParam	BSTR; Eintrag (ungenutzt bei Message = 10)
pResult	reserviert

Die Nachrichten werden in Folge eines `Directory`-Aufrufes gesendet und liefern jeweils einen Verzeichniseintrag. Der String in VParam hat folgende, durch Tabulatorzeichen (0x09) getrennte Felder:

Name	Dateiname/Unterverzeichnisname
Größe	bei Unterverzeichnissen ”<DIR>”; bei Dateien die Größe in Byte, mit Dezimalpunkten zum Absetzen der Tausender (z.B. ”1.234.567”)
Datum	Datum der letzten Änderung in sprachspezifischer Formatierung entsprechend der Systemeinstellung
Zeit	Uhrzeit der letzten Änderung (ohne Sekunden) in sprachspezifischer Formatierung entsprechend der Systemeinstellung

Nach dem letzten Verzeichniseintrag wird die Nachricht ”Ende” gesendet. Ist das Verzeichnis leer, entfallen die Nachrichten für Dateien und Unterverzeichnisse.

4.2 Verbindungsaufbau

Message	3
LParam	Fehlercode
VParam	BSTR; Fehlermeldung
pResult	reserviert

Diese Nachricht signalisiert den Abschluss des Verbindungsaufbaus nach dem Aufruf von `Connect`. Ob dieser erfolgreich war, ist dem Fehlercode (siehe dazu Abschnitt Fehlercodes) zu entnehmen. Ist der Verbindungsaufbau fehlgeschlagen wird im Parameter VParam eine Fehlermeldung geliefert, die vom Client angezeigt werden kann.

4.3 Verbindungsabbau

Message	4
LParam	Fehlercode
VParam	BSTR; Fehlermeldung
pResult	reserviert

Mit dieser Nachricht wird ein Verbindungsabbau signalisiert, sie beantwortet einen Aufruf von `Disconnect`. Auch bei einem Abbau durch die Gegenstelle wird sie gesendet. LParam und VParam enthalten Fehlercode (siehe dazu Abschnitt Fehlercodes) und Fehlermeldung.

4.4 Verzeichnis wechseln

Message	5
LParam	Fehlercode
VParam	BSTR; Fehlermeldung
pResult	reserviert

In Folge eines Aufrufes von `ChangeDirectory` wird diese Nachricht gesendet. LParam und VParam enthalten Fehlercode (siehe dazu Abschnitt Fehlercodes) und Fehlermeldung.

4.5 Verzeichnis erstellen

Message	6
LParam	Fehlercode
VParam	BSTR; Fehlermeldung
pResult	reserviert

Diese Nachricht beantwortet `MakeDirectory`. LParam und VParam enthalten Fehlercode (siehe dazu Abschnitt Fehlercodes) und Fehlermeldung.

4.6 Kopieren

Message	7
LParam	Fehlercode
VParam	BSTR; Fehlermeldung
pResult	reserviert

Diese Nachricht beantwortet `Copy`. LParam und VParam enthalten Fehlercode (siehe dazu Abschnitt Fehlercodes) und Fehlermeldung.

4.7 Bewegen

Message	8
LParam	Fehlercode
VParam	BSTR; Fehlermeldung
pResult	reserviert

Diese Nachricht beantwortet `Move`. LParam und VParam enthalten Fehlercode (siehe dazu Abschnitt Fehlercodes) und Fehlermeldung.

4.8 Löschen

Message	9
LParam	Fehlercode
VParam	BSTR; Fehlermeldung
pResult	reserviert

Diese Nachricht beantwortet `Delete`. LParam und VParam enthalten Fehlercode (siehe dazu Abschnitt Fehlercodes) und Fehlermeldung.

4.9 Sicherheitsabfrage

Message	0x107: Sicherheitsabfrage Kopieren 0x108: Sicherheitsabfrage Bewegen
LParam	reserviert
VParam	BSTR; Frage
pResult	1: Ja (diese Datei überschreiben) 2: Nein (diese Datei nicht überschreiben) 3: Abbruch 4: Alle (alle Dateien überschreiben, keine weiteren Rückfragen)

Die Sicherheitsabfrage erfolgt, wenn zum Kopieren oder Bewegen Dateien überschrieben werden sollen und beim Aufruf der Methoden `Copy` und `Move` im Parameter Select Bit 2 nicht gesetzt ist. Der Name der betroffenen Datei ist im Text von VParam enthalten. Die Anforderung einer Benutzereingabe ist hier aber nicht zulässig, da bei Erreichen eines Timeouts im COM-Aufrufmechanismus die Funktion von FRITZ!data abgebrochen wird.

5 Fehlercodes

Die Fehlercodes, die in Nachrichten geliefert werden unterteilen sich entsprechend nachstehender Tabelle.

Fehler	oberes Wort	unteres Wort
kein Fehler aufgetreten	0	0
CAPI- oder ISDN-Fehler	3	CAPI- oder ISDN-Fehlercode
FRITZ!data-Fehler	4	0: keine nähere Information verfügbar <>0: Fehlercode siehe Abschnitt FRITZ!data-Fehler

5.1 CAPI- oder ISDN-Fehler

Die CAPI- und ISDN-Fehlermeldungen sind in der Datei I2ERR.HLP zu finden, sie wird vom FRITZ!-Installationsprogramm in das Windows-Verzeichnis kopiert. Die Fehlercodes (unteres Wort) entsprechen den Festlegungen der COMMON-ISDN-API Version 2.0, zweite Ausgabe vom August 1997.

Mit einem CAPI- oder ISDN-Fehlercode kann über die Methode `GetInfo` ein vollständiger Meldungstext ermittelt werden.

5.2 FRITZ!data-Fehler

Code	Fehler	Bemerkungen
0x0001	Fehler beim Dateitransfer	
0x001E	DOS-Zugriff verweigert	Betriebssystem verweigert Zugriff
0x001F	Datei/Ordner nicht gefunden	
0x0020	Zugriff auf Datei/Ordner verweigert	keine Zugriffsrechte
0x0021	Fehler beim Öffnen der Datei	
0x0022	Abbruch wegen Zeitüberschreitung	Protokollfehler (Timeout)
0x0023	Fehlerhafte Anmeldung beim Server	Benutzer und/oder Passwort ungültig
0x0026	Datei/Ordner existiert bereits	
0x0027	Abbruch wegen Benutzer-Zeitüberschreitung	automatischer Abbau einer ungenutzten Verbindung
0x0028	DOS-Fehler	
0x0064	Gegenstelle antwortet nicht	Protokollfehler
0x0065	Nicht genügend Diskettenplatz	
0x0066	Diskettenfehler	
0x0067	Gegenstelle erwartet Daten	Protokollfehler
0x0068	Gegenstelle bestätigt nicht	Protokollfehler
0x0069	Gegenstelle ist ungültig	Protokollfehler

0x006A	Nicht genügend Speicher	
0x006C	Pfad nicht zugänglich	
0x006D	Datei nicht gefunden	
0x006E	Verbindungskollision	Protokollfehler
0x006F	Funktion abgebrochen	Abbruch mit `Abort`
0x0070	Gegenstelle ist nicht kompatibel	Protokollfehler