



obj -> FsiSeek([int1]) : int

Dateizeigerposition ermitteln/setzen

obj      Datei-Deskriptor

int1      Neue Dateizeiger-Position  
(optional)

Resultat int Aktuelle Dateizeigerposition

Verwandte Befehle,

Siehe FsiSeek64(), FsiRead(),  
FsiWrite()

Diese Funktion dient zur Abfrage (ein Argument) oder zum Setzen (zwei Argumente) des Dateizeigers in der externen Datei (obj). Ab dieser Position wird anschließend gelesen (siehe FsiRead()) oder geschrieben (siehe FsiWrite()). Die Positionsangabe erfolgt in Bytes - das erste Byte einer Datei steht an Position 0. FsiSeek() kann nicht bei Dateien benutzt werden, die mit der Option \_FsiBuffer bei FsiOpen() geöffnet wurden.

Im Resultat wird die angegebene Position zurückgegeben.



Ab einer Dateigröße von 2 GB muss der Befehl FsiSeek64() verwendet werden.

Mit dem Befehl FsiSeek() kann der Dateizeiger auch hinter das Ende einer Datei positioniert werden. Bei einem anschließenden FsiWrite() wird die Datei auf die entsprechende Größe vergrößert. Dies kann ebenfalls mit dem Befehl FsiSize() erfolgen.

### Beispiel:

```
// Dateizeiger an Dateianfang setzentHandle->FsiSeek(0);
```

Mögliche Laufzeitfehler:

\_ErrHdlInvalid Der Datei-Deskriptor (obj) ungültig



obj -> FsiSeek64([bigint1]) : bigint  
Dateizeigerposition ermitteln/setzen (64 Bit)

obj Datei-Deskriptor

bigint1 Neue Dateizeiger-Position  
(optional)

Resultat bigint Aktuelle Dateizeigerposition

Siehe Verwandte Befehle, FsiSeek(),  
FsiRead(), FsiWrite()

Diese Funktion dient zur Abfrage (ein Argument) oder zum Setzen (zwei Argumente) des Dateizeigers in der externen Datei (obj). Ab dieser Position wird anschließend gelesen (siehe FsiRead()) oder geschrieben (siehe FsiWrite()). Die Positionsangabe erfolgt in Bytes - das erste Byte einer Datei steht an Position 0. FsiSeek64() kann nicht bei Dateien benutzt werden, die mit der Option FsiBuffer bei FsiOpen() geöffnet wurden.

Im Resultat wird die angegebene Position zurückgegeben.



Im Gegensatz zu dem Befehl FsiSeek() kann dieser Befehl auch bei Dateien mit mehr als 2 GB Größe verwendet werden.


Mit dem Befehl FsiSeek64() kann der Dateizeiger auch hinter das Ende einer Datei positioniert werden. Bei einem anschließenden FsiWrite() wird die Datei auf die entsprechende Größe vergrößert. Dies kann ebenfalls mit dem Befehl FsiSize64() erfolgen.

### Beispiel:

```
// Dateizeiger an Dateianfang setzentHandle->FsiSeek64(0);
```

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Datei-Deskriptor (obj) ist ungültig.

obj -> FsiSize([int1]) : int   
 Dateigröße ermitteln/setzen  
 obj Datei-Deskriptor  
 int1 Neue Dateigröße  
 (optional)

Resultat int Aktuelle Dateigröße

Siehe Verwandte Befehle,  
FsiSize64(), FsiOpen()

Mit dieser Funktion wird die Größe der externen Datei (obj) abgefragt (ein Argument) oder geändert (zwei Argumente). Sofern die Datei vergrößert wird, ist der Inhalt zwischen dem alten und dem neuen Dateiende undefiniert. Beim Ändern der Dateigröße muss die Datei mit Schreibberechtigung geöffnet sein. Das Ändern der Größe ist nicht bei Dateien möglich, die mit der Option FsiBuffer bei FsiOpen() geöffnet wurden. In diesem Fall liefert die Abfrage die Dateigröße zum Zeitpunkt von FsiOpen() zurück.

Das Resultat gibt die aktuelle Dateigröße in Byte zurück. Ist das Resultat negativ, ist ein Fehler aufgetreten und das Resultat enthält den Fehlerwert.



Ab einer Dateigröße von 2 GB muss der Befehl FsiSize64() verwendet werden.

FsiSize() kann auch zur Abfrage der Dateigröße eines Verzeichniseintrags benutzt werden (siehe FsiDirRead()). Dazu wird in (obj) der Deskriptor des Verzeichnisses übergeben.

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Datei-Deskriptor (obj) ist ungültig.



obj -> FsiSize64([bigint1]) : bigint  
Dateigröße ermitteln/setzen (64 Bit)

obj Datei-Deskriptor

bigint1 Neue Dateigröße  
(optional)

Resultat bigint Aktuelle Dateigröße

Siehe Verwandte Befehle,  
FsiSize(), FsiOpen()

Mit dieser Funktion wird die Größe der externen Datei (obj) abgefragt (ein Argument) oder geändert (zwei Argumente). Sofern die Datei vergrößert wird, ist der Inhalt zwischen dem alten und dem neuen Dateiende undefiniert. Beim Ändern der Dateigröße muss die Datei mit Schreibberechtigung geöffnet sein. Das Ändern der Größe ist nicht bei Dateien möglich, die mit der Option FsiBuffer bei FsiOpen() geöffnet wurden. In diesem Fall liefert die Abfrage die Dateigröße zum Zeitpunkt von FsiOpen() zurück.

Das Resultat gibt die aktuelle Dateigröße in Byte zurück. Ist das Resultat negativ, ist ein Fehler aufgetreten und das Resultat enthält den Fehlerwert.




Im Gegensatz zu dem Befehl FsiSize() kann dieser Befehl auch bei Dateien mit mehr als 2 GB Größe verwendet werden.

FsiSize64() kann auch zur Abfrage der Dateigröße eines Verzeichniseintrags benutzt werden (siehe FsiDirRead()). Dazu wird in (obj) der Deskriptor des Verzeichnisses übergeben.

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Datei-Deskriptor (obj) ist ungültig.

FsiSplitName(alpha1, ) : alpha

Dateinamen zerlegen

alpha1 Dateiname/-pfad

Optionen

<u>_FsiNamePNE</u>	Laufwerk, Pfad, Datei und Erweiterung ermitteln
<u>_FsiNamePN</u>	Laufwerk, Pfad und Datei ermitteln
<u>_FsiNameP</u>	Laufwerk und Pfad mit Separator ermitteln
<u>_FsiNameNE</u>	Datei und Erweiterung ermitteln
<u>_FsiNameN</u>	Datei ermitteln
<u>_FsiNameE</u>	Erweiterung ermitteln
<u>_FsiNamePP</u>	Laufwerk und Pfad ohne Separator ermitteln

Resultat alpha Teil des  
Dateinamens/-pfades

Siehe Verwandte Befehle,  
FsiDirRead(), FsiPath()

Mit dieser Funktion kann ein qualifizierter Dateiname (alpha1) in seine Bestandteile zerlegt werden. Je nach Option in (int2) wird der entsprechende Namensteil extrahiert und als Resultat zurückgeliefert.

## Beispiele

```
FsiSplitName('C:\DIR\FILE.EXT', _FsiNamePNE);    // 'C:\DIR\FILE.EXT'FsiSplitName('C:\DIR\FILE.EXT', _FsiNamePNE);
```



Es können bis zu 4096 Zeichen verarbeitet werden. Längere Angaben werden bei 4096 Zeichen abgeschnitten.



obj -> FsiStamp(int1[, bigint2]) : bigint

Dateiinformationen ermitteln/setzen (Zeitstempel)

obj Datei-Deskriptor

Informationstyp

FsiDtModified Letzten

Änderungszeitstempel  
ermitteln/setzen

int1 FsiDtAccessed Letzten

Zugriffszeitstempel  
ermitteln/setzen

FsiDtCreated Erstellungszeitstempel  
ermitteln/setzen

bigint2 Neuer Zeitstempel (optional)

Resultat bigint Aktueller Zeitstempel

Siehe Verwandte Befehle, FsiOpen(),  
FsiDate(), FsiTime()

Mit dieser Funktion können Zeitstempel in UTC-Zeit der externen Datei (obj) abgefragt (zwei Argumente) oder geändert werden (drei Argumente).

Bei FAT-Dateisystemen ist nur der Zeitstempel der letzten Änderung verfügbar. Bei Linux-Dateisystemen ist der Zeitstempel des letzten Zugriffs nicht verfügbar. Bei NTFS-Dateisystemen sind alle Zeitstempel verfügbar. Sofern ein Zeitstempel nicht verfügbar ist, wird 0\b zurückgeliefert.

FsiStamp() kann auch zur Abfrage der Zeitstempel eines Verzeichniseintrags benutzt werden (siehe FsiDirRead()). Dazu wird in (obj) der Deskriptor des Verzeichnisses übergeben.

Der Zeitstempel kann mit CnvCB() in einen caltime-Wert konvertiert werden.

Mögliche Laufzeitfehler:

ErrHdlInvalid Datei-Deskriptor (obj) ungültig



obj -> FsiTime(int1[, time2]) : time

Dateiinformatoren ermitteln/setzen (Zeitwerte)

obj      Datei-Deskriptor

Informationstyp

FsiDtModified    Letzte  
                         Änderungsuhrzeit  
                         ermitteln/setzen

int1      FsiDtAccessed Letzte  
                         Zugriffsuhrzeit  
                         ermitteln/setzen

FsiDtCreated    Erstellungsuhrzeit  
                         ermitteln/setzen

time2    Neue Uhrzeit (optional)

Resultat time      Aktuelle Uhrzeit

Siehe    Verwandte Befehle, FsiOpen(),  
          FsiDate(), FsiStamp()


Mit dieser Funktion können Zeitwerte der externen Datei (obj) ermittelt (zwei Argumente) oder gesetzt werden (drei Argumente).

Bei FAT-Dateisystemen ist nur die Uhrzeit der letzten Änderung verfügbar. Bei Linux-Dateisystemen ist die Uhrzeit des letzten Zugriffs nicht verfügbar. Bei NTFS-Dateisystemen sind alle Zeitwerte verfügbar. Sofern ein Zeitwert nicht verfügbar ist, wird 0:0:0.0 zurückgeliefert.

FsiTime() kann auch zur Abfrage der Zeitwerte eines Verzeichniseintrags benutzt werden (siehe FsiDirRead()). Dazu wird in (obj) der Deskriptor des Verzeichnisses übergeben.

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Datei-Deskriptor (obj) ist ungültig.


obj -> FsiWrite(var1[,   
int2]) : int

Dateibereich schreiben

obj      Datei-Deskriptor

var1      Speicherquelle

int2      Bereichsgröße (optional)  
            Anzahl geschriebener

Resultat int Zeichen oder   
            Fehlerwert

Verwandte Befehle,

Siehe FsiOpen(), FsiRead(),  
FsiSeek(), Fehlerwerte,  
Beispiel

Mit dieser Funktion werden Daten in die externe Datei (obj) ab der aktuellen Position geschrieben (siehe FsiSeek()). In (var1) kann ein Datenbankfeld, eine Variable, ein Array oder eine Zeichenketten-Konstante angegeben werden. Arrays aus Alphafeldern sind dabei nicht zulässig. Sofern (int2) nicht angegeben ist, wird die der Größe der Variablen (var1) entsprechende Anzahl von Bytes geschrieben. Der Wert in (int2) kann daher auch nicht größer als die Variable selbst sein.

Leerzeichen am Ende eines Alphawerts bleiben erhalten. Für die Erstellung von ASCII-Dateien muss ausschließlich mit alphanumerischen Werten gearbeitet werden. Andere Feldtypen müssen in der Prozedur entsprechend umgewandelt werden.

Beim Schreiben werden die Zeichenketten standardmäßig von der internen Zeichencodierung in die OEM-Zeichencodierung gewandelt. Durch entsprechende Angaben bei der Anweisung FsiOpen() kann eine andere Wandlung vorgenommen oder die Wandlung verhindert werden.

Sollen in eine externe Datei Nullen geschrieben werden, kann dafür keine Zeichenkette verwendet werden. Statt dessen kann zum Beispiel ein Array vom Typ byte verwendet werden. Da immer das gesamte Array geschrieben wird, werden auch die auf Null gesetzten Zellen geschrieben.

Das Resultat gibt die Anzahl der geschriebenen Bytes zurück. Ist das Resultat negativ ist ein Fehler aufgetreten und das Resultat enthält den Fehlerwert (\_ErrFsi...). Der Fehlerwert des Betriebssystems kann über die Eigenschaft FsiError abgefragt werden.


### Beispiel:

```
// Kopieren einer Dateidefine{ sMem                      : _Mem8K                      // 8 Kilobytes}sub FsiCopy( aSrc
// Datum & Uhrzeit wiederherstellen                      tDstHdl->FsiDate(_FsiDtModified, FsiDate(tSrcHdl, _Fsi
```

Mögliche Laufzeitfehler:

\_ErrHdlInvalid Datei-Deskriptor (obj) ungültig



obj -> FsiWriteMem(handle1, int2, )  
int3) : int


Memory-Objekt in Datei schreiben

obj Datei-Deskriptor

handle1 Deskriptor des Memory-Objekts

int2 Position im Memory-Objekt

int3 Anzahl der zu schreibenden Bytes

Resultat int Anzahl der geschriebenen Bytes   
oder Fehlerwert

Siehe Verwandte Befehle, FsiOpen()

Mit dieser Funktion werden Daten in die externe Datei (obj) ab der aktuellen Position geschrieben (siehe FsiSeek() bzw. FsiSeek64()). In (handle1) muss der Deskriptor eines Memory-Objekts angegeben werden. Aus dem Memory-Objekt werden ab der Position (int2) eine Anzahl von (int3) Bytes in die Datei geschrieben.

Sollen Zeichenketten in die externen Datei geschrieben, muss die Eigenschaft Charset des Memory-Objekts auf den Zeichensatz der externen Datei gesetzt werden, damit die Zeichenketten beim Einfügen in das Memory-Objekt (siehe MemWriteStr()) korrekt verarbeitet werden können. Eine Konvertierung der Zeichenkodierung aufgrund der Angaben bei FsiOpen() findet nicht statt.

Das Resultat gibt die Anzahl der geschriebenen Bytes zurück. Ist das Resultat negativ ist ein Fehler aufgetreten und das Resultat enthält den Fehlerwert (\_ErrFsi...). Der Fehlerwert des Betriebssystems kann über die Eigenschaft FsiError abgefragt werden.

Mögliche Laufzeitfehler:

\_ErrHdlInvalid Datei-Deskriptor (obj) oder der Deskriptor des Memory-Objekts ist ungültig.

\_ErrValueRange Der in (int2) oder (int3) übergebene Wert ist außerhalb des zulässigen Bereichs.

Konstanten für Dateibefehle (Extern)

Konstanten für Dateibefehle (Extern)

Siehe Dateibefehle  
(Extern)

- ComprFmtDeflate
- ComprFmtGzip
- ComprFmtZlib
- ComprLvlDefault
- FsiAcsR
- FsiAcsRW
- FsiAcsW
- FsiANSI
- FsiAppend
- FsiAttrArchive
- FsiAttrDir
- FsiAttrExec
- FsiAttrHidden
- FsiAttrRead
- FsiAttrSystem
- FsiAttrWrite
- FsiBuffer
- FsiCompressFast
- FsiCompressMed
- FsiCompressSlow
- FsiCompressStd
- FsiCreate
- FsiCreateNew
- FsiDecode
- FsiDeleteOnClose
- FsiDenyNone
- FsiDenyR
- FsiDenyRW
- FsiDenyW
- FsiDiskAvailMB
- FsiDiskExists
- FsiDiskFree
- FsiDiskFreeMB
- FsiDiskReady
- FsiDiskTotal
- FsiDiskTotalMB
- FsiDtAccessed
- FsiDtCreated
- FsiDtModified
- FsiEncrypt
- FsiFileCRC32
- FsiFileMD5
- FsiFileRMD160
- FsiFileSHA1
- FsiFileSHA256
- FsiFileSHA384

- FsiFileSHA512
- FsiFileVersion
- FsiFileVersionHex
- FsiGroupR
- FsiGroupW
- FsiMonActionCreate
- FsiMonActionDelete
- FsiMonActionModify
- FsiMonActionRename
- FsiMonFlagsSubDirs
- FsiMonitorStart
- FsiMonitorStop
- FsiNameC16
- FsiNameE
- FsiNameN
- FsiNameNE
- FsiNameP
- FsiNamePN
- FsiNamePNE
- FsiNamePP
- FsiNameUtf8
- FsiNoCache
- FsiOtherR
- FsiOtherW
- FsiPure
- FsiStdRead
- FsiStdWrite
- FsiSyncWrite
- FsiTruncate
- FsiUserR
- FsiUserW

\_FsiAcsR  
Nur Lesezugriff  
Wert 1 /  
0x00000001

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei mit ausschließlichem Lesezugriff geöffnet werden kann.

Auf eine mit ausschließlichem Lesezugriff geöffnete Datei sind keine Schreiboperationen möglich.

\_FsiAcsRW  
Lese- und Schreibzugriff  
Wert 3 /  
0x00000003

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei mit Lese- und Schreibzugriff geöffnet werden kann.

\_FsiAcsW  
Nur Schreibzugriff  
Wert 2 /  
0x00000002

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei mit ausschließlichem Schreibzugriff geöffnet werden kann.

Auf eine mit ausschließlichem Schreibzugriff geöffnete Datei sind keine Leseoperationen möglich.

\_FsiANSI  
Lesen/Schreiben einer ANSI-Datei

Wert 33.554.432 /  
0x02000000

Verwandte

Befehle,

Siehe FsiOpen(),

FsiRead(),

FsiWrite()

Option bei FsiOpen() durch die eine automatische Zeichenwandlung beim Lesen (siehe FsiRead()) bzw. Schreiben (siehe FsiWrite()) durchgeführt werden kann. Die Option kann nicht mit \_FsiPure kombiniert werden.

\_FsiAppend  
Datei erweitern  
Wert 64 /  
0x00000040

Verwandte  
Siehe Befehle,  
FsiOpen(),  
FsiTruncate

Option bei FsiOpen() durch die eine Datei erweitert werden kann. Der Dateizeiger wird auf das Dateiende positioniert. Die Datei muss bereits existieren, ansonsten wird diese Option ignoriert.

Diese Option ersetzt somit folgende Anweisung:

FsiSeek(tHandle, FsiSize(tHandle))



FsiAttrArchive  
Archivdatei ermitteln/setzen  
Wert 32 / 0x0020

Verwandte

Siehe Befehle,

FsiAttributes()

Option bei FsiAttributes(). Mit dieser Konstanten kann das Archiv-Attribut einer Datei ermittelt oder gesetzt werden.

\_FsiAttrDir  
Verzeichnis ermitteln  
Wert 16 / 0x0010

Verwandte

Siehe Befehle,  
FsiAttributes()

Option bei FsiAttributes(). Ist dieses Dateiaattribut gesetzt, handelt es sich um ein Verzeichnis. Das Attribut kann nur gelesen werden.

FsiAttrExec  
Ausführbarkeit ermitteln (nur UNIX)  
Wert 1.024 / 0x0400

Verwandte

Siehe Befehle,  
FsiAttributes()

Option bei FsiAttributes(). Ist dieses Attribut gesetzt, handelt es sich um eine ausführbare Datei. Das Dateiattribut "Executable" existiert nur unter Linux.

\_FsiAttrHidden  
Versteckte Datei ermitteln/setzen  
Wert 2 / 0x0002

Verwandte

Siehe Befehle,

FsiAttributes()

Option bei FsiAttributes() durch die das Datei-Attribut "Versteckt" ermittelt/gesetzt werden kann.

\_FsiAttrRead  
Leseberechtigung ermitteln  
Wert 256 / 0x0100

Verwandte

Siehe Befehle,  
FsiAttributes()

Option bei FsiAttributes(). Ist dieses Dateiattribut gesetzt, kann die Datei gelesen werden. Dies ist unter Windows-Betriebssystemen immer der Fall.



Mit dieser Anweisung werden nur die Datei-Attribute überprüft. Es werden keine Verzeichnis- oder Dateirechte berücksichtigt.

\_FsiAttrSystem

Systemdatei ermitteln/setzen

Wert 4 / 0x0004

Verwandte

Siehe Befehle,

FsiAttributes()

Option bei FsiAttributes() durch die eine Markierung für eine Systemdatei ermittelt/gesetzt werden kann.

\_FsiAttrWrite  
Schreibberechtigung ermitteln/setzen  
Wert 512 / 0x0200

Verwandte

Siehe Befehle,

FsiAttributes()

Option bei FsiAttributes(). Ist dieses Dateiaattribut gesetzt, kann die Datei geschrieben werden.



Mit dieser Anweisung werden nur die Datei-Attribute überprüft. Es werden keine Verzeichnis- oder Dateirechte berücksichtigt.

\_FsiBuffer  
Gepuffertes Lesen/Schreiben

Wert 16.777.216 /  
0x01000000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die die Lese- bzw. Schreibzugriffe auf eine Datei durch einen 8 KB großen Puffer beschleunigt werden können. Dies bringt Geschwindigkeitsvorteile beim Verarbeiten vieler kleiner, sequenzieller Lese- bzw. Schreiboperationen.

Dabei ist folgendes zu beachten:

- \_FsiBuffer ist nur wirksam in Verbindung mit \_FsiAcsR oder \_FsiAcsW. Dementsprechend ist die Pufferung dann für Schreiben oder Lesen aktiv.
- Beim gepufferten Betrieb können folgende Funktionen nicht oder nur eingeschränkt verwendet werden:

FsiSeek() ist nicht möglich.

FsiSize() Das Setzen einer neuen Dateigröße ist nicht möglich, die Abfrage der aktuellen Größe liefert die Größe zum Zeitpunkt von FsiOpen().



\_ComprFmtDeflate  
Komprimierung im DEFLATE-Format  
Wert 0

Siehe FsiFileCompress(),  
MemCompress(),  
\_ComprFmtGzip,  
\_ComprFmtZlib

Diese Konstante kann bei der Anweisung FsiFileCompress() und MemCompress() angegeben werden, um den Inhalt der Datei im DEFLATE-Format (siehe RFC 1951) zu komprimieren.

\_ComprFmtGzip  
Komprimierung im GZIP-Format  
Wert 1

Siehe FsiFileCompress(),  
MemCompress(),  
\_ComprFmtDeflate,  
\_ComprFmtZlib

Diese Konstante kann bei der Anweisung FsiFileCompress() und MemCompress() angegeben werden, um den Inhalt der Datei im GZIP-Format (siehe RFC 1952) zu komprimieren.

\_ComprFmtZlib  
Komprimierung im ZLIB-Format  
Wert 2

Siehe FsiFileCompress(),  
MemCompress(),  
\_ComprFmtDeflate,  
\_ComprFmtGzip

Diese Konstante kann bei der Anweisung FsiFileCompress() und MemCompress() angegeben werden, um den Inhalt der Datei im ZLIB-Format (siehe RFC 1950) zu komprimieren.

\_ComprLvlDefault  
Standard-Kompressionsstufe  
Wert -1

Siehe FsiFileCompress(),  
MemCompress()

Mit dieser Konstante kann bei der Anweisung FsiFileCompress() und MemCompress() angegeben werden, dass die Standard-Kompressionsstufe verwendet werden soll.

\_FsiCreate  
Datei anlegen oder öffnen  
Wert 16 /  
0x00000010

Verwandte  
Befehle,  
Siehe FsiOpen(),  
\_FsiTruncate,  
\_FsiAppend

Option bei FsiOpen() durch die eine Datei mit der Größe 0 Byte angelegt werden kann.  
Existiert die Datei bereits, wird sie erweiternd geöffnet (siehe \_FsiAppend).

Soll eine bestehende Datei überschrieben werden, muss die Option \_FsiTruncate verwendet werden.

\_FsiCreateNew  
Neue Datei explizit anlegen  
Wert 268.435.456  
/ 0x10000000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei explizit angelegt werden kann. Existiert die Datei bereits, wird der Fehlerwert \_ErrFsiExists zurückgegeben.

\_FsiDecode

Datei dekomprimieren und entschlüsseln

Wert 512 / 0x0200

Verwandte

Befehle,

Siehe FsiFileProcess(),

\_FsiFileMD5,

\_FsiEncrypt

Option bei FsiFileProcess(), durch die eine externe Datei entschlüsselt und dekomprimiert werden kann.



FsiDeleteOnClose  
Datei nach dem Schließen löschen  
Wert 32.768 /  
0x00008000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei temporär angelegt werden kann. Die Datei wird nach dem Schließen mit FsiClose() wieder gelöscht.

Die Option muss mit Optionen zum Lesen und Schreiben kombiniert werden.

-  Die Datei kann nach dem Öffnen von keinem anderen Programm mehr geöffnet werden.
-  Die Option kann nur unter Windows verwendet werden. Sie wird unter Linux ignoriert.



\_FsiDenyNone  
Kein exklusiver Zugriff  
Wert 0 /  
0x00000000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei ohne exklusiven Zugriff geöffnet werden kann.

Auf eine ohne exklusivem Zugriff geöffnete Datei sind Zugriffe anderer Benutzer oder Prozesse möglich.

\_FsiDenyR  
Exklusiver Lesezugriff

Wert 4 /  
0x00000004

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei mit exklusivem Lesezugriff geöffnet werden kann.

Auf eine mit exklusivem Lesezugriff geöffnete Datei sind keine weiteren Lesezugriffe anderer Benutzer oder Prozesse möglich.

Diese Option ist unter UNIX ohne Wirkung.

\_FsiDenyRW  
Exklusiver Lese- und Schreibzugriff  
Wert 12 /  
0x0000000C

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei mit exklusivem Lese- und Schreibzugriff geöffnet werden kann.

Auf eine mit exklusivem Lese- und Schreibzugriff geöffnete Datei sind keine weiteren Lese- und Schreibzugriffe anderer Benutzer oder Prozesse möglich.

Diese Option ist unter UNIX ohne Wirkung.

\_FsiDenyW  
Exklusiver Schreibzugriff  
Wert 8 /  
0x00000008

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei mit exklusivem Schreibzugriff geöffnet werden kann.

Auf eine mit exklusivem Schreibzugriff geöffnete Datei sind keine weiteren Schreibzugriffe anderer Benutzer oder Prozesse möglich.

Diese Option ist unter UNIX ohne Wirkung.

FsiDiskAvailMB  
Verfügbare Kapazität in MB ermitteln  
Wert 36

Verwandte

Siehe Befehle,

FsiDiskInfo()

Option bei FsiDiskInfo() durch die die verfügbare Kapazität eines Datenträgers in MB ermittelt werden kann. Sind mehr als 2 PB verfügbar, wird MaxInt zurückgegeben.

Die verfügbare Kapazität wird durch Datenträgerkontingente beschränkt.

FsiDiskExists  
Datenträgerexistenz ermitteln  
Wert 3

Verwandte

Siehe Befehle,

FsiDiskInfo()

Option bei FsiDiskInfo() durch die die Existenz eines Datenträgers ermittelt werden kann.

Falls der Datenträger existiert: Resultat = 1, ansonsten Resultat = 0.

\_FsiDiskFree  
Freie Kapazität in KB ermitteln  
Wert 0

Verwandte

Siehe Befehle,  
FsiDiskInfo(),  
\_FsiDiskFreeMB

Option bei FsiDiskInfo() durch die die freie Kapazität eines Datenträgers in KB ermittelt werden kann. Sind mehr als 2 TB frei, wird \_MaxInt zurückgegeben. In diesem Fall sollte \_FsiDiskFreeMB verwendet werden.

FsiDiskFreeMB  
Freie Kapazität in MB ermitteln  
Wert 32

Verwandte

Siehe Befehle,  
FsiDiskInfo(),  
FsiDiskFree

Option bei FsiDiskInfo() durch die die freie Kapazität eines Datenträgers in MB ermittelt werden kann. Sind mehr als 2 PB frei, wird MaxInt zurückgegeben.



FsiDiskReady  
Datenträgerbereitschaft ermitteln  
Wert 2

Verwandte

Siehe Befehle,

FsiDiskInfo()

Option bei FsiDiskInfo() durch die die Bereitschaft eines Datenträgers ermittelt werden kann.

Falls der Datenträger bereit ist: Resultat = 1, ansonsten Resultat = 0.

\_FsiDiskTotal  
Gesamte Kapazität in KB ermitteln  
Wert 1

Verwandte  
Siehe Befehle,  
FsiDiskInfo(),  
\_FsiDiskTotalMB

Option bei FsiDiskInfo() durch die die gesamte Kapazität eines Datenträgers in KB ermittelt werden kann. Sind mehr als 2 TB vorhanden, wird \_MaxInt zurückgegeben. In diesem Fall sollte \_FsiDiskTotalMB verwendet werden.

\_FsiDiskTotalMB  
Gesamte Kapazität in MB ermitteln  
Wert 33

Verwandte  
Siehe Befehle,  
FsiDiskInfo(),  
\_FsiDiskTotal

Option bei FsiDiskInfo() durch die die gesamte Kapazität eines Datenträgers in MB ermittelt werden kann. Sind mehr als 2 PB vorhanden, wird \_MaxInt zurückgegeben.

\_FsiDtAccessed  
Letzten Zugriffszeitpunkt ermitteln/setzen  
Wert 0

Verwandte

Siehe Befehle,  
FsiDate(),  
FsiTime()

Option bei FsiDate() und FsiTime() durch die das Datum bzw. die Uhrzeit des letzten Zugriffs ermittelt/gesetzt werden kann.

Diese Werte sind nicht bei FAT- und UNIX-Dateisystemen verfügbar.

\_FsiDtCreated  
Erstellungszeitpunkt ermitteln/setzen  
Wert 2

Verwandte  
Siehe Befehle,  
FsiDate(),  
FsiTime()

Option bei FsiDate() und FsiTime() durch die das Datum bzw. die Uhrzeit der Erstellung ermittelt/gesetzt werden kann.

Diese Werte sind nicht bei FAT-Dateisystemen verfügbar.

FsiDtModified  
Letzten Änderungszeitpunkt ermitteln/setzen  
Wert 1

Verwandte  
Siehe Befehle,  
FsiDate(),  
FsiTime()

Option bei FsiDate() und FsiTime() durch die das Datum bzw. die Uhrzeit der letzten Änderung ermittelt/gesetzt werden kann.

Diese Werte sind bei allen Dateisystemen verfügbar.

\_FsiEncrypt

Datei verschlüsseln

Wert 256 / 0x0100

Verwandte

Befehle,

Siehe FsiFileProcess(),

\_FsiFileMD5,

\_FsiDecode

Option bei FsiFileProcess(), durch die eine externe Datei verschlüsselt werden kann.

FsiFileCRC32  
CRC32-Prüfsumme der Datei  
Wert 9

Verwandte  
Befehle,  
FsiFileInfo(),  
FsiFileProcess(),  
Siehe FsiFileRMD160,  
FsiFileSHA1,  
FsiFileSHA256,  
FsiFileSHA384,  
FsiFileSHA512

Option bei FsiFileInfo() und FsiFileProcess(). Aus der Datei wird die CRC32-Prüfsumme berechnet. Die Summe wird als Zeichenkette aus 8 Hexadezimalziffern zurückgegeben. Über diese Funktion kann bestimmt werden, ob Dateien verändert wurden.



Dieses Verfahren ist nicht kollisionssicher. Statt dessen sollte FsiFileSHA256, FsiFileSHA384 oder FsiFileSHA512 verwendet werden.



FsiFileMD5  
MD-5 Hash der Datei  
Wert 3

Verwandte  
Befehle,  
FsiFileInfo(),  
FsiFileProcess(),  
Siehe FsiFileRMD160,  
FsiFileSHA1,  
FsiFileSHA256,  
FsiFileSHA384,  
FsiFileSHA512

Option bei FsiFileInfo() und FsiFileProcess(). Aus der Datei wird mit der MD-5 Hash-Funktion eine Summe berechnet. Die Summe wird als Zeichenkette aus 32 Hexadezimalziffern zurückgegeben. Über diese Funktion kann bestimmt werden, ob Dateien verändert wurden.



Dieses Verfahren ist nicht kollisionssicher. Statt dessen sollte FsiFileSHA256, FsiFileSHA384 oder FsiFileSHA512 verwendet werden.

FsiFileRMD160  
Hash der Datei nach RIPEMD-160-Standard  
Wert 4

Verwandte  
Befehle,  
FsiFileInfo(),  
FsiFileProcess(),

Siehe FsiFileMD5,  
FsiFileSHA1,  
FsiFileSHA256,  
FsiFileSHA384,  
FsiFileSHA512

Option bei FsiFileInfo() und FsiFileProcess(). Aus der Datei wird ein Hash-Wert nach RIPEMD-160-Standard berechnet. Der Wert wird als Zeichenkette aus 40 Hexadezimalziffern zurückgegeben. Über diese Funktion kann bestimmt werden, ob Dateien verändert wurden.

Im Vergleich zum MD5-Hash-Wert dauert die Ermittlung des Hashwerts praktisch nicht länger, die Sicherheit ist jedoch höher, da im Gegensatz zu MD5 bisher keine Berechnung von Kollisionen möglich ist. Dadurch wird ein besserer Schutz vor Manipulationen der Dateiinhalte erreicht.

FsiFileSHA1  
SHA-1-Hash der Datei  
Wert 5

Verwandte  
Befehle,  
FsiFileInfo(),  
FsiFileProcess(),

Siehe FsiFileMD5,  
FsiFileRMD160,  
FsiFileSHA256,  
FsiFileSHA384,  
FsiFileSHA512

Option bei FsiFileInfo() und FsiFileProcess(). Aus der Datei wird ein Hash-Wert nach SHA-1-Standard berechnet. Der Wert wird als Zeichenkette aus 40 Hexadezimalziffern zurückgegeben. Über diese Funktion kann bestimmt werden, ob Dateien verändert wurden.

FsiFileSHA256  
SHA-256-Hash der Datei  
Wert 6

Verwandte  
Befehle,  
FsiFileInfo(),  
FsiFileProcess(),

Siehe FsiFileMD5,  
FsiFileRMD160,  
FsiFileSHA1,  
FsiFileSHA384,  
FsiFileSHA512

Option bei FsiFileInfo() und FsiFileProcess(). Aus der Datei wird ein 256-Bit-Hash-Wert nach SHA-2-Standard berechnet. Der Wert wird als Zeichenkette aus 64 Hexadezimalziffern zurückgegeben. Über diese Funktion kann bestimmt werden, ob Dateien verändert wurden.

FsiFileSHA384  
SHA-384-Hash der Datei  
Wert 7

Verwandte  
Befehle,  
FsiFileInfo(),  
FsiFileProcess(),

Siehe FsiFileMD5,  
FsiFileRMD160,  
FsiFileSHA1,  
FsiFileSHA256,  
FsiFileSHA512

Option bei FsiFileInfo() und FsiFileProcess(). Aus der Datei wird ein 384-Bit-Hash-Wert nach SHA-2-Standard berechnet. Der Wert wird als Zeichenkette aus 96 Hexadezimalziffern zurückgegeben. Über diese Funktion kann bestimmt werden, ob Dateien verändert wurden.

FsiFileSHA512  
SHA-512-Hash der Datei  
Wert 8

Verwandte  
Befehle,  
FsiFileInfo(),  
FsiFileProcess(),

Siehe FsiFileMD5,  
FsiFileRMD160,  
FsiFileSHA1,  
FsiFileSHA256,  
FsiFileSHA384

Option bei FsiFileInfo() und FsiFileProcess(). Aus der Datei wird ein 512-Bit-Hash-Wert nach SHA-2-Standard berechnet. Der Wert wird als Zeichenkette aus 128 Hexadezimalziffern zurückgegeben. Über diese Funktion kann bestimmt werden, ob Dateien verändert wurden.

FsiFileVersion  
Version der Datei ermitteln  
Wert 1

Verwandte

Siehe Befehle,  
FsiFileInfo()

Option bei FsiFileInfo(). Wird diese Option angegeben, wird die Version der Datei zurückgegeben. Die einzelnen Zahlen der Version sind durch Punkte voneinander getrennt. Unter Linux-Systemen wird eine leere Zeichenkette zurückgegeben.

FsiFileVersionHex  
Versionsnummer der Datei im Hexadezimalformat  
Wert 2

Verwandte

Siehe Befehle,

FsiFileInfo()

Option bei FsiFileInfo(). Wird diese Option angegeben besteht der Rückgabewert aus 16 Hexadezimalzahlen. Jeweils vier Ziffern bilden einen Teil der Versionsnummer. Die einzelnen Blöcke sind nicht voneinander getrennt. Unter Linux-Systemen wird eine leere Zeichenkette zurückgegeben.



\_FsiGroupR  
Leseberechtigung für Gruppe  
Wert 262.144 /  
0x00040000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die einer Datei beim Anlegen eine Leseberechtigung für die Benutzergruppen zugeteilt werden kann, sofern die gesetzte UMASK dies erlaubt.

Diese Option ist nur unter UNIX-Systemen relevant.

\_FsiGroupW  
Schreibberechtigung für Gruppe  
Wert 524.288 /  
0x00080000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die einer Datei beim Anlegen eine Schreibberechtigung für die Benutzergruppen zugeteilt werden kann, sofern die gesetzte UMASK dies erlaubt.

Diese Option ist nur unter UNIX-Systemen relevant.

\_FsiMonActionCreate

Datei angelegt

Wert 1

Siehe EvtFsiMonitor

Übergabeparameter bei dem Ereignis EvtFsiMonitor. In dem überwachten Verzeichnis wurde eine Datei angelegt.

## Kontakt

\_FsiMonActionDelete

Datei wurde gelöscht

Wert 2

Siehe EvtFsiMonitor

Übergabeparameter bei dem Ereignis EvtFsiMonitor. Eine Datei im überwachten Verzeichnis wurde gelöscht.

## Kontakt

\_FsiMonActionModify  
Datei wurde verändert  
Wert 3

Siehe EvtFsiMonitor

Übergabeparameter bei dem Ereignis EvtFsiMonitor. Eine Datei im überwachten Verzeichnis wurde verändert.

## Kontakt

\_FsiMonActionRename  
Datei wurde umbenannt  
Wert 4

Siehe EvtFsiMonitor

Übergabeparameter bei dem Ereignis EvtFsiMonitor. Eine Datei im überwachten Verzeichnis wurde umbenannt.

FsiMonFlagsSubDirs

Untergeordnete Verzeichnisse mitüberwachen

Wert 1 / 0x0001

Siehe EvtFsiMonitor,  
FsiMonitorAdd()

Übergabeparameter bei dem Befehl FsiMonitorAdd(). Neben dem angegebenen Verzeichnis werden auch alle untergeordneten Verzeichnisse überwacht.

\_FsiMonitorStart

Verzeichnisüberwachung starten

Wert 1

Siehe FsiMonitorControl()

Übergabeparameter bei dem Befehl FsiMonitorControl(). Die Überwachung des Verzeichnisses wird gestartet.



FsiMonitorStop

Verzeichnisüberwachung anhalten

Wert 2

Siehe FsiMonitorControl()

Übergabeparameter bei dem Befehl FsiMonitorControl(). Die Überwachung des Verzeichnisses wird ausgesetzt.

FsiNameC16  
Dateiname im CONZEPT 16-Zeichensatz  
Wert 2.048 /  
0x00000800

Verwandte  
Befehle,  
Siehe FsiOpen(),  
FsiDirOpen(),  
FsiAttributes(),  
FsiNameUtf8

Option bei FsiOpen(), FsiDirOpen() und FsiAttributes(), mit der Dateinamen im CONZEPT 16-Zeichensatz interpretiert werden. Die Option kann nicht mit FsiNameUtf8 kombiniert werden.

\_FsiNameE

Erweiterung ermitteln

Wert 6 / 0x06

Verwandte

Siehe Befehle,

FsiSplitName()

Option bei FsiSplitName() durch die aus einem Dateiname/-pfad die Erweiterung ermittelt werden kann.

**Beispiel:**

```
FsiSplitName('C:\TEST\INFO.ASC', _FsiNameE) // 'ASC'FsiSplitName('INFO.ASC', _FsiNameE)
```

\_FsiNameN

Datei ermitteln

Wert 5 / 0x05

Verwandte

Siehe Befehle,

FsiSplitName()

Option bei FsiSplitName() durch die aus einem Dateiname/-pfad die Datei ermittelt werden kann.

**Beispiel:**

```
FsiSplitName('C:\TEST\INFO.ASC', _FsiNameN) // 'INFO' FsiSplitName('INFO.ASC', _FsiNameN)
```

\_FsiNameNE

Datei und Erweiterung ermitteln

Wert 4 / 0x04

Verwandte

Siehe Befehle,

FsiSplitName()

Option bei FsiSplitName() durch die aus einem Dateiname/-pfad die Datei und die Erweiterung ermittelt werden können.

**Beispiel:**

```
FsiSplitName('C:\TEST\INFO.ASC', _FsiNameNE) // 'INFO.ASC'FsiSplitName('INFO.ASC', _FsiNameNE)
```

\_FsiNameP

Laufwerk und Pfad mit Separator ermitteln

Wert 3 / 0x03

Verwandte

Siehe Befehle,

FsiSplitName()

Option bei FsiSplitName() durch die aus einem Dateiname/-pfad das Laufwerk und der Pfad mit Separator ermittelt werden können.

**Beispiele:**

```
FsiSplitName('C:\TEST\INFO.ASC', _FsiNameP) // 'C:\TEST\FsiSplitName('INFO.ASC', _FsiNameP)
```

\_FsiNamePN

Laufwerk, Pfad und Datei ermitteln

Wert 1 / 0x01

Verwandte

Siehe Befehle,

FsiSplitName()

Option bei FsiSplitName() durch die aus einem Dateiname/-pfad das Laufwerk, der Pfad und die Datei ermittelt werden können.

**Beispiel:**

```
FsiSplitName('C:\TEST\INFO.ASC', _FsiNamePN) // 'C:\TEST\INFO'FsiSplitName('INFO.ASC', _FsiNamePN)
```

\_FsiNamePNE

Laufwerk, Pfad, Datei und Erweiterung ermitteln

Wert 0 / 0x00

Verwandte

Siehe Befehle,

FsiSplitName()

Option bei FsiSplitName() durch die aus einem Dateiname/-pfad das Laufwerk, der Pfad, die Datei und die Erweiterung ermittelt werden können.

Der übergebene Dateiname/-pfad bleibt unverändert.

**Beispiel:**

```
FsiSplitName('C:\TEST\INFO.ASC', _FsiNamePNE) // 'C:\TEST\INFO.ASC' FsiSplitName('INFO.ASC', _FsiNamePNE)
```



\_FsiNamePP

Laufwerk und Pfad ohne Separator ermitteln

Wert 11 / 0x0B

Verwandte

Siehe Befehle,

FsiSplitName()

Option bei FsiSplitName() durch die aus einem Dateiname/-pfad das Laufwerk und der Pfad ohne Separator ermittelt werden können.

**Beispiel:**

```
FsiSplitName('C:\TEST\INFO.ASC', _FsiNamePP) // 'C:\TEST'FsiSplitName('INFO.ASC', _FsiNamePP)
```

\_FsiNameUtf8  
Dateiname im UTF-8-Zeichensatz  
Wert 536.870.912 /  
0x20000000

Verwandte  
Befehle,  
Siehe FsiOpen(),  
FsiDirOpen(),  
FsiAttributes(),  
\_FsiNameC16

Option bei FsiOpen(), FsiDirOpen() und FsiAttributes(), mit der Dateinamen im UTF-8-Zeichensatz interpretiert werden. Die Option kann nicht mit \_FsiNameC16 kombiniert werden.

FsiNoCache  
Ungepuffertes Lesen/Schreiben  
Wert 256 /  
0x00000100

Verwandte

Siehe Befehle,

FsiOpen()

Option bei FsiOpen() durch die ein Zugriff auf den Cache beim Lesen bzw. Schreiben umgangen werden kann.

\_FsiOtherR  
Leseberechtigung für Andere  
Wert 1.048.576 /  
0x00100000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die einer Datei beim Anlegen eine Leseberechtigung für andere Benutzer zugeteilt werden kann, sofern die gesetzte UMASK dies erlaubt.

Diese Option ist nur unter UNIX-Systemen relevant.

\_FsiOtherW  
Schreibberechtigung für Andere  
Wert 2.097.152 /  
0x00200000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die einer Datei beim Anlegen eine Schreibberechtigung für andere Benutzer zugeteilt werden kann, sofern die gesetzte UMASK dies erlaubt.

Diese Option ist nur unter UNIX-Systemen relevant.

\_FsiPure  
Keine Zeichenwandlung beim Lesen oder Schreiben einer Datei  
Wert 134.217.728  
/ 0x08000000

Verwandte

Befehle,

Siehe FsiOpen(),

FsiRead(),

FsiWrite()

Option bei FsiOpen(). Standardmäßig erfolgt eine Umwandlung zwischen der internen und der OEM-Zeichencodierung beim Lesen (siehe FsiRead()) bzw. Schreiben (siehe FsiWrite()) von bzw. in eine externe Datei. Durch die Angabe dieser Konstanten wird keine Umwandlung durchgeführt, sodass bereits konvertierte Zeichenketten geschrieben werden können. Die Option kann nicht mit \_FsiANSI kombiniert werden.

\_FsiStdRead  
Standard-Lesemodus

Wert 521 /  
0x00000209

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei im Standard-Lesemodus geöffnet werden kann.

Diese Option entspricht folgender Zugriffsoption:

\_FsiAcsR | \_FsiDenyW

\_FsiStdWrite  
Standard-Schreibmodus

Wert 574 /  
0x0000023E

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die eine Datei im Standard-Schreibmodus geöffnet werden kann.

Diese Option entspricht folgender Zugriffsoption:

\_FsiAcsW | \_FsiDenyRW | \_FsiCreate | \_FsiTruncate



FsiSyncWrite  
Synchrones Schreiben

Wert 128 /  
0x00000080

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die nachfolgende Schreiboperationen synchron ausgeführt werden können. Das bedeutet, dass die Schreiboperation erst dann beendet ist, wenn die Daten physikalisch auf dem Datenträger geschrieben wurden. Dies kann Schreiboperationen erheblich verlangsamen.

Diese Option ist nicht bei allen Betriebssystemen wirksam. Insbesondere in Netzwerkkumgebungen wird ein synchrones Schreiben nicht immer unterstützt.

Synchrones Schreiben ist zur Zeit nur unter Windows-Systemen möglich.

\_FsiTruncate  
Datei leeren  
Wert 32 /  
0x00000020

Verwandte  
Siehe Befehle,  
FsiOpen(),  
\_FsiAppend

Option bei FsiOpen() durch die eine Datei auf die Größe 0 Byte geleert werden kann.  
Die Datei muss bereits existieren, ansonsten wird diese Option ignoriert.

Soll eine neue Datei angelegt werden, muss die Option \_FsiCreate verwendet werden.

\_FsiUserR  
Leseberechtigung für Benutzer  
Wert 65.536 /  
0x00010000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die einer Datei beim Anlegen eine Leseberechtigung für den Benutzer zugeteilt werden kann, sofern die gesetzte UMASK dies erlaubt.

Diese Option ist nur unter UNIX-Systemen relevant.

\_FsiUserW  
Schreibberechtigung für Benutzer  
Wert 131.072 /  
0x00020000

Verwandte

Siehe Befehle,  
FsiOpen()

Option bei FsiOpen() durch die einer Datei beim Anlegen eine Schreibberechtigung für den Benutzer zugeteilt werden kann, sofern die gesetzte UMASK dies erlaubt.

Diese Option ist nur unter UNIX-Systemen relevant.

Systemfunktionen

Befehle für Systemfunktionen

Siehe Befehlsgruppen,  
Befehlsliste

## **Befehle**

- ProcCacheClear
- ProcCompile
- RmtDataRead
- RmtDataSearch
- RmtDataWrite
- SysBeep
- SysClone
- SysDate
- SysExecute
- SysGetArg
- SysGetEnv
- SysOS
- SysSleep
- SysTicks
- SysTime
- SysTimerClose
- SysTimerCreate

## **Konstanten**

- CloneAdvanced
- CloneMaximized
- CloneMinimized
- CloneStandard
- ExecHidden
- ExecMaximized
- ExecMinimized
- ExecWait
- RmtDataTemp
- TimeHSec
- TimeSec
- TimeServer

ProcCompile(alpha1) : int  
 Prozedur zur Laufzeit übersetzen



alpha1 Name der  
 Prozedur

Resultat int Fehlerwert 

Siehe ProcCacheClear(),  
System

Mit dieser Anweisung wird die in (alpha1) übergebene Prozedur übersetzt. Als Resultat wird der Fehler beim Übersetzen zurückgegeben. Bei der Rückgabe von ErrOk konnte die Prozedur übersetzt werden. Der Fehlerwert steht ebenfalls in der Eigenschaft ErrCode zur Verfügung. Er kann mit einer der ErrCpl...-Konstanten verglichen werden.

Ist bei der Übersetzung ein Fehler aufgetreten, können weitere Informationen über folgende Eigenschaften des System-Objekts ermittelt werden:

<u>ErrCode</u>	Fehlerwert
<u>ErrText</u>	Fehlertext
<u>ErrProc</u>	Name der übersetzten Prozedur
<u>ErrSource</u>	Name der Quelltextprozedur (kann vom Namen der übersetzten Prozedur abweichen, wenn eine Prozedur mit der <u>Include</u> -Anweisung eingebunden wird)
<u>ErrSourceLine</u>	Quelltextzeile in der der Fehler aufgetreten ist
<u>ErrLine</u>	Nummer der Fehlerzeile im Quelltext
<u>ErrPos</u>	Fehlerposition in der Zeile

Unter folgenden Umständen kann eine Prozedur **nicht** übersetzt werden:

- Die Prozedur enthält ein with-Konstrukt.
- Die Prozedur wird gerade ausgeführt oder ist von einem anderen Client gesperrt.
- Die Prozedur ist keine A+ Prozedur (siehe @A+).

Falls die übersetzte Prozedur globale Datenbereiche (siehe global) enthält, dürfen diese nicht verändert werden, da die Änderungen nicht in die Runtime-Umgebung übernommen werden können und somit die Gefahr von Laufzeitfehlern bei Prozeduren, die diese Datenbereiche verwenden entsteht.



Falls Elemente der Prozedur geändert werden, die von anderen Prozeduren verwendet werden (andere Argumente bei Funktionen, Modifikationen von globalen Datenbereichen oder ähnliches) kann dies zu Laufzeitfehlern oder im ungünstigsten Fall zum Beenden des Clients führen. Dies gilt auch für alle anderen aktiven Clients der Datenbank, sofern diese die übersetzte Prozedur verwenden.

Sofern die übersetzte Prozedur auf anderen Arbeitsstationen aufgerufen wird, müssen in diesen Clients eventuell vorhandene ältere Codefragmente der übersetzten Prozedur aus dem Prozedurcache entfernt werden. Andernfalls können Laufzeitfehler entstehen. Dies wird mit dem Befehl ProcCacheClear() durchgeführt.

ProcCacheClear(alpha1)



Prozedur aus Cache entfernen

alpha1 Name der  
Prozedur

Siehe ProcCompile()

Diese Anweisung entfernt die in (alpha1) übergebene Prozedur aus dem Prozedurcache des Clients. Die Funktion muss aufgerufen werden, wenn die Prozedur von einem anderen Client neu übersetzt wurde. Wird die Prozedur nicht aus dem Cache entfernt, werden möglicherweise Fragmente aus der alten Prozedur ausgeführt, was zu Laufzeitfehlern oder im ungünstigsten Fall zum Beenden des Clients führen kann.

SysBeep(int1, int2)



Akkustiksignal generieren

int1    Signalfrequenz in  
Hertz

int2    Signaldauer in  
Millisekunden

Verwandte

Befehle,

Siehe WinBeep(),

Benutzerdefinierte

Sounds (Blog)

Mit diesem Befehl kann ein akustisches Signal erzeugt werden.


### **Beispiel:**

```
SysBeep(440, 500) // Erzeugt eine halbe Sekunde lang einen 440 Hz Ton.
```



Wird der Befehl vom Server durchgeführt, kann abhängig von der Plattform die Tonhöhe und -dauer nicht beeinflusst werden.



SysClone(int1[, alpha2[,  
alpha3[, alpha4[, alpha5]]]]) :   
int

Neuen Client-Prozess starten

Optionen


	<u>_CloneMinimized</u>	Neuen Client minimiert starten
int1	<u>_CloneMaximized</u>	Neuen Client maximiert starten
	<u>_CloneStandard</u>	Standard-Client starten
	<u>_CloneAdvanced</u>	Advanced-Client starten

alpha2 abweichende Startprozedur

alpha3 Kommandozeilenargumente

alpha4 abweichender Benutzer

alpha5 Passwort für Benutzer

Resultat int Prozess-ID oder  
Fehlerwert 

Siehe Verwandte Befehle

Mit diesem Befehl wird ein neuer Client-Prozess gestartet.

Folgende Optionen (int1) können angegeben werden:

\_CloneMinimized Der neue Client wird minimiert gestartet.

\_CloneMaximized Der neue Client wird maximiert gestartet.

\_CloneStandard Es wird der Standard-Client gestartet. Ist die Datenbank auf Stand 3.x wird der Client für 3.x-Datenbanken verwendet.

\_CloneAdvanced Es wird der Advanced-Client gestartet.

Die Optionen zur Größe (\_CloneMinimized und \_CloneMaximized) können mit den Optionen zum Clienttyp (\_CloneStandard und \_CloneAdvanced) kombiniert werden, jedoch nicht untereinander.

Ist keine der Optionen \_CloneStandard und \_CloneAdvanced angegeben, wird der aktuelle Clienttyp gestartet.

Es kann eine abweichende Startprozedur (alpha2) angegeben werden. Ist diese nicht angegeben, wird die Startprozedur ausgeführt, die in der Benutzerverwaltung für den Benutzer hinterlegt ist.

In dem Parameter (alpha3) können Kommandozeilenargumente angegeben werden. Bisherige Argumente können mit SysGetArg() ermittelt werden.

Wird ein Benutzer (alpha4) angegeben, wird dieser zum Start des Clients verwendet. In diesem Fall muss, sofern für den Benutzer ein Passwort definiert ist, sein Passwort (alpha5) angegeben werden. Ist kein Benutzer angegeben, wird der aktuelle Benutzer

verwendet.

Im Fehlerfall wird \_ErrGeneric zurückgegeben, andernfalls die Prozess-ID des gestarteten Prozesses.

### **Beispiele:**

```
// Neuen Client abhängig vom aktuellen Clienttyp startenSysClone(0);// Standard-Client unabhängig
```

SysDate() : date



Systemdatum ermitteln

Resultat date Systemdatum

Verwandte

Befehle,

Siehe Datumsfunktionen,


SysTime(),

SysTics(), caltime

Mit diesem Befehl kann das aktuelle Systemdatum ermittelt werden.



**Die Anweisung ist nur noch aus Kompatibilitätsgründen im Sprachumfang enthalten. Es sollte die Methode vmSystemTime() oder vmServerTime() verwendet werden.**

SysExecute(alpha1, ,  
 alpha2, int3) : int  
 Programm starten  
 alpha1 Programmname und -pfad  
 alpha2 Programmparameter  
 Optionen  
     ExecWait           Synchrone  
                           Programmverarbeitung  
     ExecMaximized Programm maximiert  
 int3                    starten  
     ExecMinimized Programm minimiert  
                           starten  
     ExecHidden       Programm versteckt  
                           starten  
 Startresultat  
 Programmrückgabewert (in  
 Verbindung mit ExecWait)  
 Resultat int ErrOk     Starten erfolgreich  
               ErrGeneric Programm (alpha1)  
                           kann nicht  
                           gefunden/gestartet  
                           werden.

#### Siehe Verwandte Befehle

Mit diesem Befehl wird ein Programm gestartet, in (alpha1) wird der Pfad und der Name des Programms angegeben und in (alpha2) die Parameter, die an das aufzurufende Programm übergeben werden sollen. Sollen keine Parameter übergeben werden, kann in (alpha2) " übergeben werden.



Die Anweisung kann von der DLL-Schnittstelle, jedoch nicht von der PHP-Schnittstelle ausgeführt werden.

#### Beispiel:

```
// Starten des Windows-TaschenrechnerSysExecute('Calc', '', 0);
```

Über SysExecute() ist auch das direkte Öffnen von registrierten Dateitypen möglich. In diesem Fall wird in (alpha1) ein Stern (\*) gefolgt vom Namen des Dokumentes angegeben.

#### Beispiel:

```
// Starten eines Word-DokumentsSysExecute('*D:\Doc\ReadMe.doc', '', 0);
```

Um Befehle der Shell des Betriebssystems ausführen zu können, muss in (alpha1) der Windows-Befehlsprozessor durch die Angabe von 'cmd' gestartet werden. In (alpha2) wird der Shell-Befehl inklusive Parameter angegeben. Damit der Shell-Befehl auch ausgeführt wird, ist in (alpha2) zusätzlich die Angabe der cmd-Parameter '/c' oder '/k' (Optionen des Befehlsprozessors) notwendig. Eine Übersicht der Parameter des Befehlsprozessors kann mit "help cmd" in der Kommandozeile abgerufen werden.

**Beispiel:**

```
// Ausführen des Shell-Befehls copySysExecute('cmd', '/c copy ' + _Sys->spPathTemp + '\a.dat ' +
```

In (int3) können folgende symbolische Konstanten übergeben werden:

- ExecWait

Das Programm wird solange angehalten, bis das aufgerufene Programm beendet wurde. Ohne die Angabe dieses Parameters werden beide Programme fortgesetzt.

- ExecMaximized

Das Programm wird maximiert gestartet.

- ExecMinimized

Das Programm wird minimiert gestartet.

- ExecHidden

Das Programm wird versteckt gestartet.



SysGetArg(alpha1) : alpha  
Kommandozeilenargumente ermitteln  
alpha1 Argumentname  
Resultat alpha Argumentwert

Siehe [Verwandte Befehle](#),  
[SysGetEnv\(\)](#), [Blog](#)

Auf der Kommandozeile bzw. in den Startparametern von CONZEPT 16 können anwendungsspezifische Argumente in folgender Form angegeben werden:

```
/<Argumentname>=<Argumentwert>
```

Mit SysGetArg() kann dann der Argumentwert des Arguments (alpha1) ermittelt werden. Ist kein Argument mit diesem Namen vorhanden, wird ein leeres Resultat geliefert.

Die Argumente [c16](#), [c16cfg](#), [c16tmp](#), [c16lang](#) und [c16splashon](#) (siehe [FAQ](#)) lassen sich mit dieser Funktion ebenfalls ermitteln.

### **Beispiel:**

```
// CONZEPT 16 wurde mit dem Argument '/UserMode=Pflege' aufgerufen.tArg # SysGetArg('UserMode');
```



SysGetEnv(alpha1) : alpha

Werte von Umgebungsvariablen ermitteln

alpha1 Variablenname

Resultat alpha Variablenwert

Verwandte Befehle,

Siehe Dateipfade mit

Umgebungsvariablen

(Blog)

Mit dieser Funktion wird der Wert einer Umgebungsvariable ermittelt. Das Resultat ist leer, wenn kein Eintrag mit dem Namen (alpha1) vorhanden ist. Einige der Systemvariablen können auch über Eigenschaften des System-Objekts ermittelt werden.

### **Beispiel:**

```
SysGetEnv('TEMP') // Liefert das Temporärverzeichnis zurück.
```

SysOS([logic1]) : alpha 

Betriebssystem ermitteln

logic1    Buildnummer ermitteln  
(optional)

Resultat alpha Betriebssystemname  
(siehe Text)

Siehe Verwandte Befehle

Dieser Befehl ermittelt den Namen des aktuellen Betriebssystems.

Als Resultat können folgende Werte zurückgegeben werden:

'Windows XP' 'Windows XP (64-Bit)' 'Windows Server 2003' 'Windows Server 2003 (64-Bit)' 'Windows Serv

Wird im optionalen Argument (logic1) true angegeben, wird unter Windows zusätzlich die Buildnummer ermittelt. Das Resultat kann beispielsweise wie folgt aussehen:

'Windows 10 (Build: 16299)' 'Windows 10 (64-Bit, Build: 16299)'



Wird die Funktion SysOS() über RmtCall() auf einem Linux-Server ausgeführt, besteht das Resultat aus Informationen über den Linux-Kernel.

### Beispiel

'Linux kernel 5.2 (64-Bit)'



SysSleep(int1)



Verarbeitung anhalten

Anhaltedauer

int1 in

Millisekunden

Verwandte

Siehe Befehle,

WinSleep()

Diese Funktion hält die Verarbeitung in einer Prozedur für eine bestimmte Dauer (int1) in Millisekunden an.

Hier kann ebenfalls eine Wartezeit von 0 Millisekunden angegeben werden. Mit diesem Kommando wird die aktuelle Zeitscheibe des Prozesses freigegeben und kann durch das Betriebssystem an einen anderen Prozess vergeben werden.

SysTics() : int



Systemlaufzeit ermitteln

Resultat int Systemlaufzeit in  
Millisekunden

Verwandte Befehle,

Siehe vmSystemTime(),  
vmServerTime()

Mit diesem Befehl kann unter Windows-Betriebssystemen die Zeit seit dem Systemstart in Millisekunden ermittelt werden. Unter Linux ist der ermittelte Wert unabhängig vom Systemstart, dafür abhängig von Datum und Uhrzeit.

Der Wert steht nicht in Beziehung zur Uhrzeit. Mit dem Resultat von SysTics() kann ein Zeitintervall relativ exakt bestimmt werden. Es ist zu beachten, dass auf Windows Rechnern die Zeitauflösung ca. 10 Millisekunden beträgt.

Der Wert wird vom Betriebssystem ohne Vorzeichen zurückgeben, in CONZEPT 16 aber mit Vorzeichen interpretiert. Zur Vereinfachung von Berechnungen sind nur positive Werte zulässig. Läuft das System länger als ca. 24 Tage 20 Stunden, fängt der Wert wieder bei 0 an.



Um eine genauere Zeitauflösung zu erhalten sollte ein 64-Bit-Zeitstempel mit vmSystemTime() ermittelt werden.

### Beispiel:

```
tTimeTics # SysTics();
```

Berechnung des Zeitabstandes zwischen zwei mit SysTics() ermittelten Werten.

```
define{ // Differenz mit Beachtung des Übertrags Sys.TicsDiff(aTicsBegin, aTicsEnd) : (int(aTicsEnd - aTicsBegin) + (aTicsBegin > 0 ? 1 : 0))
```

SysTime(int1) : time  
Systemzeit ermitteln



Optionen

	<u>_TimeSec</u>	Zeit mit Sekunden ermitteln
int1	<u>_TimeHSec</u>	Zeit mit Hundertstelsekunden ermitteln
	<u>_TimeServer</u>	Systemzeit des Servers ermitteln

Resultat time Systemzeit

Verwandte Befehle,

Siehe Zeitfunktionen, SysDate(),  
SysTics(), caltime

Mit diesem Befehl kann die aktuelle Systemzeit ermittelt werden.



**Die Anweisung ist nur noch aus Kompatibilitätsgründen im Sprachumfang enthalten. Um die aktuelle Uhrzeit zu ermitteln sollte die Methode vmSystemTime() oder vmServerTime() verwendet werden.**

Mit der Option \_TimeServer kann statt der Uhrzeit des eigenen Rechners die Uhrzeit des Servers ermittelt werden. Die Systemzeit des eigenen Rechners bleibt dabei unverändert. Diese Option kann mit \_TimeSec und \_TimeHSec kombiniert werden.

### Beispiele:

```
SysTime(_TimeSec);SysTime(_TimeServer | _TimeSec);
```

obj -> SysTimerClose()



Zeitgesteuertes Ereignis beenden

obj Timer-Deskriptor

Verwandte

Befehle,

Siehe SysTimerCreate(),

EvtTimer

Mit dieser Funktion wird ein mit SysTimerCreate() erzeugter Timer beendet. Der Timer-Deskriptor ist danach nicht mehr gültig.

Mögliche Laufzeitfehler:

\_ErrHdlInvalid Timer-Deskriptor (obj) ungültig

SysTimerCreate(int1, int2[,  
handle3]) : handle



Zeitgesteuertes Ereignis starten

int1      Zeitintervall in  
            Millisekunden

int2      Wiederholungsanzahl

handle3   Zielfenster-Deskriptor  
            (optional)

Resultat handle Timer-Deskriptor

Verwandte Befehle,

Siehe      SysTimerClose(),  
            EvtTimer

Mit dieser Funktion wird in einem bestimmten Intervall (int1) in Millisekunden das Ereignis EvtTimer ausgelöst. Der Minimalwert von (int1) beträgt 100 ms. Bei Angabe von -1 in (int2) wird das Ereignis unbegrenzt wiederholt, ansonsten enthält (int2) die Anzahl der auszulösenden Ereignisse. Bei (int2) gleich 0 wird kein Ereignis ausgelöst.

In (handle3) kann der Deskriptor des Frame-Objekts angegeben werden, welches das Ereignis erhält. Wird (handle3) nicht angegeben oder auf 0 gesetzt, erhalten alle Top-Level-Frames (die Frames ohne Parent) das Ereignis. Eine entsprechende Prozedurfunktion für das Ereignis EvtTimer muss beim jeweiligen Frame angegeben werden.

Das Resultat ist der Deskriptor des Timers.

Das Zeitintervall beginnt erst nach der kompletten Verarbeitung von EvtTimer erneut. Damit wird verhindert, dass während der Ereignisverarbeitung bereits ein weiteres Ereignis ausgelöst wird.

Die maximale Anzahl von Timern ist nicht beschränkt. Es ist zu beachten, dass ein Timer immer mit SysTimerClose() entfernt werden muss, auch wenn er keine Ereignisse mehr auslöst.

### Beispiel:

```
// Ein Timer auf den Dialog $Frame, mit// einem Intervall von 300 ms und einer// unendlichen Wiederholung
```

Konstanten für Systemfunktionen  
Konstanten für Systemfunktionen  
Siehe Systemfunktionen

- \_CloneAdvanced
- \_CloneMaximized
- \_CloneMinimized
- \_CloneStandard
- \_ExecHidden
- \_ExecMaximized
- \_ExecMinimized
- \_ExecWait
- \_RmtDataTemp
- \_TimeHSec
- \_TimeSec
- \_TimeServer

\_CloneAdvanced

Neuen Client als Advanced-Client starten

Wert 2 / 0x0002

Verwandte

Siehe Befehle,

SysClone()

Option bei SysClone() durch die der Advanced-Client gestartet werden kann.

\_CloneMaximized

Neuen Client maximiert starten

Wert 32 / 0x0020

Verwandte

Siehe Befehle,

SysClone(),

\_CloneMinimized

Option bei SysClone() durch die der Client mit maximierter Darstellung gestartet werden kann.



\_CloneMinimized

Neuen Client minimiert starten

Wert 16 / 0x0010

Verwandte

Siehe Befehle,

SysClone(),

\_CloneMaximized

Option bei SysClone() durch die der Client mit minimierter Darstellung gestartet werden kann.

\_CloneStandard

Neuen Client als Standard-Client starten

Wert 1 / 0x0001

Verwandte

Siehe Befehle,

SysClone()

Option bei SysClone() durch die der Standard-Client gestartet werden kann.

ExecHidden  
Programm versteckt ausführen  
Wert 6 /  
0x00000006

Verwandte

Siehe Befehle,

SysExecute()

Option bei SysExecute() durch die das Programm versteckt ausgeführt werden kann.



Diese Option wird nicht von allen Programmen unterstützt, vornehmlich aber von Kommandozeilen-basierten Anwendungen.

\_ExecMaximized  
Programm maximiert starten  
Wert 4 / 0x00000004

Verwandte  
Siehe Befehle,  
SysExecute(),  
\_ExecMinimized

Option bei SysExecute() durch die ein Programm mit maximierter Darstellung gestartet werden kann.

\_ExecMinimized  
Programm minimiert starten  
Wert 2 / 0x00000002

Verwandte  
Siehe Befehle,  
SysExecute(),  
\_ExecMaximized

Option bei SysExecute() durch die ein Programm mit minimierter Darstellung gestartet werden kann.

\_ExecWait  
Synchrone Programmverarbeitung  
Wert 1 /  
0x00000001

Verwandte

Siehe Befehle,

SysExecute()

Option bei SysExecute() durch die die Verarbeitung der Prozedur bis zum Beenden des gestarteten Programms angehalten werden kann.

RmtDataTemp

Aufbewahrung des zentralen Datenobjekts bis zur Abmeldung

Wert 1.073.741.824 /

0x40000000

Siehe RmtDataWrite()

Die Option wird beim Befehl RmtDataWrite() angegeben. Die geschriebenen Daten bleiben erhalten, bis sich der Benutzer, der die Daten geschrieben hat, von der Datenbank abmeldet. Wird die Option nicht angegeben, bleiben die Daten erhalten, bis die Datenbank geschlossen wird.

\_TimeHSec

Zeit mit Hundertstelsekunden ermitteln

Wert 2 / 0x02

Verwandte

Siehe Befehle,

SysTime()

Option bei SysTime() durch die die Systemzeit mit Hundertstelsekunden ermittelt werden kann.



TimeSec

Zeit mit Sekunden ermitteln

Wert 1 / 0x01

Verwandte

Siehe Befehle,

SysTime()

Option bei SysTime() durch die die Systemzeit mit Sekunden ermittelt werden kann.

\_TimeServer

Systemzeit des Servers ermitteln

Wert 4 / 0x04

Verwandte

Siehe Befehle,

SysTime()

Option bei SysTime() durch die die Systemzeit des Servers ermittelt werden kann.

Befehle für Systemobjekte

Befehle für Systemobjekte

Siehe [Befehlsgruppen](#),  
[Befehlsliste](#)

## **Befehle**

- [HttpClose](#)
- [HttpGetData](#)
- [HttpOpen](#)
- [LocaleLoad](#)
- [LocaleSelect](#)
- [LocaleUnload](#)
- [SysPropGet](#)
- [SysPropSet](#)

## **Konstanten**

- [\\_LclLangCzech](#)
- [\\_LclLangEnglish](#)
- [\\_LclLangFrench](#)
- [\\_LclLangGerman](#)
- [\\_LclLangHungarian](#)
- [\\_LclLangItalian](#)
- [\\_LclLangNeutral](#)
- [\\_LclLangPolish](#)
- [\\_LclLangSlovak](#)
- [\\_LclLangTurkish](#)
- [\\_LclSubLangDefault](#)
- [\\_LclSubLangEnglishAU](#)
- [\\_LclSubLangEnglishCA](#)
- [\\_LclSubLangEnglishIE](#)
- [\\_LclSubLangEnglishNZ](#)
- [\\_LclSubLangEnglishUK](#)
- [\\_LclSubLangEnglishUS](#)
- [\\_LclSubLangEnglishZA](#)
- [\\_LclSubLangFrench](#)
- [\\_LclSubLangFrenchBE](#)
- [\\_LclSubLangFrenchCA](#)
- [\\_LclSubLangFrenchCH](#)
- [\\_LclSubLangFrenchLU](#)
- [\\_LclSubLangFrenchMC](#)
- [\\_LclSubLangGerman](#)
- [\\_LclSubLangGermanAT](#)
- [\\_LclSubLangGermanCH](#)
- [\\_LclSubLangGermanLI](#)
- [\\_LclSubLangGermanLU](#)
- [\\_LclSubLangItalian](#)
- [\\_LclSubLangItalianCH](#)
- [\\_LclSubLangNeutral](#)

```
obj -> SysPropGet(int1,  
var2[,int3]) : logic
```



Systemobjekteigenschaft ermitteln

obj      Objekt

int1      Eigenschaftskonstante

var2      Eigenschaftswert

int3      Position

Resultat logic Ermittlungserfolg

Verwandte Befehle,

Siehe SysPropSet(), Liste der

Systemobjekt-Eigenschaften

Dieser Befehl liest eine Eigenschaft eines Systemobjektes aus.

Als erster Parameter muss die Konstante der Eigenschaft übergeben werden. Die Konstanten setzen sich aus \_SysProp und dem Namen der Eigenschaft zusammen.

Im zweiten Parameter wird die Variable übergeben, in die der Wert der Eigenschaft kopiert werden soll.

## Beispiel:

```
local{ tHdlLocale      : handle; tDateFormat      : alpha;}...// Auslesen des Datumsformates tHdlLocale
```

Das Kommando kann ebenfalls dazu verwendet werden, um zu ermitteln, ob ein bestimmtes Objekt eine Eigenschaft besitzt. Ist eine Eigenschaft nicht vorhanden, liefert der Befehl den Wert false zurück.



Alternativ kann die Eigenschaft auch wie folgt ausgelesen werden:

## Beispiel:

```
local{ tHdlLocale      : handle; tDateFormat      : alpha;}...// Auslesen des Datumsformates tHdlLocale
```

Der optionale Parameter (int3) muss nur angegeben werden, wenn mehrere Werte einer Eigenschaft zugeordnet werden können. Entsprechende Hinweise befinden sich in den Beschreibungen der Eigenschaften.

## Beispiel:

```
// Name des Sonntags ermittelntHdlLocale->SysPropGet(_SysPropLclDateDayN, aSunday, 7);// Alternat
```

obj -> SysPropSet(int1, var2[,  
int3]) : logic



Systemobjekteigenschaft setzen

obj      Objekt

int1      Eigenschaftskonstante

var2      Eigenschaftswert

int3      Position

Resultat logic    Setzungserfolg

Verwandte Befehle,

Siehe SysPropGet(), Liste der

Systemobjekt-Eigenschaften

Dieser Befehl setzt eine Eigenschaft eines Systemobjektes.

Als ersten Parameter muss die Konstante der Eigenschaft übergeben werden. Die Konstanten setzen sich aus \_SysProp und dem Namen der Eigenschaft zusammen.

Im zweiten Parameter wird der zu setzende Wert übergeben.

### Beispiel:

```
// Setzen des kurzen DatumsformatesHdlLocale # LocaleLoad(_LclLangGerman, _LclSubLangGerman);tHc
```



Alternativ kann die Eigenschaft auch wie folgt gesetzt werden:


### Beispiel:

```
// Setzen des kurzen DatumsformatesHdlLocale # LocaleLoad(_LclLangGerman, _LclSubLangGerman
```

Der optionale Parameter (int3) muss nur angegeben werden, wenn mehrere Werte einer Eigenschaft zugeordnet werden können. Entsprechende Hinweise befinden sich in den Beschreibungen der Eigenschaften.

### Beispiel:

```
// Name des Sonntags setzentHdlLocale->SysPropSet(_SysPropLclDateDayN, 'Sonntag', 7);// Alternati
```

LocaleLoad(int1, int2) : 

handle

Ländereinstellungen laden

int1 Regionale ID

int2 Regionale Sub-ID

Resultat handle Locale-Objekt-Deskriptor

Siehe Verwandte Befehle, Locale,  
LocaleSelect(), LocaleUnload()

Dieser Befehl lädt die länderspezifischen Einstellungen des in (int1) und (int2) angegebenen Landes. Die Einstellungen können über ein Locale-Objekt abgefragt oder verändert werden. Der Befehl gibt den Deskriptor zu dem entsprechenden Locale-Objekt als Rückgabewert zurück.

Folgende Werte können in (int1) und (int2) übergeben werden:

- LclLangNeutral Länderunabhängige Einstellungen
  - ♦ LclSubLangNeutral Länderunabhängige Einstellungen
- LclLangCzech Tschechisch
- LclLangEnglish Englisch
  - ♦ LclSubLangEnglishUS USA
  - ♦ LclSubLangEnglishUK Großbritannien
  - ♦ LclSubLangEnglishAU Australien
  - ♦ LclSubLangEnglishCA Kanada
  - ♦ LclSubLangEnglishNZ Neuseeland
  - ♦ LclSubLangEnglishIE Irland
  - ♦ LclSubLangEnglishZA Südafrika
- LclLangFrench Französisch
  - ♦ LclSubLangFrench Frankreich
  - ♦ LclSubLangFrenchBE Belgien
  - ♦ LclSubLangFrenchCA Kanada
  - ♦ LclSubLangFrenchCH Schweiz
  - ♦ LclSubLangFrenchLU Luxemburg
  - ♦ LclSubLangFrenchMC Monaco
- LclLangGerman Deutsch
  - ♦ LclSubLangGerman Deutschland
  - ♦ LclSubLangGermanCH Schweiz
  - ♦ LclSubLangGermanAT Österreich
  - ♦ LclSubLangGermanLU Luxemburg
  - ♦ LclSubLangGermanLI Liechtenstein
- LclLangHungarian Ungarisch
- LclLangItalian Italienisch
  - ♦ LclSubLangItalian Italien
  - ♦ LclSubLangItalianCH Schweiz

- \_LclLangPolish Polnisch
- \_LclLangSlovak Slovakisch
- \_LclLangTurkish Türkisch

Wird in (int1) \_LclLangNeutral und in (int2) \_LclSubLangNeutral angegeben, wird eine länderunabhängige Einstellung geladen.

Mit der Kombination \_LclLangNeutral und \_LclSubLangDefault kann die Einstellung des Windows-Benutzers geladen werden.

Durch die Übergabe falscher Kombinationen wird ein leeres Locale-Objekt geladen.

Sind verschiedene Ländereinstellungen geladen, kann mit dem Befehl LocaleSelect() zwischen diesen Einstellungen gewechselt werden. Der Befehl LocaleUnload() entfernt das Locale-Objekt wieder aus dem Speicher.

obj -> LocaleSelect()



Ländereinstellungen wechseln

obj Locale-Objekt-Deskriptor

Siehe Verwandte Befehle,  
Locale, LocaleLoad()

Mit diesem Befehl kann zwischen mehreren geladenen Ländereinstellungen gewechselt werden. Die Ländereinstellungen müssen zuvor mit dem Befehl LocaleLoad() geladen worden sein. In (obj) wird der Deskriptor des Locale-Objektes angegeben, das die neuen Ländereinstellungen enthält.

Dies verändert die Einstellungen für alle Dialoge und A+ Prozeduren. Sollen die entsprechenden Einstellungen nicht global verwendet werden, können sie in den Konvertierungsbefehlen oder beim Oberflächenobjekt angegeben werden. Ein LocaleSelect() entfällt in diesem Fall.

Sollen die Systemeinstellungen wieder hergestellt werden, muss das zur Zeit selektierte Locale mit dem Befehl LocaleUnload() entladen werden.



obj -> LocaleUnload()



Ländereinstellungen entladen

obj Locale-Objekt-Deskriptor

Siehe Verwandte Befehle,  
Locale, LocaleLoad()

Mit diesem Befehl wird das in (obj) übergebene Locale-Objekt aus dem Speicher entfernt.

Konstanten für Systemobjekte  
Konstanten für Systemobjekte  
Siehe Befehle für  
Systemobjekte

- LclLangCzech
- LclLangEnglish
- LclLangFrench
- LclLangGerman
- LclLangHungarian
- LclLangItalian
- LclLangNeutral
- LclLangPolish
- LclLangSlovak
- LclLangTurkish
- LclSubLangDefault
- LclSubLangEnglishAU
- LclSubLangEnglishCA
- LclSubLangEnglishIE
- LclSubLangEnglishNZ
- LclSubLangEnglishUK
- LclSubLangEnglishUS
- LclSubLangEnglishZA
- LclSubLangFrench
- LclSubLangFrenchBE
- LclSubLangFrenchCA
- LclSubLangFrenchCH
- LclSubLangFrenchLU
- LclSubLangFrenchMC
- LclSubLangGerman
- LclSubLangGermanAT
- LclSubLangGermanCH
- LclSubLangGermanLI
- LclSubLangGermanLU
- LclSubLangItalian
- LclSubLangItalianCH
- LclSubLangNeutral

\_LclLangCzech  
Ländereinstellungen Tschechisch  
Wert 5 / 0x05

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen von Tschechien geladen werden können. Als regionale Sub-ID muss \_LclSubLangDefault angegeben werden.

\_LclLangEnglish  
Ländereinstellungen Englisch  
Wert 9 / 0x09

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen von englischsprachigen Ländern geladen werden können. Als regionale Sub-ID können folgende Konstanten angegeben werden:

<u>_LclSubLangDefault</u>	Entspricht der Option <u>_LclSubLangEnglishUS</u>
<u>_LclSubLangEnglishUS</u>	Einstellungen für USA
<u>_LclSubLangEnglishUK</u>	Einstellungen für Großbritannien
<u>_LclSubLangEnglishAU</u>	Einstellungen für Australien
<u>_LclSubLangEnglishCA</u>	Einstellungen für Kanada
<u>_LclSubLangEnglishNZ</u>	Einstellungen für Neuseeland
<u>_LclSubLangEnglishIE</u>	Einstellungen für Irland
<u>_LclSubLangEnglishZA</u>	Einstellungen für Südafrika

\_LclLangFrench  
Ländereinstellungen Französisch  
Wert 12 / 0x0C

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen von französischsprachigen Ländern geladen werden können. Als regionale Sub-ID können folgende Konstanten angegeben werden:

<u>_LclSubLangDefault</u>	Entspricht der Option <u>_LclSubLangFrench</u>
<u>_LclSubLangFrench</u>	Einstellungen für Frankreich
<u>_LclSubLangFrenchBE</u>	Einstellungen für Belgien
<u>_LclSubLangFrenchCA</u>	Einstellungen für Kanada
<u>_LclSubLangFrenchCH</u>	Einstellungen für Schweiz
<u>_LclSubLangFrenchLU</u>	Einstellungen für Luxemburg
<u>_LclSubLangFrenchMC</u>	Einstellungen für Monaco

\_LclLangGerman  
Ländereinstellungen Deutsch  
Wert 7 / 0x07

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen von deutschsprachigen Ländern geladen werden können. Als regionale Sub-ID können folgende Konstanten angegeben werden:

<u>_LclSubLangDefault</u>	Entspricht der Option <u>_LclSubLangGerman</u>
<u>_LclSubLangGerman</u>	Einstellungen für Deutschland
<u>_LclSubLangGermanCH</u>	Einstellungen für Schweiz
<u>_LclSubLangGermanAT</u>	Einstellungen für Österreich
<u>_LclSubLangGermanLU</u>	Einstellungen für Luxemburg
<u>_LclSubLangGermanLI</u>	Einstellungen für Liechtenstein

\_LclLangHungarian  
Ländereinstellungen Ungarn  
Wert 14 / 0x0E

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen von Ungarn geladen werden können. Als regionale Sub-ID muss \_LclSubLangDefault angegeben werden.

\_LclLangItalian  
Ländereinstellungen Italienisch  
Wert 16 / 0x10

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen von italienischsprachigen Ländern geladen werden können. Als regionale Sub-ID können folgende Konstanten angegeben werden:

<u>_LclSubLangDefault</u>	Entspricht der Option <u>_LclSubLangItalian</u>
<u>_LclSubLangItalian</u>	Einstellungen für Italien
<u>_LclSubLangItalianCH</u>	Einstellungen für Schweiz



\_LclLangNeutral  
Länderunabhängige Einstellungen  
Wert 0 / 0x00

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die eine länderunabhängige Einstellung geladen werden kann, wenn als regionale Sub-ID \_LclSubLangNeutral angegeben wird.

Mit der Angabe von \_LclSubLangDefault als regionale Sub-ID wird die Einstellung des Windows-Benutzers geladen.

\_LclLangPolish  
Ländereinstellungen Polen  
Wert 21 / 0x15

Verwandte

Siehe Befehle,  
LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen von Polen geladen werden können. Als regionale Sub-ID muss \_LclSubLangDefault angegeben werden.

\_LclLangSlovak  
Ländereinstellungen Slowakei  
Wert 27 / 0x1B

Verwandte

Siehe Befehle,  
LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen der Slowakei geladen werden können. Als regionale Sub-ID muss \_LclSubLangDefault angegeben werden.

\_LclLangTurkish  
Ländereinstellungen Türkei  
Wert 31 / 0x1F

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die die länderspezifischen Einstellungen der Türkei geladen werden können. Als regionale Sub-ID muss \_LclSubLangDefault angegeben werden.

\_LclSubLangDefault  
Regionaleinstellungen Standard  
Wert 1 / 0x01

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die eine Standard-Regionaleinstellung geladen werden kann. Wurde als regionale ID \_LclLangNeutral angegeben, werden die länderspezifischen Einstellungen des Benutzers geladen.

\_LclSubLangEnglishAU

Ländereinstellungen Australien (Englisch)

Wert 3 / 0x03

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangEnglish

Option bei LocaleLoad() durch die die Ländereinstellungen von Australien (Englisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangEnglish angegeben wurde.

\_LclSubLangEnglishCA  
Ländereinstellungen Kanada (Englisch)  
Wert 4 / 0x04

Verwandte  
Siehe Befehle,  
LocaleLoad(),  
\_LclLangEnglish

Option bei LocaleLoad() durch die die Ländereinstellungen von Kanada (Englisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangEnglish angegeben wurde.

\_LclSubLangEnglishIE  
Ländereinstellungen Irland (Englisch)  
Wert 6 / 0x06

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangEnglish

Option bei LocaleLoad() durch die die Ländereinstellungen von Irland (Englisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangEnglish angegeben wurde.



\_LclSubLangEnglishNZ

Ländereinstellungen Neuseeland (Englisch)

Wert 5 / 0x05

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangEnglish

Option bei LocaleLoad() durch die die Ländereinstellungen von Neuseeland (Englisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangEnglish angegeben wurde.

\_LclSubLangEnglishUK  
Ländereinstellungen Großbritannien (Englisch)  
Wert 2 / 0x02

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangEnglish

Option bei LocaleLoad() durch die die Ländereinstellungen von Großbritannien (Englisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangEnglish angegeben wurde.

\_LclSubLangEnglishUS  
Ländereinstellungen USA (Englisch)  
Wert 1 / 0x01

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangEnglish

Option bei LocaleLoad() durch die die Ländereinstellungen der USA (Englisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangEnglish angegeben wurde.

\_LclSubLangEnglishZA

Ländereinstellungen Südafrika (Englisch)

Wert 7 / 0x07

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangEnglish

Option bei LocaleLoad() durch die die Ländereinstellungen von Südafrika (Englisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangEnglish angegeben wurde.

\_LclSubLangFrench  
Ländereinstellungen Frankreich (Französisch)  
Wert 1 / 0x01

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangFrench

Option bei LocaleLoad() durch die die Ländereinstellungen von Frankreich (Französisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangFrench angegeben wurde.

\_LclSubLangFrenchBE

Ländereinstellungen Belgien (Französisch)

Wert 2 / 0x02

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangFrench

Option bei LocaleLoad() durch die die Ländereinstellungen von Belgien (Französisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangFrench angegeben wurde.

\_LclSubLangFrenchCA

Ländereinstellungen Kanada (Französisch)

Wert 3 / 0x03

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangFrench

Option bei LocaleLoad() durch die die Ländereinstellungen von Kanada (Französisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangFrench angegeben wurde.

\_LclSubLangFrenchCH  
Ländereinstellungen Schweiz (Französisch)  
Wert 4 / 0x04

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangFrench

Option bei LocaleLoad() durch die die Ländereinstellungen der Schweiz (Französisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangFrench angegeben wurde.



\_LclSubLangFrenchLU

Ländereinstellungen Luxemburg (Französisch)

Wert 5 / 0x05

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangFrench

Option bei LocaleLoad() durch die die Ländereinstellungen von Luxemburg (Französisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangFrench angegeben wurde.

\_LclSubLangFrenchMC  
Ländereinstellungen Monaco (Französisch)  
Wert 6 / 0x06

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangFrench

Option bei LocaleLoad() durch die die Ländereinstellungen von Monaco (Französisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangFrench angegeben wurde.

\_LclSubLangGerman  
Ländereinstellungen Deutschland (Deutsch)  
Wert 1 / 0x01

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangGerman

Option bei LocaleLoad() durch die die Ländereinstellungen von Deutschland (Deutsch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangGerman angegeben wurde.

\_LclSubLangGermanAT

Ländereinstellungen Österreich (Deutsch)

Wert 3 / 0x03

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangGerman

Option bei LocaleLoad() durch die die Ländereinstellungen von Österreich (Deutsch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangGerman angegeben wurde.

\_LclSubLangGermanCH  
Ländereinstellungen Schweiz (Deutsch)  
Wert 2 / 0x02

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangGerman

Option bei LocaleLoad() durch die die Ländereinstellungen der Schweiz (Deutsch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangGerman angegeben wurde.

\_LclSubLangGermanLI  
Ländereinstellungen Liechtenstein (Deutsch)  
Wert 5 / 0x05

Verwandte  
Siehe Befehle,  
LocaleLoad(),  
\_LclLangGerman

Option bei LocaleLoad() durch die die Ländereinstellungen von Liechtenstein (Deutsch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangGerman angegeben wurde.

\_LclSubLangGermanLU

Ländereinstellungen Luxemburg (Deutsch)

Wert 4 / 0x04

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangGerman

Option bei LocaleLoad() durch die die Ländereinstellungen von Luxemburg (Deutsch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangGerman angegeben wurde.

\_LclSubLangItalian  
Ländereinstellungen Italien (Italienisch)  
Wert 1 / 0x01

Verwandte

Siehe Befehle,  
LocaleLoad(),  
\_LclLangItalian

Option bei LocaleLoad() durch die die Ländereinstellungen von Italien (Italienisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangItalian angegeben wurde.



\_LclSubLangItalianCH

Ländereinstellungen Schweiz (Italienisch)

Wert 2 / 0x02

Verwandte

Siehe Befehle,

LocaleLoad(),

\_LclLangItalian

Option bei LocaleLoad() durch die die Ländereinstellungen der Schweiz (Italienisch) geladen werden können. Diese Option kann nur angegeben werden, wenn als regionale ID \_LclLangItalian angegeben wurde.

\_LclSubLangNeutral  
Länderunabhängige Einstellungen  
Wert 0 / 0x00

Verwandte

Siehe Befehle,

LocaleLoad()

Option bei LocaleLoad() durch die eine länderunabhängige Einstellung geladen werden kann. Spezielle Ländereinstellungen werden nicht geladen. Diese regionale Sub-ID kann nur in Verbindung mit \_LclLangNeutral angegeben werden.

Befehle für zentrale Datenobjekte

Liste der Befehle und Konstanten zur Bearbeitung von zentrale Datenobjekten

Befehlsgruppen,


Siehe Befehlsliste,

Zentrale

Datenobjekte

## **Befehle**

- RmtDataRead
- RmtDataSearch
- RmtDataWrite

RmtDataRead(alpha1, int2, )

var alpha3) : int

Zentrales Datenobjekt lesen

alpha1 Schlüsselwert

Optionen

0 Sperre nicht  
verändern

int2 \_RecLock Datenobjekt sperren

\_RecUnlock Datenobjekt  
entsperren

alpha3 Unter dem Schlüsselwert  
abgelegter Wert

Fehlerwert

\_rOk Datenobjekt  
gelesen

Resultat int \_rLocked Datenobjekt   
gesperrt

\_rNoRec Datenobjekt nicht  
vorhanden

Verwandte Befehle,

Siehe RmtDataWrite(),  
RmtDataSearch()

Mit dieser Anweisung wird ein zentrales Datenobjekt ausgelesen. In (alpha1) wird der Schlüsselwert übergeben, der schon beim Schreiben des Objekts mit RmtDataWrite() übergeben wurde. Mit der Option (int2) kann angegeben werden, ob das Datenobjekt für andere Benutzer gesperrt werden soll. Eine Sperre verhindert das Überschreiben des Objekts. Bei der Übergabe von 0 wird die Sperre nicht verändert

In (alpha3) muss eine ausreichend lange alpha-Variable übergeben werden, um den gespeicherten Wert aufzunehmen. Ist die Variable zu kurz definiert, wird die Zeichenkette abgeschnitten.

Konnte das zentrale Datenobjekt gelesen werden, wird \_rOk zurückgegeben. Folgende Fehlerwerte können zurückgegeben werden:

\_rLocked Das zentrale Datenobjekt konnte gelesen aber nicht gesperrt werden.

\_rNoRec Das Datenobjekt zu dem angegebenen Schlüsselwert wurde nicht gefunden.

### Beispiel:

```
local{ tRmtDataKey : alpha(250); tRmtDataValue : alpha(4096); tErg : int;}{ ... t
```

Mögliche Laufzeitfehler:

\_ErrStringOverflow Die in (alpha1) übergebene Zeichenkette hat mehr als 2000 Zeichen.



RmtDataSearch(alpha1, int2) : int

Schlüsselwerte der zentralen Datenobjekte durchsuchen

alpha1 Name des Schlüssels

Leseoptionen

0 angegebenes  
Datenobjekt  
lesen

\_RecFirst erstes  
Datenobjekt  
lesen

int2 \_RecPrev vorheriges  
Datenobjekt  
lesen

\_RecNext nächstes  
Datenobjekt  
lesen

\_RecLast letztes  
Datenobjekt  
lesen

Resultat alpha Schlüsselwert

Siehe Verwandte Befehle,  
Zentrale Datenobjekte

Mit dieser Anweisung können zentrale Datenobjekte gesucht werden. Dazu wird ein Schlüsselbegriff in (alpha1) übergeben. Die Optionen entscheiden darüber, welcher Schlüsselbegriff zurückgegeben wird.

### **(int2) Aktion**

0 Der in (alpha1) angegebene Schlüsselwert wird gelesen. Ist dieser Wert nicht vorhanden, wird der nächst größere Wert zurückgegeben.

\_RecFirst Der erste Schlüsselwert wird zurückgegeben. Der Wert in (alpha1) wird ignoriert.

\_RecPrev Ausgehend vom in (alpha1) übergebenen Wert wird der vorhergehende Wert zurückgegeben.

\_RecNext Ausgehend vom in (alpha1) übergebenen Wert wird der nächste Wert zurückgegeben.


\_RecLast Der letzte Schlüsselwert wird zurückgegeben. Der Wert in (alpha1) wird ignoriert.

### **Beispiel:**

```
// Alle Schlüsselwerte lesenfor aRmtDataKey # RmtDataSearch('', _RecFirst);loop aRmtDataKey #
```

Mögliche Laufzeitfehler:

\_ErrStringOverflow Der in (alpha1) angegebene Schlüsselwert ist länger als 2000 Zeichen.

RmtDataWrite(alpha1, int2, )  
alpha3) : int

Zentrales Datenobjekt erzeugen

alpha1 Schlüsselwert

Optionen

0 Sperre nicht ändern

RecLock Datenobjekt sperren


int2 RecUnlock Datenobjekt  
entsperren

RmtDataTemp temporäres  
Datenobjekt

alpha3 Daten

Fehlerwert

rOk Datenobjekt  
angelegt

Resultat int rLocked Datenobjekt   
gesperrt

ErrLimitExceeded Speicher  
erschöpft

Siehe Verwandte Befehle, RmtDataRead(),  
RmtDataSearch()

Mit dieser Anweisung wird ein zentrales Datenobjekt angelegt oder überschrieben. Das zentrale Datenobjekt kann von jedem Client, der sich an der gleichen Datenbank angemeldet hat, abgerufen werden. Das Datenobjekt bleibt erhalten bis es mit einem Leerstring überschrieben wurde, sich der anlegende Benutzer von der Datenbank abgemeldet hat (bei Verwendung von RmtDataTemp) oder die Datenbank geschlossen wurde.

In (alpha1) wird ein Schlüsselbegriff angegeben. Der Schlüsselbegriff kann bis zu 2000 Zeichen lang und muss eindeutig sein. In den Optionen (int2) kann angegeben werden, ob das Datenobjekt nach dem Anlegen gleich gesperrt sein soll (RecLock) oder nicht (RecUnlock). Wird hier 0 Übergeben, wird die Sperre nicht verändert. Diese Option kann mit RmtDataTemp kombiniert werden, wenn das Objekt maximal bis zur Abmeldung des Benutzers erhalten bleiben soll.

Die Gesamtlänge von alpha1 und alpha3 (Schlüssel- und Datenanteil) darf 4601 Byte nicht überschreiten.

Konnte das zentrale Datenobjekt geschrieben werden, wird rOk zurückgegeben. Folgende Fehlerwerte können zurückgegeben werden:

rLocked Das zentrale Datenobjekt ist von einem anderen Benutzer gesperrt.

ErrLimitExceeded Das Datenobjekt konnte nicht angelegt oder beschrieben werden, weil die maximale Speichergrenze (64 MB) erreicht ist.

ErrStringOverflow Der Schlüssel überschreitet eine Länge von 2000 Byte, die Daten eine Länge von 4096 Byte oder die Gesamtlänge überschreitet 4601 Byte.

**Beispiele:**

```
// Daten für einen Job-Server schreibenRmtDataWrite('JobNr1', _RecUnlock, 'Job Daten');// Eintrag
```

Netzwerkinformationsbefehle

Befehle zum Ermitteln von Netzwerkinformationen

Siehe Befehlsgruppen,  
Befehlsliste


## **Befehle**

- NetInfo

## **Konstanten**

- \_NtiAddress
- \_NtiAddressServer
- \_NtiAddressTSC
- \_NtiConnectTime
- \_NtiName
- \_NtiNameIP
- \_NtiNameTSC
- \_NtiPing
- \_NtiPingTimeOut
- \_NtiProtocol
- \_NtiReadVol
- \_NtiRequests
- \_NtiWriteVol



NetInfo(int1[, alpha2]) : alpha   
 Netzwerkinformationen ermitteln  
 int1 Optionen (siehe Text)  
     Erweiterte Optionen (optional)  
     Rechnername oder IP-Adresse (bei  
 alpha2 Verwendung von NtiPing oder  
     NtiAddress)  
     Maximale Wartezeit (bei Verwendung  
     von NtiPingTimeOut)  
     Ermittlungsergebnis  
 Resultat alpha Netzwerkinformation oder  
     Fehlerwert bei Verwendung von  
     NtiPing oder NtiWakeOnLan

Mit NetInfo() können diverse Netzwerkinformationen abgefragt werden. Dem Befehl wird die Informationsart, die in Erfahrung gebracht werden will, mitgegeben wie z. B. Rechnername, Netzwerkprotokoll, IP-Nummer des Rechners und Netzwerkadresse.

Folgende Optionen (int1) können verwendet werden:

- NtiName  
 Rechnername ermitteln
- NtiNameTSC  
 Rechnername (Terminal-Client) ermitteln
- NtiNameIP  
 Host-Name ermitteln
- NtiProtocol  
 Protokoll ermitteln
- NtiAddress  
 Client-Netzwerkadresse (ohne alpha2) oder Adresse eines anderen Rechners (mit alpha2) ermitteln
- NtiAddressTSC  
 Client-Netzwerkadresse (Terminal-Client) ermitteln
- NtiAddressServer  
 Server-Netzwerkadresse ermitteln
- NtiConnectTime  
 Verbindungszeit ermitteln
- NtiRequests  
 Client-Anfrageanzahl ermitteln
- NtiReadVol  
 Empfangenes Datenvolumen ermitteln

## Kontakt

- NtiWriteVol

Gesendetes Datenvolumen ermitteln

- NtiPing

Rechner-Antwortzeit ermitteln

- NtiPingTimeOut

Maximale Wartezeit setzen

- NtiWakeOnLan

Wake-on-LAN "Magic Packet" versenden

Resultate, die numerische Informationen enthalten müssen gegebenenfalls mit CnvIA() umgewandelt werden.

Konstanten für Netzwerkinformationsbefehle

Konstanten für Netzwerkinformationsbefehle

Siehe Netzwerkinformationsbefehle

- NtiAddress
- NtiAddressServer
- NtiAddressTSC
- NtiConnectTime
- NtiName
- NtiNameIP
- NtiNameTSC
- NtiPing
- NtiPingTimeOut
- NtiProtocol
- NtiReadVol
- NtiRequests
- NtiWriteVol

\_NtiAddress

Client-Netzwerkadresse ermitteln

Wert 4

Verwandte

Siehe Befehle,

NetInfo()

Option bei NetInfo() durch die die Netzwerkadresse (IP-Adresse) des Clients ermittelt werden kann.

Wird bei NetInfo() der Parameter (alpha2) angegeben, wird versucht die IP-Adresse des angegebenen Rechners zu ermitteln. Kann keine IP-Adresse ermittelt werden, weil der Zielhost nicht erreicht werden kann, wird eine leere Zeichenkette zurückgegeben. Durch die Verwendung eines Präfixes vor dem Namen kann die Auswahl der IP-Adresse und somit des Protokolls gesteuert werden. Folgende Präfixe können angegeben werden:

ip4: ausschließlich IPv4

ip4f: bevorzugt IPv4, IPv6 wenn es keine IPv4-Adresse gibt

ip6: ausschließlich IPv6

ip6f: bevorzugt IPv6, IPv4 wenn es keine IPv6-Adresse gibt

Wird kein Präfix angegeben, wird automatisch ip4f: verwendet.

Das verwendete Protokoll kann mit der Option \_NtiProtocol ermittelt werden.

### **Beispiel:**

```
// Arbeitsplatzabhängig Drucker setzenswitch (NetInfo(_NtiAddress)){ case '10.0.1.12' : aPrinter
```

Alternativ kann auch der Name des Client-Rechners mit der Option \_NtiName ermittelt werden.

## Kontakt

\_NtiAddressServer  
Server-Netzwerkadresse ermitteln  
Wert 5

### Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die die Netzwerkadresse (IP-Adresse) des Server ermittelt werden kann.

Das verwendete Protokoll kann mit der Option \_NtiProtocol ermittelt werden.

### **Beispiel:**

```
// Server-abhängig temporären Pfad setzen  
switch (NetInfo(_NtiAddressServer)){ case '10.0.0.1' :
```

\_NtiAddressTSC

Client-Netzwerkadresse (Terminal-Client) ermitteln

Wert 13

Verwandte

Siehe Befehle,

NetInfo()

Option bei NetInfo() durch die die Netzwerkadresse des Terminal-Clients ermittelt werden kann, falls der CONZEPT 16-Client in einer Terminal-Sitzung läuft.

Bei Verwendung der Option \_NtiAddress wird nur die Adresse des Terminal-Servers ermittelt.

Zusätzlich kann auch der Name des Client-Rechners mit der Option \_NtiNameTSC ermittelt werden.

Der Befehl ist besonders in einer Terminal-Server-Umgebung nützlich, wenn die Applikation mit dem externen Debugger untersucht werden soll.

Der Debugger kann dazu auf der lokalen Maschine (nicht in der Terminal-Session) gestartet werden. Zum Verbinden mit der lokalen Maschine wird folgende Programmzeile ausgeführt:

```
DbgConnect(NetInfo(_NtiAddressTSC), false, false);
```

\_NtiConnectTime  
Verbindungszeit ermitteln  
Wert 6

Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die die Verbindungszeit zwischen CONZEPT 16-Server und -Client in Sekunden ermittelt werden kann.

Um Rechenoperationen mit dem Rückgabewert durchführen zu können, muss dieser erst mit dem Befehl CnvIA() in einen ganzzahligen Wert gewandelt werden.

\_NtiName  
Rechnername ermitteln  
Wert 1

### Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die der Rechnername ermittelt werden kann. Unter Windows ist dies der NetBIOS-Name, bei Netware der Server-Name und bei Unix der Rechnername (uname).

Zusammen mit dem Namen kann mit der Option \_\_NtiNameIP noch der Host-Name ermittelt werden.

### **Beispiel:**

```
// Arbeitsplatzabhängig Drucker setzenswitch (NetInfo(_NtiName)){ case 'OFFICE1' : aPrinterName
```

Alternativ kann auch die IP-Adresse des Client-Rechners mit der Option \_\_NtiAddress ermittelt werden.



\_NtiNameIP  
Host-Name ermitteln  
Wert 2

Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die der Host-Name des Rechners ermittelt werden kann. Der Host-Name wird entweder über die Hosts-Tabelle oder einen DNS ermittelt. Je nach Resolver ist die Domain-Information im Namen enthalten.

**Beispiel:**

```
NetInfo(_NtiNameIP); // -> 'support1.vectorsoft.de'
```

Soll nur der Name des Rechners oder seine IP-Adresse ermittelt werden, muss die Option \_NtiName bzw. \_NtiAddress verwendet werden.

\_NtiNameTSC

Rechnername (Terminal-Client) ermitteln

Wert 12

Verwandte

Siehe Befehle,

NetInfo()

Option bei NetInfo() durch die der Rechnername des Terminals ermittelt werden kann, falls der CONZEPT 16-Client in einer Terminal-Sitzung läuft.

Bei der Verwendung der Option \_NtiName würde in diesem Fall nur der Name des Terminal-Servers ermittelt werden. Läuft der CONZEPT 16-Client nicht in einer Terminal-Sitzung, wird mit \_NtiNameTSC und \_NtiName die gleiche Information ermittelt.

Zusätzlich kann auch die IP-Adresse des Client-Rechners mit der Option \_NtiAddressTSC ermittelt werden.

\_NtiPing

Rechner-Antwortzeit ermitteln

Wert 10

Verwandte

Siehe Befehle,

NetInfo()

Option bei NetInfo() durch die die Antwortzeit eines im Netzwerk befindlichen Rechners ermittelt werden kann.

Damit kann überprüft werden ob ein bestimmter Rechner im Netz erreichbar ist.

Das Resultat ist 0, wenn ein Timeout aufgetreten ist, bei positiven Werten entspricht dies der Antwortverzögerung in Millisekunden, negative Werte sind Fehlerwerte.

Standardmäßig wird nach drei Sekunden ein Timeout festgestellt. Die maximale Wartezeit kann mit der Option \_NtiPingTimeOut verändert werden.

Folgende Fehlerwerte können zurückgegeben werden:

<u>_ErrNetNoHost</u>	Host-Adresse konnte nicht ermittelt werden
<u>_ErrNetCreate</u>	ICMP-Sockets nicht verfügbar
<u>_ErrNetSelect</u>	Fehler bei Socket-Abfrage
<u>_ErrNetRead</u>	Fehler beim Lesen der Antwort
<u>_ErrNetReadLess</u>	Antwortpaket ungültig
<u>_ErrNetIcmpType</u>	Antwortpaket vom falschen Typ
<u>_ErrNetIcmpID</u>	Antwortpaket mit falscher ID
<u>_ErrNetWrite</u>	Anfrage konnte nicht versendet werden

\_NtiPingTimeOut  
Maximale Wartezeit setzen  
Wert 11

Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die die maximale Wartezeit auf eine Antwort eines Rechners in Millisekunden gesetzt werden kann.

Standardmäßig ist die maximale Wartezeit bei einer Anfrage mit der Option \_NtiPing auf 3 Sekunden eingestellt. Sie kann auf einen Wert zwischen 100 Millisekunden und 30 Sekunden verändert werden.

**Beispiel:**

```
// Server-Adresse ermitteltAdresse # NetInfo(_NtiAddressServer); // Maximale Wartezeit auf 100ms
```

## Kontakt

\_NtiProtocol  
Protokoll ermitteln  
Wert 3

### Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die das zur Kommunikation mit dem Server verwendete Protokoll ermittelt werden kann.

Es wird nur die Kommunikation über TCP/IP unterstützt. Es wird immer 'TCP' zurückgegeben.

\_NtiReadVol  
Empfangenes Datenvolumen ermitteln  
Wert 9

Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die die Größe des empfangenen Datenvolumens vom CONZEPT 16-Server seit dem Verbindungsaufbau in KB ermittelt werden kann.

Um Rechenoperationen mit dem Rückgabewert durchführen zu können, muss dieser erst mit dem Befehl CnvIA() in einen ganzzahligen Wert gewandelt werden.

\_NtiRequests  
Client-Anfrageanzahl ermitteln  
Wert 7

Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die die Anzahl der Anfragen des CONZEPT 16-Clients an den Server seit dem Verbindungsaufbau ermittelt werden kann.

Um Rechenoperationen mit dem Rückgabewert durchführen zu können, muss dieser erst mit dem Befehl CnvIA() in einen ganzzahligen Wert gewandelt werden.

\_NtiWriteVol  
Gesendetes Datenvolumen ermitteln  
Wert 8

Verwandte

Siehe Befehle,  
NetInfo()

Option bei NetInfo() durch die die Größe des gesendeten Datenvolumens an den CONZEPT 16-Server seit dem Verbindungsaufbau in KB ermittelt werden kann.

Um Rechenoperationen mit dem Rückgabewert durchführen zu können, muss dieser erst mit dem Befehl CnvIA() in einen ganzzahligen Wert gewandelt werden.



Funktionen der Benutzeroberfläche

Funktionen der Benutzeroberfläche

Siehe Befehlsgruppen,  
Befehlsliste

- Befehle für dynamische Objekte
- Deskriptor-Befehle
- Oberflächenobjektbefehle
- Objektinformationsbefehle

Deskriptor-Befehle

Befehle zum Umgang mit Deskriptoren


Siehe Befehlsgruppen,  
Befehlsliste

## **Befehle**

- HdlEnum
- HdlInfo
- HdlLink

## **Konstanten**

- HdlExists
- HdlSubType
- HdlType

obj -> HdEnum([int1]) : 

handle

Deskriptor enumerieren

obj Vorgänger-Deskriptor

int1 Deskriptor-Typ  
(optional)

Resultat handle Deskriptor

Siehe Verwandte Befehle,  
Deskriptoren

Mit dieser Funktion können alle aktiven Deskriptoren enumeriert werden.

Wird als Vorgänger-Deskriptor (obj) 0 angegeben, wird der erste Deskriptor, sonst der nachfolgende Deskriptor ermittelt. Aufgrund der internen Verwaltung der Deskriptoren werden diese nicht in numerisch aufsteigender Reihenfolge ermittelt.

Optional kann ein Deskriptor-Typ (int1) (siehe HdlType) angegeben werden, um nur Deskriptoren dieses Typs zu ermitteln.



Sollen alle Deskriptoren eines Typs entfernt werden, muss vor dem Entfernen der nächste Deskriptor ermittelt werden.

### Resultate

Ist der Vorgänger-Deskriptor (obj) ungültig, oder kein weiterer Deskriptor vorhanden, wird 0 zurückgegeben. Sonst wird der nächste Deskriptor zurückgegeben.

### Beispiele

```
// Alle Deskriptoren enumerierenfor tHdl # HdEnum(0);loop tHdl # tHdl->HdEnum();while (tHdl >
```



obj -> HdInfo(int1) : int

Deskriptor-Informationen ermitteln

obj      Deskriptor

Informationstyp

HdlExists      Deskriptor-Existenz  
ermitteln

int1      HdlType      Deskriptor-Typ  
ermitteln

HdlSubType Deskriptor-Untertyp  
ermitteln

Resultat int Deskriptor-Information

Siehe Verwandte Befehle, Deskriptoren

Mit dieser Funktion können verschiedene Informationen über eine Deskriptor (obj) ermittelt werden. Die möglichen Rückgabewerte sind bei den Optionen beschrieben.



obj -> Hdllink([int1]) : int

Deskriptor-Verknüpfung ermitteln/erstellen

obj      Deskriptor

int1      Verknüpfung (optional)

Resultat int Deskriptor-Verknüpfung

Siehe [Verwandte Befehle](#),  
[Deskriptoren](#), [Blog](#)

Mit dieser Funktion kann an den Deskriptor (obj) eine ganzzahlige Information (typischerweise ein weiterer Deskriptor) angehängt oder eine vorhandene Verknüpfung ermittelt werden. Zum Erstellen einer Verknüpfung muss (int1) gesetzt werden.

Mit diesem Befehl kann zum Beispiel eine einfach verkettete Liste von Oberflächen-Objekten aufgebaut werden, die zu einem späteren Zeitpunkt gemeinsam verarbeitet werden sollen. Zum Beispiel eine Liste von Objekten, deren Darstellung an bestimmte Ländereinstellungen angepasst werden müssen. Genauso kann an ein Oberflächen-Objekt der Deskriptor zu einem [globalen Datenbereich](#), einer [dynamischen Struktur](#) oder einem Element aus einer dynamischen Struktur angebunden werden. So können zu jedem Objekt weitere Informationen zur Verfügung gestellt werden.

Mögliche Laufzeitfehler:

ErrHdlInvalid Deskriptor (obj) ungültig

Konstanten für Deskriptor-Befehle  
Konstanten für Deskriptor-Befehle  
Siehe Deskriptor-Befehle

- HdlExists
- HdlSubType
- HdlType

\_HdlExists

Deskriptor-Existenz ermitteln

Wert 0 / 0x00

Verwandte

Siehe Befehle,

HdlInfo()

Option bei HdlInfo() durch die die Existenz eines Deskriptors ermittelt werden kann.

Falls Deskriptor vorhanden: Resultat = 1, ansonsten Resultat = 0.

## HdlSubType

Deskriptor-Untertyp ermitteln

Wert 2 / 0x02

### Verwandte

Siehe Befehle,

### HdlInfo()

Option bei HdlInfo() durch die der Untertyp eines Deskriptors ermittelt werden kann.

0 wird zurückgegeben, wenn es nicht um einen gültigen Deskriptor handelt, ansonsten einer der folgenden Werte:

- BinHdlDir (1)
- BinHdlObj (2)
- ChartDataHdlPie (2)
- ChartDataHdlPyramid (3)
- ChartDataHdlSurface (4)
- ChartDataHdlXy (1)
- ChartHdlPie (2)
- ChartHdlPyramid (3)
- ChartHdlSurface (4)
- ChartHdlXy (1)
- ComHdlArray (2)
- ComHdlObject (1)
- FsiHdlBufRead (2)
- FsiHdlBufWrite (4)
- FsiHdlDir (8)
- FsiHdlStd (1)
- HttpRecvRequest (2)
- HttpRecvResponse (4)
- HttpSendRequest (1)
- HttpSendResponse (3)
- JobProcess (2)
- JobThread (1)
- OdbcHdlApi (1)
- OdbcHdlClm (6)
- OdbcHdlCon (2)
- OdbcHdlStm (3)
- OdbcHdlStmParam (4)
- OdbcHdlTbl (5)
- StoHdlDir (1)
- StoHdlObj (2)
- TapiHdlDeviceItem (1)
- TapiHdlDeviceList (0)
- TapiHdlStatusItem (2)
- UrmTypeElmGroup (3)
- UrmTypeUser (1)
- UrmTypeUserGroup (2)

Wird der Untertyp von einem Datensatzpuffer ermittelt, wird die Dateinummer der dazugehörigen Datei zurückgegeben.





Ist der Datensatz von einer untergeordneten Datei, die keine eigenen Feldpuffer hat, wird die Nummer der Hauptdatei ermittelt.

**Beispiele:**

```
switch (tHdl->HdlInfo(_HdlSubType)){ case _BinHdlDir : ... case _BinHdlObj : .....if (tHdl->HdlInfo(_HdlSubType) == _BinHdlObj){
```

HdlType  
Deskriptor-Typ ermitteln  
Wert 1 / 0x01

Verwandte

Siehe Befehle,  
HdlInfo()

Option bei HdlInfo() durch die der Typ eines Deskriptors ermittelt werden kann.

0 wird zurückgegeben, wenn es sich um keinen gültigen Deskriptor handelt,  
ansonsten einer der folgenden Werte:

- HdlBinary (17)
- HdlChart (39)
- HdlChartData (40)
- HdlCom (15)
- HdlCteItem (18)
- HdlCteList (19)
- HdlCteNode (34)
- HdlCteTree (20)
- HdlDataSpace (6)
- HdlDDE (7)
- HdlDDEItem (9)
- HdlDLL (13)
- HdlFile (1)
- HdlFilter (2)
- HdlFsiMonitor (24)
- HdlHTTP (31)
- HdlJob (36)
- HdlJobControl (37)
- HdlLocale (14)
- HdlMail (11)
- HdlMem (30)
- HdlMsx (26)
- HdlNewSelection (28)
- HdlOdbc (41)
- HdlPDF (33)
- HdlRecBuf (5)
- HdlSelection (4)
- HdlSocket (12)
- HdlSocketListen (22)
- HdlStorage (25)
- HdlSvcSocket (35)
- HdlSvcTime (27)
- HdlSystem (16)
- HdlTapi (23)
- HdlText (8)
- HdlTheme (45)
- HdlThemeSet (46)
- HdlTimer (10)
- HdlUrm (29)
- HdlWinObject (3)

- HdlXML (32)
- HdlXmlReader (43)
- HdlXmlWriter (44)

Deskriptor-Typen

Deskriptor-Typen

Siehe Deskriptor-Befehle,  
Deskriptoren

## **Typen**

- HdlBinary
- HdlChart
- HdlChartData
- HdlCom
- HdlCteItem
- HdlCteList
- HdlCteNode
- HdlCteTree
- HdlDataSpace
- HdlDDE
- HdlDDEItem
- HdlDLL
- HdlFile
- HdlFilter
- HdlFsiMonitor
- HdlHTTP
- HdlJob
- HdlJobControl
- HdlLocale
- HdlMail
- HdlMem
- HdlMsx
- HdlNewSelection
- HdlOdbc
- HdlPDF
- HdlRecBuf
- HdlSelection
- HdlSocket
- HdlSocketListen
- HdlStorage
- HdlSvcSocket
- HdlSvcTime
- HdlSystem
- HdlTapi
- HdlText
- HdlTheme
- HdlTimer
- HdlUrm
- HdlValidation
- HdlWinObject
- HdlXML
- HdlXmlReader
- HdlXmlWriter

## **Untertypen**

- BinHdlDir
- BinHdlObj
- ChartDataHdlPie
- ChartDataHdlPyramid
- ChartDataHdlSurface
- ChartDataHdlXy
- ChartHdlPie
- ChartHdlPyramid
- ChartHdlSurface
- ChartHdlXy
- ComHdlArray
- ComHdlObject
- FsiHdlBufRead
- FsiHdlBufWrite
- FsiHdlDir
- FsiHdlStd
- HttpRecvRequest
- HttpRecvResponse
- HttpSendRequest
- HttpSendResponse
- JobProcess
- JobThread
- OdbcHdlApi
- OdbcHdlClm
- OdbcHdlCon
- OdbcHdlStm
- OdbcHdlStmParam
- OdbcHdlTbl
- StoHdlDir
- StoHdlObj
- TapiHdlDeviceItem
- TapiHdlDeviceList
- TapiHdlStatusItem
- UrmTypeElmGroup
- UrmTypeUser
- UrmTypeUserGroup
- VldHdlDir
- VldHdlObj

\_BinHdlDir  
Verzeichnis-Deskriptor  
Wert 1

Verwandte  
Siehe Befehle,  
HdlInfo(),  
BinDirOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Verzeichnisses hinweist.

\_BinHdlObj  
Objekt-Deskriptor  
Wert 2

Verwandte

Siehe Befehle,  
HdlInfo(),  
BinOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines binären Objekts hinweist.

\_ComHdlArray  
COM-Array-Deskriptor  
Wert 2

Verwandte

Befehle,

Siehe HdlInfo(),  
ComOpen(),  
Array

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines COM-Arrays hinweist.



\_ComHdlObject  
COM-Objekt-Deskriptor  
Wert 1

Verwandte

Siehe Befehle,  
HdlInfo(),  
ComOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines COM-Objekts hinweist.

\_FsiHdlBufRead  
Datei-Deskriptor  
Wert 2

Verwandte  
Siehe Befehle,  
HdlInfo(),  
FsiOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor einer Datei mit gepuffertem Lesezugriff hinweist.

Der Deskriptor muss mit dem Befehl FsiOpen() mit den Optionen \_FsiBuffer und \_FsiAcsR angelegt worden sein.

\_FsiHdlBufWrite  
Datei-Deskriptor  
Wert 4

Verwandte

Siehe Befehle,  
HdlInfo(),  
FsiOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor einer Datei mit gepuffertem Schreibzugriff hinweist.

Der Deskriptor muss mit dem Befehl FsiOpen() mit den Optionen \_FsiBuffer und \_FsiAcsW angelegt worden sein.

\_FsiHdlDir  
Verzeichnis-Deskriptor  
Wert 8

Verwandte  
Siehe Befehle,  
HdlInfo(),  
FsiDirOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Verzeichnisses hinweist.

\_FsiHdlStd  
Datei-Deskriptor  
Wert 1

Verwandte  
Siehe Befehle,  
HdlInfo(),  
FsiOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor einer Datei mit Standard-Optionen hinweist.

Der Deskriptor muss mit dem Befehl FsiOpen() mit der Option \_FsiStdRead oder \_FsiStdWrite angelegt worden sein.

\_HdlBinary  
Objekt/Verzeichnis-Deskriptor  
Wert 17

Verwandte

Befehle,

Siehe HdlInfo(),

BinOpen(),

BinDirOpen()

Rückgabewert von HdlInfo() mit der Option \_\_HdlType der auf einen Deskriptor eines binären Objekts/Verzeichnisses hinweist.

Über die Option \_\_HdlSubType kann ermittelt werden, ob es sich um ein binäres Objekt oder ein Verzeichnis handelt.

\_HdlChart  
Chart-Objekt-Deskriptor  
Wert 39

Verwandte

Siehe Befehle,  
HdlInfo(),  
ChartOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Diagramm-Objekts hinweist.

\_HdlChartData  
ChartData-Objekt-Deskriptor  
Wert 40

Verwandte

Siehe Befehle, HdlInfo(),  
ChartDataOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines ChartData-Objekts hinweist.



HdlCom  
COM-Objekt-Deskriptor  
Wert 15

Verwandte  
Siehe Befehle,  
HdlInfo(),  
ComOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines COM-Objekts hinweist.

HdlCteItem  
Element-Deskriptor  
Wert 18

Verwandte

Siehe Befehle,  
HdlInfo(),  
CteOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Elements einer dynamischen Struktur hinweist.

\_HdlCteList  
Listenstruktur-Deskriptor  
Wert 19

Verwandte  
Siehe Befehle,  
HdlInfo(),  
CteOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer Liste einer dynamischen Struktur hinweist.

\_HdlCteNode  
Knoten-Deskriptor  
Wert 34

Verwandte

Siehe Befehle,  
HdlInfo(),  
CteOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Knotens einer dynamischen Struktur hinweist.

\_HdlCteTree  
Baumstruktur-Deskriptor  
Wert 20

Verwandte

Siehe Befehle,  
HdlInfo(),  
CteOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer sortierten Liste einer dynamischen Struktur hinweist.

HdlDataSpace  
Datenbereich-Deskriptor  
Wert 6

Verwandte  
Siehe Befehle,  
HdlInfo(),  
VarAllocate()

Rückgabewert von HdlInfo() mit der Option HdlType der auf einen Deskriptor eines globalen Datenbereichs hinweist.

HdlDDE  
DDE-Kanal-Deskriptor  
Wert 7

Verwandte  
Siehe Befehle,  
HdlInfo(),  
DdeInit()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines DDE-Kanals hinweist.

\_HdlDDEItem  
DDE-Element-Deskriptor  
Wert 9

Verwandte

Siehe Befehle,  
HdlInfo()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines DDE-Elements hinweist.



HdlDLL  
DLL-Deskriptor  
Wert 13

Verwandte  
Siehe Befehle,  
HdlInfo(),  
DllLoad()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer geladenen DLL hinweist.

\_HdlFile  
Datei-Deskriptor  
Wert 1

Verwandte  
Siehe Befehle,  
HdlInfo(),  
FsiOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer externen Datei hinweist.

\_HdlFilter  
Filter-Deskriptor  
Wert 2

Verwandte

Siehe Befehle, HdlInfo(),  
RecFilterCreate()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Datensatzfilters hinweist.

HdlFsiMonitor  
Deskriptor einer Dateiüberwachung  
Wert 24

Verwandte

Siehe Befehle, HdlInfo(),  
FsiMonitorOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer externen Dateiüberwachung hinweist.

\_HdlHTTP

Deskriptor auf ein Http-Objekt

Wert 31

Verwandte

Siehe Befehle,

HdlInfo(),

HttpOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines HTTP-Objekts hinweist.

\_HdlJob  
Deskriptor eines Job-Objekts  
Wert 36

Verwandte  
Siehe Befehle,  
HdlInfo(),  
Job

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Job-Objekts hinweist.

\_HdlJobControl  
Deskriptor eines JobControl-Objekts  
Wert 37

Verwandte  
Siehe Befehle,  
HdlInfo(),  
JobControl

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines JobControl-Objekt hinweist.

\_HdlLocale  
Locale-Objekt-Deskriptor  
Wert 14

Verwandte

Siehe Befehle,  
HdlInfo(),  
LocaleLoad()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Locale-Objekts hinweist.



\_HdlMail  
Mail-Objekt-Deskriptor  
Wert 11

Verwandte  
Siehe Befehle,  
HdlInfo(),  
MailOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Mail-Objekts hinweist.

HdlMem  
Deskriptor auf ein Memory-Objekt  
Wert 30

Verwandte

Siehe Befehle,  
HdlInfo(),  
Memory

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Memory-Objekts hinweist.

HdlMsx  
Message-Exchange-Deskriptor  
Wert 26

Verwandte  
Siehe Befehle,  
HdlInfo(),  
MsxOpen()

Rückgabewert von HdlInfo() mit der Option HdlType der auf einen Deskriptor eines Nachrichtenkanals hinweist.

HdlNewSelection  
Neuer Selektions-Deskriptor  
Wert 28

Verwandte  
Siehe Befehle,  
HdlInfo(),  
SelCreate()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer dynamischen Selektion hinweist.

\_HdlPDF  
PDF-Objekt-Deskriptor  
Wert 33

Verwandte  
Siehe Befehle,  
HdlInfo(),  
PdfOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines PDF-Objekts hinweist.

\_HdlOdbc

Deskriptor eines ODBC-Objekts

Wert 41

Verwandte Befehle,

Siehe HdlInfo(), Befehle für

ODBC-Verbindungen

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines OdbcApi-Objekts hinweist.

HdlRecBuf  
Datensatzpuffer-Deskriptor  
Wert 5

Verwandte  
Siehe Befehle,  
HdlInfo(),  
RecBufCreate()

Rückgabewert von HdlInfo() mit der Option HdlType der auf einen Deskriptor eines Datensatzpuffers hinweist.

Die Dateinummer des Datensatzpuffers kann über den HdlSubType ermittelt werden.

\_HdlSelection  
Selektions-Deskriptor  
Wert 4

Verwandte

Siehe Befehle,  
HdlInfo(),  
SelOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer offenen Selektion hinweist.



\_HdlSocket  
Socket-Verbindungs-Deskriptor  
Wert 12

Verwandte

Siehe Befehle,  
HdlInfo(),  
SckConnect()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer Socket-Verbindung hinweist.

\_HdlSocketListen  
Socket-Verbindungs-Deskriptor  
Wert 22

Verwandte


Siehe Befehle,  
HdlInfo(),  
SckListen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor einer passiven Socket-Verbindung hinweist.

HdlStorage  
Deskriptor eines Storage-Objekts  
Wert 25

Verwandte

Siehe Befehle,  
HdlInfo(),  
Storage-Objekte

Rückgabewert von HdlInfo() mit der Option HdlType der auf einen Deskriptor eines Storage-Objekts  hinweist.

\_HdlSvcSocket

Deskriptor eines Task mit Betriebsart SOCKET

Wert 35

Verwandte

Siehe Befehle,  
HdlInfo(),  
mode

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Task mit Betriebsart SOCKET hinweist.

\_HdlSvcTime  
Deskriptor eines Task mit Betriebsart TIME  
Wert 27

Verwandte  
Siehe Befehle,  
HdlInfo(),  
mode

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Task mit Betriebsart TIME hinweist.

\_HdlSystem  
System-Objekt-Deskriptor  
Wert 16

Verwandte  
Siehe Befehle,  
HdlInfo(),  
\_Sys

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines System-Objekts hinweist.

HdlTapi  
Tapi-Objekt-Deskriptor  
Wert 23

Verwandte  
Siehe Befehle,  
HdlInfo(),  
TapiOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Tapi-Objekts hinweist.

HdlText  
Text-Deskriptor  
Wert 8

Verwandte  
Siehe Befehle,  
HdlInfo(),  
TextOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines geöffneten Texts hinweist.



HdlTheme

Theme-Deskriptor

Wert 45

Verwandte

Siehe Befehle, HdlInfo(),

WinThemeOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Theme-Objekts hinweist.

\_HdlTimer  
Timer-Deskriptor  
Wert 10

Verwandte

Siehe Befehle, HdlInfo(),  
SysTimerCreate()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Timer-Ereignisses hinweist.

HdlUrm  
Deskriptor auf ein Objekt der Benutzerverwaltung  
Wert 29

Verwandte

Siehe Befehle,  
HdlInfo(),  
UrmOpen()

Rückgabewert von HdlInfo() mit der Option HdlType der auf einen Deskriptor eines Objekts der Benutzerverwaltung hinweist. Genauere Informationen werden bei der Ermittlung des HdlSubType. Dort wird dann eine der Konstanten UrmTypeUser, UrmTypeUserGroup oder UrmTypeElmGroup zurückgegeben.

\_HdlValidation  
Deskriptor eines Validierungsverzeichnisses bzw. Validierungselementes  
Wert 47

Verwandte

Befehle,

Siehe HdlInfo(),

VldOpen(),

VldDirOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Validierungsverzeichnisses bzw. Validierungselementes hinweist.

Über die Option \_HdlSubType kann ermittelt werden, ob es sich um ein Validierungselement oder ein Verzeichnis handelt.

\_HdlWinObject  
Oberflächen-Objekt-Deskriptor  
Wert 3

Verwandte

Siehe Befehle,  
HdlInfo(),  
\_App

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines Oberflächen-Objekts oder des \_App-Objekts hinweist.

\_HdlXML  
XML-Objekt-Deskriptor  
Wert 32

Verwandte  
Siehe Befehle,  
HdlInfo(),  
XmlLoad()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines XML-Objekts hinweist.

\_HdlXmlReader  
XmlReader-Deskriptor  
Wert 43

Verwandte

Siehe Befehle, HdlInfo(),  
XmlOpenReader()

Rückgabewert von HdlInfo() mit der Option \_HdlType der auf einen Deskriptor eines  
XmlReader-Objekts hinweist.

HdlXmlWriter  
XmlWriter-Deskriptor  
Wert 44

Verwandte

Siehe Befehle, HdlInfo(),  
XmlOpenWriter()

Rückgabewert von HdlInfo() mit der Option HdlType der auf einen Deskriptor eines  
XmlWriter-Objekts hinweist.



\_OdbcHdlApi

ODBC-Schnittstellen-Deskriptor

Wert 1

Verwandte

Siehe Befehle,

HdlInfo(),

OdbcOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines ODBC-Schnittstellen-Objekts hinweist.

\_OdbcHdlCln  
ODBC-Spalten-Deskriptor  
Wert 6

Verwandte Befehle,  
Siehe HdlInfo(),  
OdbcCatalogCln()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines ODBC-Spalten-Objekts hinweist.

\_OdbcHdlCon

ODBC-Verbindungs-Deskriptor

Wert 2

Verwandte Befehle,

Siehe HdlInfo(),

OdbcConnect(),

OdbcConnectDriver()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines ODBC-Verbindungs-Objekts hinweist.

\_OdbcHdlStm  
ODBC-Statement-Deskriptor  
Wert 3

Verwandte Befehle,

Siehe HdlInfo(),  
OdbcExecuteDirect()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType, der auf einen Deskriptor eines ODBC-Statement-Objekts hinweist, dass mit OdbcExecuteDirect() angelegt wurde.

\_OdbcHdlStmParam  
ODBC-Statement-Deskriptor  
Wert 4

Verwandte

Siehe Befehle,  
HdlInfo(),  
OdbcPrepare()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType, der auf einen Deskriptor eines ODBC-Statement-Objektes hinweist, dass mit OdbcPrepare() angelegt wurde.

\_OdbcHdlTbl  
ODBC-Tabellen-Deskriptor  
Wert 5

Verwandte

Siehe Befehle, HdlInfo(),  
OdbcCatalogTbl()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines ODBC-Tabellen-Objekts hinweist.

\_StoHdlDir  
Storage-Verzeichnis  
Wert 1

Verwandte  
Siehe Befehle,  
HdlInfo(),  
StoDirOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Storage-Verzeichnisses hinweist.

\_StoHdlObj  
Storage-Objekt  
Wert 2

Verwandte  
Siehe Befehle,  
HdlInfo(),  
StoOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Storage-Objekts hinweist.



\_TapiHdlDeviceItem

Deskriptor auf ein Cte-Item mit TAPI-Geräteinformationen

Wert 1

Verwandte

Siehe Befehle,

HdlInfo(),

TapiOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Cte-Items mit Geräteinformationen über ein TAPI-Gerät hinweist. Die Liste wurde mit der Anweisung TapiOpen() gefüllt.

\_TapiHdlDeviceList

Deskriptor auf eine Cte-Liste mit TAPI-Geräteinformationen

Wert 0

Verwandte

Siehe Befehle,

HdlInfo(),

TapiOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Cte-Liste mit Geräteinformationen über TAPI-Geräte hinweist. Die Liste wurde mit der Anweisung TapiOpen() erstellt und gefüllt.

\_TapiHdlStatusItem

Deskriptor auf ein Cte-Item mit TAPI-Statusinformationen

Wert 2

Verwandte

Siehe Befehle,  
HdlInfo(),  
TapiInfo()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Cte-Items mit Statusinformationen über ein TAPI-Gerät hinweist. Die Liste wurde mit der Anweisung TapiInfo() gefüllt.

VldHdlDir  
Deskriptor eines Validierungsverzeichnisses  
Wert 1

Verwandte  
Siehe Befehle,  
HdlInfo(),  
VldDirOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Validierungsverzeichnisses hinweist.

\_VldHdlObj

Deskriptor eines Validierungselementes

Wert 2

Verwandte

Siehe Befehle,  
HdlInfo(),  
VldOpen()

Rückgabewert von HdlInfo() mit der Option \_HdlSubType der auf einen Deskriptor eines Validierungselementes hinweist.

Objektinformationsbefehle

Befehle zum Ermitteln von Objektinformationen

Siehe Befehlsgruppen,

Befehlsliste

## **Befehle**

- WinInfo

## **Konstanten**

- WinApplication
- WinClientHeight
- WinClientWidth
- WinContextMenu
- WinCount
- WinDynamic
- WinErrorCode
- WinFirst
- WinFocusKey
- WinFrame
- WinFrameActive
- WinFrameForeground
- WinHighContrast
- WinIndex
- WinInteriorHeight
- WinInteriorWidth
- WinItem
- WinLast
- WinLstEditObject
- WinMenu
- WinNext
- WinNodeMouseSelect
- WinObject
- WinObjectActive
- WinObjectMaximized
- WinParent
- WinPrev
- WinRoot
- WinScreenBPP
- WinScreenDpiX
- WinScreenDpiY
- WinScreenHeight
- WinScreenWidth
- WinState
- WinThemeActive
- WinType
- WinUnicode



obj -> WinInfo(int1[, int2, int3]) : int  
Oberflächen-Objekt-Informationen ermitteln

obj Oberflächenobjekt

int1 Optionen (siehe Text)

int2 Zähler (optional)

int3 Objekttyp (optional)

Resultat int Objektinformation (siehe Text)

Siehe WinSearch(), HdlInfo(),  
OrderPass, Blog

Mit diesem Befehl können verschiedene Informationen zu dem übergebenen Objekt (obj) oder zu der Umgebung, in der die Applikation gestartet wurde, ermittelt werden. Zudem können relativ zum angegebenen Objekt, andere Objekte ermittelt werden. Die unterschiedlichen Funktionen werden durch die Übergabe entsprechender Konstanten in (int1) aufgerufen.

Die Funktionen können in folgende Bereiche aufgeteilt werden:

- Navigation durch den Objektbaum
- Objekte nach Typ iterieren
- Ermitteln anderer Objekte
- Informationen über das Objekt
- Informationen über die Umgebung

## Navigation durch den Objektbaum

Werden die folgenden Konstanten übergeben, ist die Rückgabe des Befehls immer der Deskriptor eines Objekts. In (obj) kann ein beliebiges Oberflächenobjekt übergeben werden.

- WinFirst

Das Resultat ist das erste untergeordnete Objekt des Objekts (obj).

- WinLast

Das Resultat ist das letzte untergeordnete Objekt des Objekts (obj).

- WinNext

Das Resultat ist das nachfolgende Objekt des Objekts (obj).

- WinPrev

Das Resultat ist das vorhergehende Objekt des Objekts (obj).

- WinParent

Das Resultat ist das übergeordnete Objekt des Objekts (obj).

- WinRoot

Das Resultat ist das Wurzel-Objekt des Objekts (obj).

- WinFrame

Das Resultat ist das übergeordnete Frame-Objekt des Objekts (obj).



Die Reihenfolge, in der über die Parameter \_WinFirst, \_WinPrev, \_WinNext und \_WinLast zugegriffen wird, entspricht der Erstellungsreihenfolge der einzelnen Objekte. Die Intervall-Objekte innerhalb des GanttGraph-Objekts werden vom Betriebssystem nicht in einer Liste verwaltet. Die Reihenfolge beim Lesen der Intervalle kann somit von der Erstellungsreihenfolge abweichen. Bei den Objekten Notebook, RecList und DataList kann alternativ über die Eigenschaft OrderPass die Anzeigereihenfolge ermittelt werden.

In (int3) kann ein Objekt-Typ angegeben werden. Als Werte stehen die bei \_WinType beschriebenen Konstanten zur Verfügung. Der Objekt-Typ wird nur in Kombination mit den Optionen \_WinFirst, \_WinPrev, \_WinNext, \_WinLast und \_WinParent ausgewertet. Es wird das entsprechende Objekt des angegebenen Typs zurückgegeben.

### Beispiel:

```
// Übergeordnetes Notebook-Objekt ermittelntObj->WinInfo(_WinParent, 0, _WinTypeNotebook);
```

### Objekte nach Typ iterieren

Werden Objekte nach Typ iteriert, stellt der übergebene Objekttyp dabei nicht immer den exakten Objekttyp dar, sondern eine Objektgruppe.

### Beispiel:

```
for tChild # tParent->WinInfo(_WinFirst, 1, _WinTypeEdit);loop tChild # tChild->WinInfo(_WinNext,
```

Die Schleife ermittelt nicht nur Edit-Objekte vom Typ \_WinTypeEdit, sondern auch DateEdit, FloatEdit, TextEdit, etc, da diese zur Objektgruppe der Eingabeobjekte gehören.

Die folgende Schleife ermittelt dagegen nur DateEdit-Objekte, da diese keine eigene Objektgruppe bilden.

```
for tChild # tParent->WinInfo(_WinFirst, 1, _WinTypeDateEdit);loop tChild # tChild->WinInfo(_WinNext,
```

Folgende Objektgruppen existieren:

Eingabeobjekte ( <u>_WinTypeEdit</u> )	<u>Edit</u> , <u>IntEdit</u> , <u>BigIntEdit</u> , <u>FloatEdit</u> , <u>DecimalEdit</u> , <u>TimeEdit</u> , <u>DateEdit</u> , <u>ColorEdit</u> , <u>FontNameEdit</u> , <u>FontSizeEdit</u> , <u>TextEdit</u>
Frame-Objekte ( <u>_WinTypeFrame</u> )	<u>Frame</u> , <u>AppFrame</u> , <u>MdiFrame</u> , <u>TrayFrame</u> , <u>Dialog</u>
Schaltflächen ( <u>_WinTypeButton</u> )	<u>Button</u> , <u>ColorButton</u> , <u>Checkbox</u> , <u>Radiobutton</u>
Schalter ( <u>_WinTypeCheckBox</u> )	<u>Checkbox</u> , <u>Radiobutton</u>
Label-Objekte ( <u>_WinTypeLabel</u> )	<u>Label</u> , <u>HyperLink</u>
Toolbar-Objekte ( <u>_WinTypeToolbar</u> )	<u>Toolbar</u> , <u>Statusbar</u>
PrintForm-Objekte ( <u>_PrtTypePrintForm</u> )	<u>PrintForm</u> , <u>PrintDoc</u> , <u>PrintDocRecord</u> , <u>PrintFormList</u>

Sollen nur Objekte des angegebenen Typs ermittelt werden, muss die Konstante \_WinTypeExact mit dem Objekttyp verodert werden:



```
for tChild # tParent->WinInfo(_WinFirst, 1, _WinTypeEdit | _WinTypeExact);loop tChild # tChild
```

Die Konstante wird bei den Optionen WinFirst, WinLast, WinNext, WinPrev und WinParent ausgewertet und bei allen anderen Optionen ignoriert.

## Ermitteln anderer Objekte

Bei diesen Funktionen werden bestimmte Objekte ermittelt. Als Übergabeparameter müssen entweder bestimmte Objekttypen oder 0 übergeben werden. Welche Objekte übergeben werden können, ist bei den Konstanten erläutert.

- WinApplication

Das Resultat ist das Application-Objekt. In (obj) kann 0 oder ein beliebiges Objekt angegeben werden.

- WinFrameActive

Das Resultat ist das aktive MDIFrame-Objekt des in (obj) übergebenen AppFrame-Objekts.

- WinObject

Das Resultat ist ein bestimmtes Unterobjekt des Objekts (obj). Das übergebene Objekt muss von einem bestimmten Typ sein.

- WinObjectActive

Das Resultat ist das aktive GroupTile-Objekt des in (obj) übergebenen GroupSplit-Objekts.

- WinObjectMaximized

Das Resultat ist das maximierte GroupTile-Objekt des in (obj) übergebenen GroupSplit-Objekts.

- WinMenu

Das Resultat ist das Menü des in (obj) übergebenen Frame bzw. AppFrame-Objekts.

- WinContextMenu

Das Resultat ist das Kontextmenü-Objekt des Objekts (obj).

- WinLstEditObject

Wird in einem RecList- oder DataList-Objekt ein Eintrag editiert, wird mit dieser Konstante das Editier-Objekt zurückgegeben.

- WinFrameForeground

Das Resultat ist das Vordergrundfenster. Als Objekt (obj) muss 0 als Zähler (int2) muss 0 oder 1 angegeben werden.

## Informationen über das Objekt

In dieser Gruppe von Konstanten werden Informationen über das in (obj) angegebene Objekt zurückgegeben. Bei einigen Konstanten können nur bestimmte Objekttypen

übergeben werden. Dies ist bei den Konstanten erläutert.

- WinCount

Das Resultat ist die Anzahl der untergeordneten Objekte des Objekts (obj).

- WinClientHeight

Es wird die Höhe des Clientbereichs (inklusive Tool- und Windowbars) zurückgegeben. Es können nur Objekte vom Typ Frame, AppFrame oder MdiFrame übergeben werden.

- WinClientWidth

Es wird die Breite des Clientbereichs (inklusive Tool- und Windowbars) zurückgegeben. Es können nur Objekte vom Typ Frame, AppFrame oder MdiFrame übergeben werden.

- WinInteriorHeight

Es wird die Höhe des Anzeigebereiches ohne Menü, Tool- und Windowbars zurückgegeben. Bei AppFrame-Objekten entspricht das der Höhe des Bereiches, in dem MdiFrame-Objekte dargestellt werden können. Es können nur Objekte vom Typ Frame bzw. AppFrame übergeben werden.

- WinInteriorWidth

Es wird die Breite des Anzeigebereiches ohne Menü, Tool- und Windowbars zurückgegeben. Bei AppFrame-Objekten entspricht das der Breite des Bereiches, in dem MdiFrame-Objekte dargestellt werden können. Es können nur Objekte vom Typ Frame bzw. AppFrame übergeben werden.

- WinType

Das Resultat ist der Typ des Objekts (obj).

- WinFocusKey

Das Resultat ist die Taste, mit der das Objekt verlassen wurde.

- WinIndex

Das Resultat ist die Position des Objekts (obj) innerhalb des übergeordneten Objekts.

- WinItem

Das Resultat ist die Position der Datenzeile des Spalten-Objekts (obj) eines DataList-Objekts.

- WinState

Das Resultat ist der Status des Frame/AppFrame/MDIFrame- oder des GroupTile-Objekts (obj).

- WinDynamic

Wird ein Objekt (obj) angegeben, gibt der Befehl zurück, ob es dynamisch erstellt wurde (1) oder nicht (0). Ein Objekttyp (int3) darf in diesem Fall nicht angegeben werden.

Wird als Objekt 0 und in (int3) ein Objekttyp angegeben, gibt der Befehl zurück,

ob der Objekttyp mit WinCreate() erzeugt werden kann.

- WinUnicode

Wird als Resultat 1 zurückgegeben, unterstützt das Objekt (obj) Unicode-Eigenschaften, bei 0 nicht. Die betreffenden Eigenschaften sind auf der Seite Unicode-Unterstützung ersichtlich.

## Informationen über die Umgebung

Mit diesen Konstanten werden Parameter der Umgebung abgefragt. Außer bei der Ermittlung des Fehlerwertes muss kein Objekt übergeben werden. Anstelle des Deskriptors wird der Wert 0 angegeben.

WinInfo(0, int1[, int2, int3])

- WinScreenWidth

Das Resultat ist die horizontale Bildschirmauflösung.

- WinScreenHeight

Das Resultat ist die vertikale Bildschirmauflösung.

- WinScreenDpiX

Das Resultat ist die horizontale Bildschirmauflösung in DPI.

- WinScreenDpiY

Das Resultat ist die vertikale Bildschirmauflösung in DPI.

- WinScreenBBP

Das Resultat ist die aktuelle Bildschirmfarbtiefe.

- WinThemeActive

Das Resultat ist die Aktivität der betriebssystemabhängigen Darstellung.

- WinHighContrast

Das Resultat ist 1, wenn Windows im Modus "Hoher Kontrast" läuft, sonst 0.

- WinErrorCode

Das Resultat ist der Windows-Fehlerwert. Der Fehlerwert muss nur nach der Installation zur Darstellung von Office-Dateien (InstallCtxOffice) abgefragt werden.

- WinNodeMouseSelect

Das Resultat ist die Maustaste, mit der zuletzt ein TreeNode in einem TreeView-Objekt selektiert wurde.

## Beispiel 1:

```
// Alle Objekte eines Fensters ermitteln@A+@C+sub AllObj( aStartObj : handle;) local { tObj
```

## Beispiel 2:

## Kontakt

```
// Alle Einträge eines Menüs ermitteln// Menü des Fensters ermitteltMenu # tFrame->WinInfo(_WinM
```

### Beispiel 3:

```
// Bestimmte Objekte eines Gantt-Graphs ermittelnsb Info( aGanttGraph : handle; // Übergebene
```

Der Parameter (int2) wird nur in Kombination mit den Optionen \_WinPrev, \_WinNext und \_WinParent ausgewertet. Mit dem Zähler kann eine Schrittweite übergeben werden.

### Beispiel:

```
// Übernächstes Objekt ermitteltObj->WinInfo(_WinNext, 2);// Großvater-Objekt ermitteltObj->Win
```

Bei der Angabe von negative Werten in (int2) liefert der Befehl den Wert 0 zurück.

Mögliche Laufzeitfehler:

<u>_ErrHdlInvalid</u>	Oberflächenobjekt (obj) ungültig oder muss 0 sein. Bei der Option <u>_WinDynamic</u> wurde sowohl ein Objekt (obj) als auch ein Objekttyp (int3) angegeben.
<u>_ErrValueInvalid</u>	Bei der Option <u>_WinFrameForeground</u> wurde ein Zähler (int2) < 0 oder > 1 angegeben.

Konstanten für Objektinformationsbefehle

Konstanten für Objektinformationsbefehle

Siehe Objektinformationsbefehle

- WinApplication
- WinClientHeight
- WinClientWidth
- WinContextMenu
- WinCount
- WinDynamic
- WinErrorCode
- WinFirst
- WinFocusKey
- WinFrame
- WinFrameActive
- WinFrameForeground
- WinHighContrast
- WinIndex
- WinInteriorHeight
- WinInteriorWidth
- WinItem
- WinLast
- WinLstEditObject
- WinMenu
- WinNext
- WinNodeMouseSelect
- WinObject
- WinObjectActive
- WinObjectMaximized
- WinParent
- WinPrev
- WinRoot
- WinScreenBPP
- WinScreenDpiX
- WinScreenDpiY
- WinScreenHeight
- WinScreenWidth
- WinState
- WinThemeActive
- WinType
- WinUnicode

\_WinApplication  
Application-Objekt ermitteln  
Wert 3000

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das Application-Objekt ermittelt werden kann.

\_WinClientHeight  
Höhe des Clientsbereichs  
Wert 1005

Verwandte Befehle,  
Siehe WinInfo(),  
\_WinClientWidth,  
\_WinInteriorHeight

Option bei WinInfo() durch die die Höhe des Clientbereichs eines Fensters ermittelt werden kann. Der Clientbereich ist die Größe des Fensters ohne Titel- und Menüzeile, sowie Fensterrahmen. Eventuell vorhandene Tool-, Window- oder Statusbars sind im Clientbereich enthalten.

\_WinClientWidth  
Breite des Client-Bereichs  
Wert 1004

Verwandte Befehle,  
Siehe WinInfo(),  
\_WinClientHeight,  
\_WinInteriorWidth

Option bei WinInfo() durch die die Breite des Clientbereichs eines Fensters ermittelt werden kann. Der Clientbereich ist die Größe des Fensters ohne Titel- und Menüzeile, sowie Fensterrahmen. Eventuell vorhandene Tool-, Window- oder Statusbars sind im Clientbereich enthalten.



\_WinContextMenu  
Kontextmenü-Objekt ermitteln  
Wert 12

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das Kontextmenü-Objekt eines Objekts ermittelt werden kann.

Mit dieser Konstante wird innerhalb des Ereignisses EvtMenuInitPopup der Deskriptor des aktuellen Kontextmenüs (sofern in dem Argument aMenuItem 0 übergeben wurde) ermittelt.

\_WinCount  
Anzahl untergeordneter Objekte ermitteln  
Wert 9

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die die Anzahl der unmittelbar untergeordneten Objekte ermittelt werden kann.

Beim GanttGraph-Objekt ist dies die Anzahl der Intervalle, beim AppFrame die Anzahl der MdiFrames, beim RecList-, RecView- und DataList-Objekt die Anzahl der Spalten. Beim Spalten-Objekt vom RecView wird die Anzahl der SubColumns ermittelt.

\_WinDynamic

Ermitteln, ob das Objekt dynamisch erstellt wurde

Wert 18

Verwandte

Siehe Befehle,

WinInfo(),

WinCreate()

Option bei WinInfo() durch die ermittelt werden kann, ob das Objekt dynamisch erstellt wurde.

Das Ergebnis ist 1, wenn das Objekt dynamisch erstellt wurde und 0, wenn nicht.

\_WinErrorCode  
Windows-Fehlerwert  
Wert 2500

Verwandte  
Siehe Befehle,  
WinInfo(),  
InstallCtxOffice

Option bei WinInfo() durch der Windows-Fehlerwert ermittelt werden kann. Es muss kein Objekt übergeben werden.

\_WinFirst

Erstes untergeordnetes Objekt ermitteln

Wert 4

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das erste untergeordnete Objekt eines Objekts ermittelt werden kann.

Die Reihenfolge wird bei den Objekten NoteBook, DataList und RecList durch die Eigenschaft OrderPass beeinflusst.

## Kontakt

\_WinFocusKey

Taste ermitteln, mit dem das Objekt verlassen wurde

Wert 2003

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die die Taste ermittelt werden kann, die dazu führte, dass das Objekt den Fokus verloren hat.

### **Beispiel:**

```
switch (aEvt:Obj->WinInfo(_WinFocusKey)){ case _WinKeyTab | _WinKeyShift : {    // Shift + TAB
```

\_WinFrame  
Fenster-Objekt ermitteln  
Wert 10

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die das Fenster-Objekt eines Objektes ermittelt werden kann. Dabei wird ausgehend vom übergebenen Objekt solange das Eltern-Objekt ermittelt, bis ein Objekt gefunden wird, das vom Typ AppFrame, MdiFrame oder Frame ist.

Wird der Deskriptor eines Fenster-Objektes übergeben, wird dieser auch zurückgegeben.

\_WinFrameActive

Aktives MDI-Fenster-Objekt ermitteln

Wert 2001

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das aktive MDI-Fenster-Objekt eines Applikations-Fenster-Objekts ermittelt werden kann.



\_WinFrameForeground  
Vordergrundfenster ermitteln  
Wert 2005

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die das Vordergrundfenster ermittelt werden kann. Es handelt sich dabei um das Fenster, mit dem der Anwender gerade arbeitet.

Die Funktion liefert immer den Deskriptor eines Frame-Objektes zurück und unterscheidet sich somit vom Befehl WinFocusGet(), der das Objekt mit dem Eingabefokus liefert.

Im Falle eines AppFrame-Objektes mit untergeordneten MdiFrame-Objekten, liefert die Funktion das AppFrame-Objekt. Zur Ermittlung des aktiven MdiFrame kann WinInfo() mit der Option \_WinFrameActive aufgerufen werden.

Die Funktion liefert 0 zurück, wenn kein Vordergrundfenster vorhanden ist oder es sich nicht um ein von CONZEPT 16 erstelltes Fenster handelt.

**Beispiele:**

```
// Vordergrundfenster ermitteltWin # WinInfo(0, _WinFrameForeground);// Vordergrundfenster nur z
```

\_WinHighContrast

Prüfen, ob Windows im Modus "Hoher Kontrast" läuft

Wert 1010

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die ermittelt werden kann, ob Windows im Modus "Hoher Kontrast" läuft.



Sofern eine Anwendung unter Modern Theme Style läuft, wird empfohlen, das Theme '\_WindowsColor', oder ein Theme, welches auf diesem basiert, zu verwenden.

### **Beispiel**

```
// Aktuelles Theme abhängig vom Windows-Modus "Hoher Kontrast" aktivierenif (WinInfo(0, _WinHighC
```

\_WinIndex  
Objektposition ermitteln  
Wert 11

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die die Position eines Objekts innerhalb des übergeordneten Objekts ermittelt werden kann.

\_WinInteriorHeight

Höhe des Bereiches, in dem MDI-Fenster dargestellt werden können

Wert 1009

Verwandte

Siehe Befehle, WinInfo(),

\_WinInteriorWidth

\_WinClientHeight

Option bei WinInfo(), durch die die Höhe des Anzeigebereiches ohne Menü, Tool-, Status- und Windowbars ermittelt werden kann. Bei AppFrame-Objekten entspricht das der Höhe des Bereiches, in dem MdiFrame-Objekte dargestellt werden können.

### **Beispiel:**

```
tInteriorHeight # $AppFrm->WinInfo(_WinInteriorHeight);
```

\_WinInteriorWidth

Breite des Bereiches, in dem MDI-Fenster dargestellt werden können

Wert 1008

Verwandte Befehle,

Siehe WinInfo(),

\_WinInteriorHeight

\_WinClientWidth

Option bei WinInfo(), durch die die Breite des Anzeigebereiches ohne Menü, Tool-, Status- und Windowbars ermittelt werden kann. Bei AppFrame-Objekten entspricht das der Breite des Bereiches, in dem MdiFrame-Objekte dargestellt werden können.

**Beispiel:**

```
tInteriorWidth # $AppFrm->WinInfo(_WinInteriorWidth);
```

\_WinItem

Datenzellenposition ermitteln

Wert 14

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die die Position der Datenzelle eines Spalten-Objektes eines DataList-Objektes ermittelt werden kann.

Die Position der Datenzelle kann bei den Befehlen WinLstCellSet() und WinLstCellGet() verwendet werden.

WinLast

Letztes untergeordnetes Objekt ermitteln

Wert 5

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das letzte untergeordnete Objekt eines Objekts ermittelt werden kann.

Die Reihenfolge wird bei den Objekten Notebook, DataList und RecList durch die Eigenschaft OrderPass beeinflusst.

\_WinLstEditObject  
Editier-Objekt ermitteln  
Wert 2004

Verwandte

Siehe Befehle,  
WinInfo(),  
WinLstEdit()

Option bei WinInfo() durch die das Editier-Objekt bei einer editierten RecList- oder DataList-Objekt ermittelt werden kann.



WinMenu

Menü-Objekt ermitteln

Wert 8

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das Menü-Objekt eines Fenster- / Applikations-Fenster-  
oder MenuButton-Objekts ermittelt werden kann.

WinNext  
Nachfolgendes Objekt ermitteln  
Wert 7

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das nachfolgende Objekt eines Objekts ermittelt werden kann.

Die Reihenfolge wird bei den Objekten Notebook, DataList und RecList durch die Eigenschaft OrderPass beeinflusst.

\_WinNodeMouseSelect

Maustaste ermitteln, mit der der Knoten selektiert wurde

Wert 2006

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die ermittelt werden kann, ob und mit welcher Taste ein TreeNode in einem TreeView-Objekt ausgewählt wurde. Der Wert wird jeweils vor dem Auslösen des Ereignisses EvtNodeSelect gesetzt. Bei aktivierter Mehrfachauswahl kann sich die Information auch auf das Selektieren mehrerer Knoten bzw. deselektieren von Knoten beziehen.

Der ermittelte Wert kann mit den folgenden Werten und Konstanten verglichen werden:

- \_WinMouseLeft - Selektionsaktion mit der linken Maustaste
- \_WinMouseRight - Selektionsaktion mit der rechten Maustaste
- 0 - Selektionsaktion unbekannt (Tastatur oder prozedural z. B. über CurrentInt-Eigenschaft).



Der Wert 0 wird auch zurückgegeben, wenn das TreeView erstmalig den Fokus erhält, kein TreeNode selektiert ist und dann mit der rechten Maustaste ein TreeNode selektiert wird.

\_WinObject  
Bestimmtes Objekt ermitteln  
Wert 15

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die ein bestimmtes Objekt ermittelt werden kann.

Bei einem GanttGraph kann das View-Objekt ermittelt werden.

Bei einem GroupSplit-Objekt kann über die Optionen \_WinObjectActive und \_WinObjectMaximized das Aktive bzw. Maximierte GroupTile-Objekt ermittelt werden.

**Beispiele:**

```
tView # tGanttGraph->WinInfo(_WinObject, <View-Nummer>, _WinTypeGanttView);
```

Ist bei einem GanttGraph-Objekt die Eigenschaft SplitStyle nicht auf \_WinSplitNone gesetzt, können durch ziehen des "Splitters" (über bzw. links vom Scrollbalken) ein, zwei oder vier Ansichten (Views) erzeugt werden.

Die Nummern der Ansichten werden von links nach rechts und von oben nach unten vergeben. Ist also ein GanttGraph-Objekt horizontal in zwei Ansichten geteilt, hat die obere die Nummer 1 und die untere die Nummer 2.

```
// Ermitteln, ob ein GroupTile-Objekt maximiert ist. tObj # gGroupSplit->WinInfo(_WinObject, _Win
```

\_WinObjectActive

Aktives GroupTile-Objekt ermitteln

Wert 1

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das aktive GroupTile-Objekt eines GroupSplit-Objekts ermittelt werden kann.

### **Beispiel**

```
// Aktives GroupTile-Objekt ermitteltObj # gGroupSplit->WinInfo(_WinObject, _WinObjectActive);
```

\_WinObjectMaximized

Maximiertes GroupTile-Objekt ermitteln

Wert 2

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das maximierte GroupTile-Objekt eines GroupSplit-Objekts ermittelt werden kann.

```
// Ermitteln, ob ein GroupTile-Objekt maximiert isttObj # gGroupSplit->WinInfo(_WinObject, _WinOb
```

WinParent  
Übergeordnetes Objekt ermitteln  
Wert 3

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das übergeordnete Objekt eines Objekts ermittelt werden kann.

\_WinPrev

Vorhergehendes Objekt ermitteln

Wert 6

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die das vorhergehende Objekt eines Objekts ermittelt werden kann.

Die Reihenfolge wird bei den Objekten Notebook, DataList und RecList durch die Eigenschaft OrderPass beeinflusst.



\_WinRoot  
Wurzel-Objekt ermitteln  
Wert 2

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die das Wurzel-Objekt eines Objekts ermittelt werden kann. Ausgehend vom übergebenen Objekt wird solange das Eltern-Objekt ermittelt, bis das Objekt erreicht wird, das kein Eltern-Objekt hat. Das Objekt wird zurückgegeben. Hierbei handelt es sich in der Regel um ein AppFrame- oder Frame-Objekt. Wird das Wurzel-Objekt eines MdiFrame abgefragt, wird im EvtInit der MdiFrame zurückgegeben.

\_WinScreenBPP  
Bildschirmfarbtiefe ermitteln  
Wert 1002

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die die aktuelle Bildschirmfarbtiefe in Bit ermittelt werden kann.

8 Bit = 256 Farben

15 Bit = HiColor

16 Bit = HiColor

24 Bit = TrueColor

32 Bit = TrueColor

\_WinScreenDpiX  
Horizontale DPI-Auflösung des Bildschirms  
Wert 1006

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die die horizontale Bildschirmauflösung in DPI (Dots per inch) ermittelt werden kann.

\_WinScreenDpiY  
Vertikale DPI-Auflösung des Bildschirms  
Wert 1007

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die die vertikale Bildschirmauflösung in DPI (Dots per inch) ermittelt werden kann.

WinScreenHeight

Vertikale Bildschirmauflösung ermitteln

Wert 1001

Verwandte

Siehe Befehle,

WinInfo(),

AreaScreen

Option bei WinInfo() durch die die vertikale Bildschirmauflösung des Primärbildschirms in Pixel ermittelt werden kann.

\_WinScreenWidth

Horizontale Bildschirmauflösung ermitteln

Wert 1000

Verwandte

Siehe Befehle,

WinInfo(),

AreaScreen

Option bei WinInfo() durch die die horizontale Bildschirmauflösung des Primärbildschirms in Pixel ermittelt werden kann.

\_WinState

Fensterstatus ermitteln

Wert 2000

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die der Status eines

Fensters/Applikations-Fensters/MDI-Fensters oder eines GroupTile-Objektes ermittelt werden kann.

Folgende Fenster-Statuswerte können zurückgegeben werden:

\_WinDialogMaximized Fenster maximiert

\_WinDialogMinimized Fenster minimiert

\_WinDialogNormal Fenster weder maximiert noch minimiert

**Beispiel:**

```
if (tFrame->WinInfo(_WinState) = _WinDialogMinimized){ // Dialog minimiert}
```

Folgende GroupTile-Statuswerte können zurückgegeben werden:

\_WinStateAttachMaximized GroupTile maximiert

\_WinStateAttachClosed GroupTile geschlossen

\_WinStateAttachNormal GroupTile weder maximiert noch geschlossen

**Beispiel:**

```
if (tGroupTile->WinInfo(_WinState) = _WinStateAttachClosed){ // GroupTile geschlossen}
```

\_WinThemeActive

Aktivität der betriebssystemabhängigen Darstellung ermitteln

Wert 1003

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die die Aktivität der betriebssystemabhängigen Darstellung ermittelt werden kann.

Bei Verwendung der betriebssystemabhängigen Darstellung ist das Resultat größer 0.

**Beispiel:**

```
// Betriebssystem-Darstellung aktivif (WinInfo(0, _WinThemeActive) > 0){ ...}// Betriebssystem
```



\_WinType

Objekt-Typ ermitteln

Wert 1

Verwandte

Siehe Befehle,

WinInfo()

Option bei WinInfo() durch die der Typ eines Objekts ermittelt werden kann.

Folgende Objekt-Typen können zurückgegeben werden:

- \_WinTypeAnimation
- \_WinTypeAppFrame
- \_WinTypeApplication
- \_WinTypeBarcode
- \_WinTypeBigIntEdit
- \_WinTypeButton
- \_WinTypeC16Info
- \_WinTypeCalendar
- \_WinTypeCanvas
- \_WinTypeCanvasGraphic
- \_WinTypeCheckBox
- \_WinTypeChromium
- \_WinTypeCodeEdit
- \_WinTypeCodeEditList
- \_WinTypeCodeEditListColumn
- \_WinTypeColorButton
- \_WinTypeColorEdit
- \_WinTypeComArguments
- \_WinTypeComFileOpen
- \_WinTypeComFileSave
- \_WinTypeComFont
- \_WinTypeComOdbc
- \_WinTypeComPath
- \_WinTypeComPrint
- \_WinTypeComPrintSetup
- \_WinTypeCtxAdobeReader
- \_WinTypeCtxDocEdit
- \_WinTypeCtxDocEditRuler
- \_WinTypeCtxDocEditTBar
- \_WinTypeCtxOffice
- \_WinTypeDataList
- \_WinTypeDataListPopup
- \_WinTypeDateEdit
- \_WinTypeDecimalEdit
- \_WinTypeDialog
- \_WinTypeDivider
- \_WinTypeDocView
- \_WinTypeDragDataFormat
- \_WinTypeDragDataObject
- \_WinTypeEdit
- \_WinTypeFloatEdit

- WinTypeFontNameEdit
- WinTypeFontSizeEdit
- WinTypeFrame
- WinTypeFrameClient
- WinTypeGanttAxis
- WinTypeGanttGraph
- WinTypeGanttView
- WinTypeGroup
- WinTypeGroupBox
- WinTypeGroupColumn
- WinTypeGroupSplit
- WinTypeGroupTile
- WinTypeGroupTileButton
- WinTypeHelpTip
- WinTypeHyperLink
- WinTypeIcon
- WinTypeIntEdit
- WinTypeInterval
- WinTypeIvlBox
- WinTypeIvlLine
- WinTypeLabel
- WinTypeListColumn
- WinTypeMdiFrame
- WinTypeMenu
- WinTypeMenuButton
- WinTypeMenuItem
- WinTypeMetaPicture
- WinTypeNotebook
- WinTypeNotebookPage
- WinTypePicture
- WinTypePopupColor
- WinTypePopupList
- WinTypePpcObject
- WinTypePrintDevice
- WinTypePrinter
- WinTypePrinterList
- WinTypePrintJob
- WinTypeProgress
- WinTypePrtJobPreview
- WinTypePrtPpvControl
- WinTypePrtPreviewDlg
- WinTypeRadioButton
- WinTypeRecList
- WinTypeRecListPopup
- WinTypeRecNavigator
- WinTypeRecView
- WinTypeRTFEdit
- WinTypeScrollbar
- WinTypeScrollbar
- WinTypeSelDataObject
- WinTypeStatusbar

- WinTypeStatusBarButton
- WinTypeStoList
- WinTypeStoListPopup
- WinTypeTextEdit
- WinTypeTimeEdit
- WinTypeToolbar
- WinTypeToolbarButton
- WinTypeToolbarDock
- WinTypeToolbarMenu
- WinTypeToolbarRTF
- WinTypeTrayFrame
- WinTypeTreeNode
- WinTypeTreeView
- WinTypeWebNavigator
- WinTypeWindowbar

WinUnicode  
Unicode-Unterstützung des Objektes ermitteln  
Wert 19

Verwandte

Siehe Befehle,  
WinInfo()

Option bei WinInfo() durch die die Unicode-Unterstützung des Objektes ermittelt werden kann. Bei einem Resultat von 1 besitzt das Objekt Unicode-Eigenschaften. Welche Eigenschaften dies betrifft, kann der Seite Unicode-Unterstützung entnommen werden.

Objekttypen

Objekttypen

Siehe Objektinformationsbefehle

- WinTypeAnimation
- WinTypeAppFrame
- WinTypeApplication
- WinTypeBarcode
- WinTypeBigIntEdit
- WinTypeButton
- WinTypeC16Info
- WinTypeCalendar
- WinTypeCanvas
- WinTypeCanvasGraphic
- WinTypeCheckBox
- WinTypeChromium
- WinTypeCodeEdit
- WinTypeCodeEditList
- WinTypeCodeEditListColumn
- WinTypeColorButton
- WinTypeColorEdit
- WinTypeComArguments
- WinTypeComFileOpen
- WinTypeComFileSave
- WinTypeComFont
- WinTypeComOdbc
- WinTypeComPath
- WinTypeComPrint
- WinTypeComPrintSetup
- WinTypeCtxAdobeReader
- WinTypeCtxDocEdit
- WinTypeCtxDocEditRuler
- WinTypeCtxDocEditTBar
- WinTypeCtxOffice
- WinTypeDataList
- WinTypeDataListPopup
- WinTypeDateEdit
- WinTypeDecimalEdit
- WinTypeDialog
- WinTypeDivider
- WinTypeDocView
- WinTypeDragDataFormat
- WinTypeDragDataObject
- WinTypeEdit
- WinTypeFloatEdit
- WinTypeFontNameEdit
- WinTypeFontSizeEdit
- WinTypeFrame
- WinTypeFrameClient
- WinTypeGanttAxis
- WinTypeGanttGraph

- WinTypeGanttView
- WinTypeGroup
- WinTypeGroupBox
- WinTypeGroupColumn
- WinTypeGroupSplit
- WinTypeGroupTile
- WinTypeGroupTileButton
- WinTypeHelpTip
- WinTypeHyperLink
- WinTypeIcon
- WinTypeIntEdit
- WinTypeInterval
- WinTypeIvlBox
- WinTypeIvlLine
- WinTypeLabel
- WinTypeListColumn
- WinTypeMdiFrame
- WinTypeMenu
- WinTypeMenuButton
- WinTypeMenuItem
- WinTypeMetaPicture
- WinTypeNotebook
- WinTypeNotebookPage
- WinTypePicture
- WinTypePopupColor
- WinTypePopupList
- WinTypePpcObject
- WinTypePrintDevice
- WinTypePrinter
- WinTypePrinterList
- WinTypePrintJob
- WinTypeProgress
- WinTypePrtJobPreview
- WinTypePrtPpvControl
- WinTypePrtPreviewDlg
- WinTypeRadioButton
- WinTypeRecList
- WinTypeRecListPopup
- WinTypeRecNavigator
- WinTypeRecView
- WinTypeRTFEdit
- WinTypeScrollbar
- WinTypeScrollbox
- WinTypeSelDataObject
- WinTypeStatusbar
- WinTypeStatusbarButton
- WinTypeStoList
- WinTypeStoListPopup
- WinTypeTextEdit
- WinTypeTimeEdit
- WinTypeToolbar

- WinTypeToolBarButton
- WinTypeToolBarDock
- WinTypeToolBarMenu
- WinTypeToolBarRTF
- WinTypeTrayFrame
- WinTypeTreeNode
- WinTypeTreeView
- WinTypeWebNavigator
- WinTypeWindowbar

\_WinTypeAnimation  
Animations-Objekt

Wert 61.865.986 /  
0x03B00002

Verwandte

Siehe Befehle,  
WinInfo(),  
Animation

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Animation-Objekt  
hinweist.



\_WinTypeAppFrame  
Applikations-Fenster-Objekt  
Wert 51.380.225 /  
0x03100001

Verwandte

Siehe Befehle,  
WinInfo(),  
AppFrame

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein AppFrame-Objekt  
hinweist.

\_WinTypeApplication  
Applikations-Objekt

Wert 2 /  
0x00000002

Verwandte

Siehe Befehle,  
WinInfo(),  
Application

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Application-Objekt  
hinweist.

\_WinTypeBigIntEdit  
BigInt-Eingabe-Objekt  
Wert 54.525.964 /  
0x0340000C

Verwandte

Siehe Befehle,  
WinInfo(),  
BigIntEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein BigIntEdit-Objekt  
hinweist.

\_WinTypeButton  
Schaltflächen-Objekt  
Wert 53.477.377 /  
0x03300001

Verwandte

Siehe Befehle,  
WinInfo(),  
Button

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Button-Objekt  
hinweist.

\_WinTypeC16Info  
CONZEPT 16-Dialog "Info"

Wert 65.011.760 /  
0x03E00030

Verwandte

Befehle,

Siehe WinInfo(),

WinOpen(\_WinC16Info,  
...)

Rückgabewert von WinInfo() mit der Option \_WinType der auf den Info-Dialog von CONZEPT 16 hinweist.

\_WinTypeCalendar

Kalender-Objekt

Wert 61.865.987 /  
0x03B00003

Verwandte

Siehe Befehle,  
WinInfo(),  
Calendar

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Calendar-Objekt  
hinweist.

\_WinTypeCanvas

Canvas-Objekt

Wert 56.623.105 /  
0x03600001

Verwandte

Siehe Befehle,  
WinInfo(),  
Canvas

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Canvas-Objekt  
hinweist.

\_WinTypeCanvasGraphic  
CanvasGraphic-Objekt

Wert 56.688.640 /  
0x03610000

Verwandte

Siehe Befehle,  
WinInfo(),

CanvasGraphic

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
CanvasGraphic-Objekt hinweist.



WinTypeCheckBox  
Mehrfachauswahlnopf-Objekt  
Wert 53.477.378 /  
0x03300002

Verwandte

Siehe Befehle,  
WinInfo(),  
Checkbox

Rückgabewert von WinInfo() mit der Option WinType der auf ein Checkbox-Objekt  
hinweist.

\_WinTypeCodeEdit

CodeEdit-Objekt

Wert 54.528.013 / 0x0340080D

Verwandte Befehle, WinInfo(),

Siehe \_WinTypeCodeEditList,  
\_WinTypeCodeEditListColumn,  
CodeEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein CodeEdit-Objekt  
hinweist.

\_WinTypeCodeEditList  
Autovervollständigungsliste des CodeEdit-Objektes  
Wert 54.528.016 / 0x03400810

Verwandte Befehle,

Siehe \_WinTypeCodeEdit,  
\_WinTypeCodeEditListColumn,  
CodeEdit

Auswahl der Autovervollständigungsliste eines CodeEdit-Objektes zum Setzen von Theme-Eigenschaften.

\_WinTypeCodeEditListColumn

Spalte der Autovervollständigungsliste des CodeEdit-Objektes

Wert 54.528.017 /

0x03400811

Verwandte Befehle,

Siehe \_WinTypeCodeEdit,

\_WinTypeCodeEditList,

CodeEdit

Auswahl einer Spalte der Autovervollständigungsliste eines CodeEdit-Objektes zum Setzen von Theme-Eigenschaften.

\_WinTypeColorButton  
Farbschaltflächen-Objekt

Wert 53.477.384 /  
0x03300008

Verwandte

Siehe Befehle,  
WinInfo(),  
ColorButton

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein ColorButton-Objekt  
hinweist.

\_WinTypeColorEdit  
Farbeingabe-Objekt

Wert 54.525.959 /  
0x03400007

Verwandte

Siehe Befehle,  
WinInfo(),  
ColorEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein ColorEdit-Objekt  
hinweist.

\_WinTypeComArguments  
Argument-Objekt der COM-Schnittstelle  
Wert 134.217.729 /  
0x08000001

Verwandte

Siehe Befehle,  
WinInfo(),  
EvtCtxEvent

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Argument-Objekt der COM-Schnittstelle hinweist. Diese Objekt wird als Argument aComArguments beim Ereignis EvtCtxEvent übergeben.

\_WinTypeComFileOpen  
Common-Dialog (\_WinComFileOpen)  
Wert 65.011.728 /  
0x03E00010

Verwandte Befehle,

Siehe WinInfo(),

\_WinComFileOpen

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Common-Dialog vom Typ \_WinComFileOpen hinweist.



\_WinTypeComFileSave  
Common-Dialog ( \_WinComFileSave)  
Wert 65.011.729 /  
0x03E00011

Verwandte

Siehe Befehle, WinInfo(),  
\_WinComFileSave

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Common-Dialog vom Typ \_WinComFileSave hinweist.

\_WinTypeComFont  
Common-Dialog ( \_WinComFont)  
Wert 65.011.733 /  
0x03E00015

Verwandte

Siehe Befehle,  
WinInfo(),  
\_WinComFont

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Common-Dialog vom Typ \_WinComFont hinweist.

\_WinTypeComOdbc  
Common-Dialog ( \_WinComOdbc)  
Wert 65.011.734 /  
0x03E00016

Verwandte

Siehe Befehle,  
WinInfo(),  
\_WinComOdbc

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Common-Dialog vom Typ \_WinComOdbc hinweist.

\_WinTypeComPath  
Common-Dialog (\_WinComPath)  
Wert 65.011.730 /  
0x03E00012

Verwandte

Siehe Befehle,  
WinInfo(),  
\_WinComPath

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Common-Dialog vom Typ \_WinComPath hinweist.

\_WinTypeComPrint  
Common-Dialog (\_WinComPrint)  
Wert 65.011.731 /  
0x03E00013

Verwandte

Siehe Befehle,  
WinInfo(),  
\_WinComPrint

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Common-Dialog vom Typ \_WinComPrint hinweist.

\_WinTypeComPrintSetup  
Common-Dialog ( \_WinComPrintSetup)  
Wert 65.011.732 /  
0x03E00014

Verwandte Befehle,

Siehe WinInfo(),

\_WinComPrintSetup

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Common-Dialog vom Typ \_WinComPrintSetup hinweist.

\_WinTypeCtxAdobeReader  
Kontroll-Objekt des Adobe Readers  
Wert 59.768.850 /  
0x03900012

Verwandte

Siehe Befehle,  
WinInfo(),  
CtxAdobeReader

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein CtxAdobeReader-Objekt hinweist.

\_WinTypeCtxDocEdit

CtxDocEdit-Objekt

Wert 59.768.855 /  
0x03900017

Verwandte

Siehe Befehle,  
WinInfo(),  
CtxDocEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein CtxDocEdit-Objekt  
hinweist.



WinTypeCtxtDocEditRuler  
Lineal eines CtxtDocEdit-Objektes

Wert 59.768.854 /  
0x03900016

Verwandte Befehle,

WinInfo(),

Siehe CtxtDocEdit,

WinToolbarRulerH,

WinToolbarRulerV

Rückgabewert von WinInfo() mit der Option WinType der auf ein Lineal eines CtxtDocEdit-Objektes hinweist.

\_WinTypeCtxtDocEditTBar  
Toolbar eines CtxtDocEdit-Objektes

Wert 59.768.857 /  
0x03900019

Verwandte Befehle,

Siehe WinInfo(),  
CtxtDocEdit,  
\_WinToolbarButtons

Rückgabewert von WinInfo() mit der Option \_WinType der auf eine Toolbar eines CtxtDocEdit-Objektes hinweist.

\_WinTypeCtxOffice  
Kontroll-Objekt für Microsoft Office Dokumente  
Wert 59.768.849 /  
0x03900011

Verwandte  
Siehe Befehle,  
WinInfo(),  
CtxOffice

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein CtxOffice-Objekt  
hinweist.

\_WinTypeDataList

DataList-Objekt

Wert 57.671.937 /  
0x03700101

Verwandte

Siehe Befehle,  
WinInfo(),  
DataList

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein DataList-Objekt  
hinweist.

\_WinTypeDataListPopup  
DataListPopup-Objekt

Wert 57.672.193 /  
0x03700201

Verwandte

Siehe Befehle,  
WinInfo(),  
DataListPopup

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
DataListPopup-Objekt hinweist.

\_WinTypeDateEdit  
Datumseingabe-Objekt  
Wert 54.525.957 /  
0x03400005

Verwandte

Siehe Befehle,  
WinInfo(),  
DateEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein DateEdit-Objekt  
hinweist.

\_WinTypeDecimalEdit  
Decimal-Eingabe-Objekt

Wert 54.525.965 /  
0x0340000D

Verwandte

Siehe Befehle,  
WinInfo(),  
DecimalEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein DecimalEdit-Objekt  
hinweist.

\_WinTypeDialog

Dialog-Objekt

Wert 50.331.652 /  
0x03000004

Verwandte

Siehe Befehle,  
WinInfo(),  
Frame

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Frame-Objekt  
hinweist.



WinTypeDivider  
Trennlinien-Objekt  
Wert 52.428.806 /  
0x03200006

Verwandte

Siehe Befehle,  
WinInfo(),  
Divider

Rückgabewert von WinInfo() mit der Option WinType der auf ein Divider-Objekt  
hinweist.

\_WinTypeDocView  
Dokument-Objekt  
Wert 61.866.004 /  
0x03B00014

Verwandte

Siehe Befehle,  
WinInfo(),  
DocView

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein DocView-Objekt  
hinweist.

\_WinTypeDragDataFormat

Datenformat-Objekt

Wert 100.663.298 /  
0x06000002

Verwandte

Siehe Befehle,

WinInfo(),

DragDataFormat

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein DragDataFormat-Objekt hinweist.

\_WinTypeDragDataObject  
Daten einer Drag & Drop-Operation

Wert 100.663.297  
/ 0x06000001

Verwandte

Siehe Befehle,  
WinInfo(),  
DragData

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein DragData-Objekt  
hinweist.

\_WinTypeEdit  
Alpha-Eingabe-Objekt  
Wert 54.525.953 /  
0x03400001

Verwandte

Siehe Befehle,  
WinInfo(),  
Edit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Edit-Objekt  
hinweist.

WinTypeFloatEdit  
Float-Eingabe-Objekt  
Wert 54.525.955 /  
0x03400003

Verwandte

Siehe Befehle,  
WinInfo(),  
FloatEdit

Rückgabewert von WinInfo() mit der Option WinType der auf ein FloatEdit-Objekt  
hinweist.

\_WinTypeFontNameEdit  
Schriftnameneingabe-Objekt

Wert 62.914.576 /  
0x03C00010

Verwandte

Siehe Befehle,  
WinInfo(),  
FontNameEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein FontNameEdit-Objekt hinweist.

\_WinTypeFontSizeEdit  
Schriftgrößeneingabe-Objekt  
Wert 62.914.577 /  
0x03C00011

Verwandte

Siehe Befehle,  
WinInfo(),  
FontSizeEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein FontSizeEdit-Objekt  
hinweist.



\_WinTypeFrame

Frame-Objekt

Wert 50.331.651 /  
0x03000003

Verwandte

Siehe Befehle,  
WinInfo(),  
Frame

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Frame-Objekt hinweist.



Da für die Anzeige zur Laufzeit alle Frames in den Dialogtyp konvertiert werden, ist der Rückgabewert von WinInfo() auf Fenster-Objekte immer \_WinTypeDialog.

\_WinTypeFrameClient  
FrameClient-Objekt

Wert 60.817.424 /  
0x03A00010

Verwandte

Siehe Befehle,  
WinInfo(),  
FrameClient

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein FrameClient-Objekt  
hinweist.

\_WinTypeGanttAxis

Achsen-Objekt

Wert 16.777.220 /  
0x01000004

Verwandte

Befehle,

Siehe WinInfo(),

Axis,

GanttGraph

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Axis-Objekt eines GanttGraph-Objektes hinweist.

\_WinTypeGanttGraph  
Gantt-Diagramm-Objekt

Wert 61.866.000 /  
0x03B00010

Verwandte

Siehe Befehle,  
WinInfo(),  
GanttGraph

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein GanttGraph-Objekt  
hinweist.

WinTypeGanttView  
View-Objekt

Wert 61.866.001 /  
0x03B00011

Verwandte

Befehle,

Siehe WinInfo(),

View,

GanttGraph

Rückgabewert von WinInfo() mit der Option WinType der auf ein View-Objekt eines GanttGraph-Objektes hinweist.

\_WinTypeGroup

RecView-Gruppe

Wert 61.866.012 /  
0x03B0001C

Verwandte

Befehle,

Siehe WinInfo(),

Group

RecView

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Group-Objekt eines RecView-Objektes hinweist.

\_WinTypeGroupBox  
Gruppen-Objekt

Wert 52.428.803 /  
0x03200003

Verwandte

Siehe Befehle,  
WinInfo(),  
GroupBox

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein GroupBox-Objekt  
hinweist.

\_WinTypeGroupColumn

Spalten-Objekt

Wert 61.866.008 /  
0x03B00018

Verwandte

Befehle,

Siehe WinInfo(),

Column

RecView

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Column-Objekt eines RecView-Objektes hinweist.



\_WinTypeGroupSplit  
GroupSplit-Objekt  
Wert 52.428.807 /  
0x03200007

Verwandte

Siehe Befehle,  
WinInfo(),  
GroupSplit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein GroupSplit-Objekt  
hinweist.

\_WinTypeGroupTile  
GroupTile-Objekt  
Wert 52.428.808 /  
0x03200008

Verwandte

Siehe Befehle,  
WinInfo(),  
GroupTile

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein GroupTile-Objekt  
hinweist.

\_WinTypeGroupTileButton  
GroupTile-Button-Objekt

Wert 60.817.676 /  
0x03A0010C

Verwandte

Siehe Befehle, WinInfo(),  
GroupTile-Button

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
GroupTile-Button-Objekt hinweist.

\_WinTypeHyperLink  
HyperLink-Objekt  
Wert 52.428.802 /  
0x03200002

Verwandte

Siehe Befehle,  
WinInfo(),  
HyperLink

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein HyperLink-Objekt  
hinweist.

WinTypeIcon

Symbol-Objekt

Wert 61.865.989 /  
0x03B00005

Verwandte

Siehe Befehle,  
WinInfo(),

Icon

Rückgabewert von WinInfo() mit der Option WinType der auf ein Icon-Objekt  
hinweist.

\_WinTypeIntEdit  
Int-Eingabe-Objekt  
Wert 54.525.954 /  
0x03400002

Verwandte

Siehe Befehle,  
WinInfo(),  
IntEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein IntEdit-Objekt  
hinweist.

\_WinTypeInterval

Intervall-Objekt

Wert 16.777.221 /  
0x01000005

Verwandte

Befehle,

Siehe WinInfo(),

Interval,

GanttGraph

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Interval-Objekt eines GanttGraph-Objektes hinweist.

\_WinTypeIvlBox

Box-Objekt

Wert 16.777.224 /  
0x01000008

Verwandte

Befehle,

Siehe WinInfo(),

Box,

GanttGraph

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Box-Objekt eines GanttGraph-Objektes hinweist.



WinTypeIvlLine

Linien-Objekt

Wert 16.777.225 /  
0x01000009

Verwandte

Befehle,

Siehe WinInfo(),

Line,

GanttGraph

Rückgabewert von WinInfo() mit der Option WinType der auf ein Line-Objekt eines GanttGraph-Objektes hinweist.

\_WinTypeLabel  
Bezeichner-Objekt  
Wert 52.428.801 /  
0x03200001

Verwandte

Siehe Befehle,  
WinInfo(),  
Label

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Label-Objekt  
hinweist.

\_WinTypeListColumn

Spalten-Objekt

Wert 57.671.688 /  
0x03700008

Verwandte

Befehle,

WinInfo(),

Siehe Column,

DataList,

RecList,

StoList

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Column-Objekt eines DataList-, RecList - oder StoList-Objektes hinweist.

\_WinTypeMdiFrame

MDI-Fenster-Objekt

Wert 51.380.227 /  
0x03100003

Verwandte

Siehe Befehle,  
WinInfo(),  
MdiFrame

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein MdiFrame-Objekt  
hinweist.

WinTypeMenu

Menü-Objekt

Wert 33.554.434 /  
0x02000002

Verwandte

Siehe Befehle,

WinInfo()

Rückgabewert von WinInfo() mit der Option WinType der auf ein Menü-Objekt hinweist.

\_WinTypeMenuButton  
MenuButton-Objekt

Wert 53.477.387 /  
0x0330000B

Verwandte

Siehe Befehle,  
WinInfo(),  
MenuButton

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein MenuButton-Objekt  
hinweist.

WinTypeMenuItem  
Menüeintrag-Objekt

Wert 33.554.435 /  
0x02000003

Verwandte

Siehe Befehle,  
WinInfo(),  
MenuItem

Rückgabewert von WinInfo() mit der Option WinType der auf ein MenuItem-Objekt eines Menü-Objektes hinweist.

\_WinTypeMetaPicture  
Meta-Grafik-Objekt  
Wert 61.866.002 /  
0x03B00012

Verwandte

Siehe Befehle,  
WinInfo(),  
MetaPicture

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein MetaPicture-Objekt hinweist.



\_WinTypeNotebook  
Notizbuch-Objekt  
Wert 55.574.529 /  
0x03500001

Verwandte

Siehe Befehle,  
WinInfo(),  
Notebook

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Notebook-Objekt  
hinweist.

\_WinTypeNotebookPage  
Notizbuchseiten-Objekt

Wert 55.574.530 /  
0x03500002

Verwandte

Befehle,

Siehe WinInfo(),  
NotebookPage,  
Notebook

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
NotebookPage-Objekt eines Notebook-Objektes hinweist.

\_WinTypePicture

Grafik-Objekt

Wert 16.777.218 /  
0x01000002

Verwandte

Siehe Befehle,  
WinInfo(),  
Picture

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Picture-Objekt  
hinweist.

\_WinTypePopupColor

Popup eines ColorButton- oder ColorEdit-Objekts

Wert 63.963.139 /

0x03d00003

Verwandte

Siehe Befehle,

WinInfo()

Rückgabewert von WinInfo() mit der Option \_WinType der auf das PopupColor-Objekt eines ColorButton- oder ColorEdit-Objekts hinweist.

\_WinTypePopupList

PopupList-Objekt

Wert 63.963.137 /

0x03D00001

Verwandte

Siehe Befehle,

WinInfo(),

PopupList

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein PopupList-Objekt eines Eingabe-Objekts hinweist.

\_WinTypePpcObject  
Druckprozessor-Objekt  
Wert 83.886.082 /  
0x05000002

Verwandte

Siehe Befehle,  
WinInfo(),  
ppcObject

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
Druckprozessor-Objekt hinweist.

\_WinTypePrintDevice

PrintDevice-Objekt

Wert 67.108.866 /  
0x04000002

Verwandte

Siehe Befehle,  
WinInfo(),  
PrintDevice

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein PrintDevice-Objekt  
hinweist.

\_WinTypePrinter

Printer-Objekt

Wert 67.109.122 /  
0x04000102

Verwandte

Siehe Befehle,  
WinInfo(),  
Printer

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Printer-Objekt  
hinweist.



\_WinTypePrinterList  
PrinterList-Objekt

Wert 67.109.121 /  
0x04000101

Verwandte

Siehe Befehle,  
WinInfo(),  
PrinterList

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein PrinterList-Objekt  
hinweist.

\_WinTypePrintJob

PrintJob-Objekt

Wert 67.108.865 /  
0x04000001

Verwandte

Siehe Befehle,  
WinInfo(),  
PrintJob

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein PrintJob-Objekt  
hinweist.

WinTypeProgress  
Fortschrittsanzeigen-Objekt  
Wert 61.865.985 /  
0x03B00001

Verwandte  
Siehe Befehle,  
WinInfo(),  
Progress

Rückgabewert von WinInfo() mit der Option WinType der auf ein Progress-Objekt  
hinweist.

\_WinTypePrtJobPreview  
Druckvorschau-Objekt

Wert 16.777.260 /  
0x0100002C

Verwandte

Siehe Befehle,  
WinInfo(),  
PrtJobPreview

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
PrtJobPreview-Objekt hinweist.

\_WinTypePrtPreviewDlg  
Druck-Vorschau-Container-Objekt  
Wert 16.777.264 /  
0x01000030

Verwandte

Siehe Befehle,  
WinInfo(),  
PrtFrame

Rückgabewert von WinInfo() mit der Option \_WinType der auf das Control-Objekt der Druckvorschau hinweist.

\_WinTypePrtPreviewDlg  
Druckvorschau-Dialog

Wert 65.011.745 /  
0x03E00021

Verwandte

Siehe Befehle,  
WinInfo(),  
PrtFrame

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
Druckvorschau-Dialog hinweist. Dies dient zur Unterscheidung zwischen dem  
Vorschau-Objekt und dem Vorschau-Dialog.

WinTypeRadioButton  
Einfachauswahlnopf-Objekt  
Wert 53.477.379 /  
0x03300003

Verwandte

Siehe Befehle,  
WinInfo(),  
Radiobutton

Rückgabewert von WinInfo() mit der Option WinType der auf ein Radiobutton-Objekt  
hinweist.

\_WinTypeRecList  
RecList-Objekt

Wert 57.671.690 /  
0x0370000A

Verwandte

Siehe Befehle,  
WinInfo(),  
RecList

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein RecList-Objekt  
hinweist.



\_WinTypeRecListPopup  
RecListPopup-Objekt

Wert 57.672.192 /  
0x03700200

Verwandte

Siehe Befehle,  
WinInfo(),  
RecListPopup

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
RecListPopup-Objekt hinweist.

\_WinTypeRecNavigator  
Datensatznavigations-Objekt

Wert 59.768.834 /  
0x03900002

Verwandte

Siehe Befehle,  
WinInfo(),  
RecNavigator

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
RecNavigator-Objekt hinweist.

\_WinTypeRecView  
RecView-Objekt  
Wert 61.866.010 /  
0x03B0001A

Verwandte

Siehe Befehle,  
WinInfo(),  
RecView

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein RecView-Objekt  
hinweist.

\_WinTypeRTFEdit  
RTF-Eingabe-Objekt  
Wert 54.525.968 /  
0x03400010

Verwandte

Siehe Befehle,  
WinInfo(),  
RtfEdit

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein RtfEdit-Objekt  
hinweist.

\_WinTypeScrollbar  
Scrollbar-Objekt

Wert 52.428.813 /  
0x0320000D

Verwandte

Siehe Befehle,

WinInfo()

Rückgabewert von WinInfo() mit der Option \_WinType der auf eine Scrollbar hinweist.

\_WinTypeScrollbox  
Scrollbox-Objekt  
Wert 52.428.811 /  
0x0320000B

Verwandte  
Siehe Befehle,  
WinInfo(),  
Scrollbox

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Scrollbox-Objekt  
hinweist.

\_WinTypeSelDataObject  
Selektions-Objekt der Mehrfachauswahl

Wert 117.440.513 /  
0x07000001

Verwandte

Siehe Befehle,  
WinInfo(),  
SelectionData

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein  
SelectionData-Objekt hinweist.

\_WinType Statusbar  
Statusleisten-Objekt

Wert 60.817.668 /  
0x03A00104

Verwandte

Siehe Befehle,  
WinInfo(),  
Statusbar

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Statusbar-Objekt  
hinweist.



\_WinTypeStatusBarButton  
Statusleisten-Objekt

Wert 60.817.669 /  
0x03A00105

Verwandte

Befehle,

Siehe WinInfo(),

Statusbar-Button,

Statusbar

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Statusbar-Button-Objekt eines Statusbar-Objektes hinweist.

\_WinTypeStoList

StoList-Objekt

Wert 57.672.705 /  
0x03700401

Verwandte

Siehe Befehle,  
WinInfo(),  
StoList

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein StoList-Objekt  
hinweist.

\_WinTypeStoListPopup  
StoListPopup-Objekt

Wert 57.672.706 /  
0x03700402

Verwandte

Siehe Befehle,  
WinInfo(),  
StoListPopup

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein StoListPopup-Objekt hinweist.

WinTypeTextEdit  
Texteingabe-Objekt  
Wert 54.525.961 /  
0x03400009

Verwandte

Siehe Befehle,  
WinInfo(),  
TextEdit

Rückgabewert von WinInfo() mit der Option WinType der auf ein TextEdit-Objekt  
hinweist.

WinTypeTimeEdit  
Zeiteingabe-Objekt  
Wert 54.525.956 /  
0x03400004

Verwandte

Siehe Befehle,  
WinInfo(),  
TimeEdit

Rückgabewert von WinInfo() mit der Option WinType der auf ein TimeEdit-Objekt  
hinweist.

\_WinTypeToolbar  
Werkzeuggesteigen-Objekt  
Wert 60.817.666 /  
0x03A00102

Verwandte  
Siehe Befehle,  
WinInfo(),  
Toolbar

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Toolbar-Objekt  
hinweist.

WinTypeToolbarButton  
Werkzeugleisten-Schaltflächen-Objekt  
Wert 60.817.667 /  
0x03A00103

Verwandte

Befehle,

Siehe WinInfo(),  
Toolbar-Button,  
Toolbar

Rückgabewert von WinInfo() mit der Option WinType der auf ein Toolbar-Button-Objekt eines Toolbar-Objektes hinweist.

\_WinTypeToolbarDock  
Werkzeigleisten-Dock-Objekt  
Wert 60.817.665 /  
0x03A00101

Verwandte

Befehle,

Siehe WinInfo(),  
ToolbarDock,  
Toolbar

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein ToolbarDock-Objekt für ein Toolbar-Objekt hinweist.



\_WinTypeToolBarMenu  
ToolBar-Menü-Objekt

Wert 60.817.677 /  
0x03A0010D

Verwandte

Siehe Befehle,  
WinInfo()

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein ToolBarMenu-Objekt hinweist.

\_WinTypeToolbarRTF  
RTF-Werkzeugleisten-Objekt  
Wert 60.817.674 /  
0x03A0010A

Verwandte

Siehe Befehle,  
WinInfo(),  
ToolbarRtf

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein ToolbarRtf-Objekt  
hinweist.

\_WinTypeTrayFrame  
Tray-Fenster-Objekt  
Wert 50.331.661 /  
0x0300000D

Verwandte

Siehe Befehle,  
WinInfo(),  
TrayFrame

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein TrayFrame-Objekt  
hinweist.

\_WinTypeTreeNode

Knoten-Objekt

Wert 58.720.261 /  
0x03800005

Verwandte

Befehle,

Siehe WinInfo(),

TreeNode,

TreeView

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein TreeNode-Objekt eines TreeView-Objektes hinweist.

\_WinTypeTreeView

Baum-Objekt

Wert 58.720.259 /  
0x03800003

Verwandte

Siehe Befehle,  
WinInfo(),  
TreeView

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein TreeView-Objekt  
hinweist.

\_WinTypeWebNavigator

Dateianzeige-Objekt

Wert 59.768.835 /  
0x03900003

Verwandte

Siehe Befehle,

WinInfo(),

WebNavigator

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein WebcNavigator-Objekt hinweist.

\_WinTypeWindowbar  
Windowbar-Objekt

Wert 60.817.673 /  
0x03A00109

Verwandte

Siehe Befehle,  
WinInfo(),  
Windowbar

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Windowbar-Objekt  
hinweist.

\_WinTypeChromium

Chromium-Objekt

Wert 62.914.581 /  
0x03C00015

Verwandte

Siehe Befehle,  
WinInfo(),  
Chromium

Rückgabewert von WinInfo() mit der Option \_WinType der auf ein Chromium-Objekt  
hinweist.



Objektstatus

Objektstatus

Siehe Objektinformationsbefehle

- WinDialogMaximized
- WinDialogMinimized
- WinDialogNormal
- WinStateAttachClosed
- WinStateAttachMaximized
- WinStateAttachNormal

\_WinDialogMaximized  
Fenster wird maximiert dargestellt  
Wert 16 / 0x00000010

Verwandte  
Befehle,  
WinDialog(),  
Siehe WinDialogRun(),  
WinAdd(),  
WinAddByName(),  
WinUpdate(),  
WinInfo()

Je nach Befehl hat \_WinDialogMaximized folgende Bedeutung:

- WinDialog(), WinDialogRun(), WinAdd() und WinAddByName()

Dialog wird maximiert geladen.

- WinUpdate()

Dialog wird zur Laufzeit maximiert. Die Option kann nur angegeben werden, wenn \_WinUpdState als ersten Parameter übergeben wird.

- WinInfo()

Liefert WinInfo(\_WinState) den Wert \_WinDialogMaximized zurück, ist der Dialog maximiert.

\_WinDialogMinimized  
Fenster wird minimiert dargestellt  
Wert 32 / 0x00000020

Verwandte  
Befehle,  
WinDialog(),  
Siehe WinDialogRun(),  
WinAdd(),  
WinAddByName(),  
WinUpdate(),  
WinInfo()

Je nach Befehl hat \_WinDialogMinimized folgende Bedeutung:

- WinDialog(), WinDialogRun(), WinAdd() und WinAddByName()

Dialog wird minimiert geladen.

- WinUpdate()

Dialog wird zur Laufzeit minimiert. Die Option kann nur angegeben werden, wenn \_WinUpdState als ersten Parameter übergeben wird.

- WinInfo()

Liefert WinInfo(\_WinState) den Wert \_WinDialogMinimized zurück ist der Dialog minimiert.

\_WinDialogNormal  
Fenster weder maximiert noch minimiert

Wert 64 /  
0x00000040

Siehe WinUpdate(),  
WinInfo()

Je nach Befehl hat \_WinDialogNormal folgende Bedeutung:

- WinDialog() und WinDialogRun()

Der Dialog wird weder minimiert, noch maximiert angezeigt. Ist durch die Darstellung das Fenster in keinem Bildschirm sichtbar, wird es im primären Bildschirm zentriert dargestellt.

- WinUpdate()

Dialog wird zur Laufzeit von einem minimierten oder maximierten Status zurückgesetzt. Die Option kann nur angegeben werden, wenn \_WinUpdState als ersten Parameter übergeben wird.

- WinInfo()

Liefert WinInfo(\_WinState) den Wert \_WinDialogNormal zurück ist der Dialog weder maximiert noch minimiert.

\_WinStateAttachClosed

Objekt ist geschlossen

Wert 3

Siehe EvtAttachState,

\_WinState

Das GroupTile-Objekt ist/wird geschlossen.

\_WinStateAttachMaximized

Maximierte Darstellung

Wert 2

Siehe EvtAttachState,

\_WinState

Das GroupTile-Objekt wird maximiert dargestellt.

\_WinStateAttachNormal

Normale Darstellung

Wert 1

Siehe EvtAttachState,  
\_WinState

Das GroupTile-Objekt ist/wird normal dargestellt.

Oberflächenobjektbefehle

Befehle für Oberflächenobjekte

Liste sortiert

nach

Siehe Gruppen,  
Alphabetische  
Liste aller  
Befehle

Befehle für Objekte:

- CtxDocEdit-Befehle
- DataList-Befehle
- GanttGraph-Befehle
- Menu-Befehle
- RtfEdit-Befehle
- Theme-Befehle
- TreeView-Befehle

Befehle zum Laden/Anzeigen von Dialogen:

- WinAdd
- WinAddByName
- WinClose
- WinDialog
- WinDialogBox
- WinDialogResult
- WinDialogRun
- WinOpen
- WinSave
- WinUrmDialog

Befehle zum Setzen/Ermitteln des Focus:

- WinFocusGet
- WinFocusSet

Befehle für Ereignisse:

- ComEvtProcessGet
- ComEvtProcessSet
- WinEvtProcessGet
- WinEvtProcessSet
- WinEvtProcNameGet
- WinEvtProcNameSet

Sonstige Befehle:

- ColorMake
- FontMake
- PointMake
- RangeMake



- RectMake
- RtfTabMake
- WinBarcodeSaveImage
- WinBeep
- WinBoxScrollVisible
- WinColorOpacityGet
- WinColorOpacitySet
- WinFlash
- WinHalt
- WinIconPreload
- WinLayer
- WinLstEdit
- WinPicSaveImage
- WinSearch
- WinSearchClear
- WinSearchPath
- WinSearchPathGet
- WinShutdownBlock
- WinUpdate
- WinWbnExecCommand



obj -> WinPropGet(int1, var2) : logic

Ermitteln einer Eigenschaft eines Oberflächenobjekts

obj        Objekt

int1       Eigenschaftsausprägung

var2       Variable

Resultat logic Ermittlungserfolg

Verwandte Befehle,

Siehe WinPropSet(),

Eigenschaftsliste

Dieser Befehl liest eine Eigenschaft eines Oberflächenobjektes aus.

Als erster Parameter muss die Konstante der Eigenschaft übergeben werden. Die Konstanten setzen sich aus \_WinProp und dem Namen der Eigenschaft zusammen.

Im zweiten Parameter wird die Variable übergeben, in die der Wert der Eigenschaft kopiert werden soll.

### Beispiel:

```
local{ tTitle      : alpha; }...// Auslesen des Objektitels $Obj->WinPropGet(_WinPropCaption, tTitl
```

Das Kommando kann ebenfalls dazu verwendet werden, um zu ermitteln, ob ein bestimmtes Objekt eine Eigenschaft besitzt. Ist eine Eigenschaft nicht vorhanden, liefert der Befehl den Wert false zurück.



Alternativ kann die Eigenschaft auch wie folgt ermittelt werden:

### Beispiel:

```
local{ tTitle      : alpha; }...// Auslesen des Objektitels tTitle # $Obj->wpCaption;
```

Mögliche Laufzeitfehler:

\_ErrHdlInvalid Objekt (obj) ungültig

obj -> WinPropSet(int1, var2[, int3]) : logic  
 Setzen einer Eigenschaft eines Oberflächenobjekts



obj      Objekt  
 int1     Eigenschaftsausprägung  
 var2     Ausprägungswert  
 int3     Flag

Resultat logic Setzungserfolg

Verwandte Befehle,

Siehe WinPropGet(),  
Eigenschaftsliste

Dieser Befehl setzt eine Eigenschaft eines Oberflächenobjektes.

Als erster Parameter muss die Konstante der Eigenschaft übergeben werden. Die Konstanten setzen sich aus `_WinProp` und dem Namen der Eigenschaft zusammen.

Im zweiten Parameter wird der zu setzende Wert übergeben.

### Beispiel:

```
// Setzen des Objektitels $Obj->WinPropSet(_WinPropCaption, 'New caption');
```



Alternativ kann die Eigenschaft auch wie folgt gesetzt werden:

### Beispiel:

```
// Setzen des Objektitels $Obj->wpCaption # 'New caption';
```

Ein Vorteil der kompakteren Schreibweise gegenüber `WinPropSet()` und `WinPropGet()` besteht darin, dass schon während der Kompilierung eine Typprüfung vorgenommen wird:

```
$Obj->WinPropSet(_WinPropCaption, 100);
```

Diese Anweisung liefert während der Laufzeit den Rückgabewert false, da 100 vom Typ int ist.

```
$Obj->wpCaption # 100;
```

Dieser Befehl erzeugt bereits während der Kompilierung der entsprechenden Prozedur einen Fehler. Typ-Fehler dieser Art werden somit vermieden. Beim Lesen einer Eigenschaft, die im referenzierten Objekt nicht vorhanden ist, wird ein Laufzeitfehler erzeugt. Der Laufzeitfehler kann durch die Kapselung in einem try-Block unterbunden werden.

### Beispiel:

```
try{ ErrTryCatch(_ErrPropInvalid, true); $Obj->wpCaptionInt # 100; ...}
```

Der optionale Parameter (int3) muss nur angegeben werden, wenn zusätzliche Informationen einer Eigenschaft zugeordnet werden können. Entsprechende Hinweise befinden sich in den Beschreibungen der Eigenschaften.

**Beispiel:**

```
// Aktuelle Notizbuchseite ohne Fokuswechsel setzentHdlNotebook->WinPropSet(_WinPropCurrent, tHdl
```

Mögliche Laufzeitfehler:


ErrHdlInvalid Objekt (Obj) ungültig

obj -> PicExifInfo(var1, int2, int3, int4[, int5]) :  
logic



Auslesen von Exif-Informationen aus JPEG-Bildern

obj      Deskriptor eines  
         Picture-Objekts  
var1     Variable für die Information  
int2     Index oder Tag-Id der  
         Information  
int3     Modus (siehe Text)  
int4     Verzeichnis  
int5     Komponente (optional)

Resultat logic Funktion erfolgreich 

Siehe Verwandte Befehle, Picture

Das Exchangeable Image File Format (Exif) speichert Informationen über das mit einer Digitalkamera aufgenommene Bild. Hierzu gehören beispielsweise Informationen wie die Marke des Aufnahmegerätes und die Modell-Bezeichnung. Die Exif-Informationen sind dabei Bestandteil eines Bildes im JPEG-Format. Das Format ist von der Japan Electronics and Information Technology Industries Association (JEITA) definiert. Nähere Informationen über das Format sind auch auf [www.exif.org](http://www.exif.org) nachzulesen.

Mit dem Befehl PicExifInfo() können die Informationen zu einem bestehenden JPEG ausgelesen werden. Dazu muss das Bild in einem Picture-Objekt geladen sein. Der Deskriptor des Objekts wird in (obj) übergeben. Nach dem Aufruf des Befehls steht in der in (var1) übergebenen Variable die gewünschte Information. Die Variable muss einen geeigneten Typ besitzen, um die entsprechende Information aufzunehmen.

In (int2) wird entweder ein Index oder eine Tag-Id übergeben. Die Tag-Ids bezeichnen bestimmte Exif-Informationen, es sind aber nur einige Tags standardisiert. Abhängig von der Quelle der Bilder, kann es zu Abweichungen sowohl bei den Tag-Ids, als auch bei den Informationen kommen. Bei Bildern aus unbekannter Quelle sollten zunächst die vorhandenen Informationen über eine Schleife ermittelt werden (siehe Beispiele).

In (int3) wird der Modus übergeben. Über den Modus wird bestimmt, welche Informationen zu dem in (int2) angegebenen Index oder angegebener Tag-Id zurückgegeben werden. Folgende Konstanten stehen zur Verfügung:

<u>PicExifTagName</u>	Name des Tags
<u>PicExifTagTitle</u>	Tag-Titel
<u>PicExifTagDescription</u>	Tag-Beschreibung
<u>PicExifTagValue</u>	Tag-Wert
<u>PicExifTagType</u>	Tag-Typ (siehe <u>Type...</u> )
<u>PicExifTagComponents</u>	Anzahl der Komponenten bei zusammengesetzten Datentypen

Jede der Konstanten kann mit PicExifTagIndex kombiniert werden, um anzuzeigen, dass in (int2) ein Index und keine Tag-Id angegeben wurde. Bei der Ermittlung des Datentyps (PicExifTagType) wird ein Wert zurückgegeben, der mit den Type...-Konstanten verglichen werden kann. Handelt es sich um einen zusammengesetzten Datentyp, wird TypeOther zurückgegeben. Die Datentypen der

einzelnen Komponenten müssen dann durch erneute Aufrufe der Anweisung ermittelt werden. Die Anzahl der Komponenten kann mit PicExifTagComponents ermittelt werden. Die Nummer wird dann in (int5) angegeben.

Die Informationen innerhalb des JPEG sind in Verzeichnissen (IFD, Image File Directory) organisiert. Wird direkt über eine Tag-Id auf eine Information zugegriffen, wird als Verzeichnis -1 angegeben. Beim Zugriff über einen Index, muss das Verzeichnis angegeben werden. Folgende Konstanten können in (int4) übergeben werden:

-1	Zugriff über eine Tag-Id
<u>PicExifIfd0</u>	Enthält unter anderem Informationen über die Kamera
<u>PicExifIfd1</u>	Enthält unter anderem Informationen zur Bild-Auflösung
<u>PicExifIfdExif</u>	Enthält unter anderem Informationen zur Exif-Version
<u>PicExifIfdGps</u>	Enthält (GPS-)Informationen über den Standort
<u>PicExifIfdInterop</u>	Enthält Informationen zu den Interoperabilitätsregeln

Die Anweisung konnte erfolgreich ausgeführt werden, wenn true zurückgegeben wurde. false wird zum Beispiel dann zurückgegeben, wenn eine angegebene Tag-Id nicht existiert.

### Beispiele:

```
// Ermitteln des ersten Tag-NamenstTagIndex # 1;tResult # aPicture->PicExifInfo(tTagName, tTagInc
```

In der Beispiel-Datenbank CodeLibrary befindet sich ein ausführliches Beispiel.

Mögliche Laufzeitfehler:

<u>ErrHdlInvalid</u>	Der in (obj) übergebene Deskriptor ist ungültig
<u>ErrValueInvalid</u>	In (int3) oder (int4) wurde eine ungültige Konstante übergeben.
<u>ErrFldType</u>	Die in (var1) übergebene Variable hat den falschen Datentyp

Barcode-Befehle

Befehle zum Verwenden eines Barcode-Objektes

Verwandte

Befehle,

Siehe Barcode,

Befehlsgruppen,

Befehlsliste

## **Befehle**

- WinBarcodeSaveImage

obj -> WinBarcodeSaveImage(alpha1, int2) : int       

Inhalt des Barcode-Objektes als Bild speichern

obj Deskriptor des Barcode-Objektes

alpha1 Dateiname

Dateiformat

WinImageFormatBmp Windows-Bitmap-Datei erstellen


int2 WinImageFormatJpg JPEG-Datei erstellen

WinImageFormatPng PNG-Datei erstellen

WinImageFormatTif TIFF-Datei erstellen

Fehlercode

ErrGeneric Es ist ein interner Verarbeitungsfehler aufgetreten.

ErrOutOfMemory Nicht genügend Arbeitsspeicher zur Durchführung der Operation. 

Resultat int

ErrFsiOpenFailed Die Datei konnte nicht erzeugt werden.

ErrFsiWriteFault Fehler beim Schreiben in die Datei.

Siehe Verwandte Befehle, Barcode

Mit dem Befehl kann der Inhalt eines Barcode-Objektes (obj) als Bild gespeichert werden.

Im Argument (alpha1) muss der Pfad und Dateiname des zu erstellenden Bildes angegeben werden.

Als Dateiformat (int2) muss eine der folgenden Konstanten angegeben werden:

WinImageFormatBmp Windows-Bitmap-Datei erstellen

WinImageFormatJpg JPEG-Datei erstellen

WinImageFormatPng PNG-Datei erstellen

WinImageFormatTif TIFF-Datei erstellen

### Resultat

Konnte die Aktion erfolgreich durchgeführt werden, liefert der Befehl ErrOk zurück, ansonsten eine der folgenden Fehlerkonstanten:

ErrGeneric Es ist ein interner Verarbeitungsfehler aufgetreten.

ErrOutOfMemory Nicht genügend Arbeitsspeicher zur Durchführung der Operation.

ErrFsiOpenFailed Die Datei (alpha1) konnte nicht erzeugt werden.

ErrFsiWriteFault Fehler beim Schreiben in die Datei (alpha1).

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Deskriptor des Barcode-Objektes (obj) ist ungültig.



## Kontakt

ErrValueInvalid Ungültiges Format (int2) angegeben.

Befehle für dynamische Objekte

Befehle zur dynamischen Erstellung und Zerstörung von Objekten

Verwandte

Siehe Befehle,  
Befehlsgruppen,  
Befehlsliste

## **Befehle**

- WinAdd
- WinCopy
- WinCreate
- WinDestroy
- WinRemove



obj -> WinAdd(handle1[, int2[, handle3]]) : int  
Oberflächenobjekt in ein anderes Oberflächenobjekt laden

obj Elternojekt

handle1 einzufügendes Oberflächenobjekt


Optionen (optional)

WinAddHidden MDI-Fenster  
unsichtbar  
laden

int2 WinDialogMaximized MDI-Fenster  
maximiert  
laden

WinDialogMinimized MDI-Fenster  
minimiert  
laden

handle3 Deskriptor des nachfolgenden  
Objektes (optional)

Resultat int Laderesultat   
ErrOk Laden erfolgreich

Siehe Verwandte Befehle,  
WinAddByName(), WinCreate(),  
WinRemove(), WinDestroy(),  
Ereignisabläufe MdiFrame

Der Befehl fügt dem angegebenen Elternojekt (obj) das Oberflächenobjekt (handle1) hinzu. Bei dem Oberflächenobjekt muss es sich um einen MdiFrame oder um ein mit WinCreate() angelegtes Objekt handeln. Das Objekt (handle1) kann bereits weitere mit WinCreate() und WinAdd() untergeordnete Oberflächenobjekte enthalten.

Ist das Objekt (handle1) ein MdiFrame, kann über den optionalen Parameter (int2) mit der Konstanten WinAddHidden angegeben werden, dass das MdiFrame-Objekt unsichtbar geladen werden soll. In diesem Fall muss das MdiFrame-Objekt später mit dem Befehl WinUpdate() mit dem Parameter WinUpdOn gezeichnet werden. Mit den Optionen WinDialogMaximized bzw. WinDialogMinimized kann der MdiFrame maximiert oder minimiert werden.

Sind dem MDI-Fenster über die Eigenschaft DbRecBuf eigene Feldpuffer zugeordnet worden, stehen diese nach diesem Befehl zur Verfügung und können initialisiert werden.

Wurde das Oberflächenobjekt (handle1) mit WinCreate() erstellt, kann das nachfolgende Objekt im Argument (handle3) angegeben und somit die Objektreihenfolge definiert werden.



Das Objekt (handle1) darf zuvor nicht bereits mit WinAdd() zu einem anderen Objekt hinzugefügt werden, außer es wurde anschließend mit WinRemove() wieder entfernt.

Bei dem Oberflächenobjekt (handle1) kann es sich nicht um ein Frame-, AppFrame- oder TrayFrame-Objekt handeln. Um diese zu starten, muss, auch für dynamisch erstellte, weiterhin der Befehl WinDialogRun() verwendet werden.

Wird das Elternobjekt (obj) bereits angezeigt, erfolgt auch die Darstellung des hinzugefügten Oberflächenobjektes (handle1) direkt nach der Ausführung des Befehls.

Als Resultat kann der Fehlerwert ErrType zurückgegeben werden, wenn das hinzuzufügende Objekt (handle1) nicht in das Elternobjekt (obj) eingefügt werden kann. Das Resultat ist ErrExists, wenn das hinzuzufügende Objekt (handle1) bereits einem Objekt hinzugefügt wurde. Konnte ein MdiFrame nicht zu einem AppFrame hinzugefügt werden, ist das Resultat ErrOutOfMemory. Das Resultat ist ErrUnavailable, wenn eine Spalte vor einer bestehenden Spalte eingefügt wird und die Liste bereits Inhalt besitzt. Ist kein Fehler aufgetreten, wird ErrOk zurückgegeben.

### **Hinweise für Spalten-Objekte**

Dynamische Spalten-Objekte können auch einer nicht dynamisch erstellten DataList oder RecList hinzugefügt werden. Diese darf auch bereits nicht dynamisch erstellte Spalten enthalten.

Ein Spalten-Objekt besitzt eine Anzeigeposition und eine Indexposition. Die Anzeigeposition definiert, wo die Spalte innerhalb der Liste angezeigt wird und kann sich z. B. durch Benutzerinteraktionen mit der Spalte (z. B. Verschieben der Spalte durch den Anwender) ändern. Die Anzeigeposition kann durch die Eigenschaft ClmOrder gesetzt oder auch abgefragt werden.

Die Indexposition definiert, welche Position die Zelle im Datensatz (Zeile) besitzt. Diese Position wird den Befehlen WinLstCellSet() und WinLstCellGet() übergeben, wenn die Daten einer Zelle gesetzt oder abgefragt werden sollen. Die Indexposition kann mit dem Befehl WinInfo() und der Option WinItem ermittelt werden. Bei WinAdd() unter Angabe einer nachfolgenden Spalte erhält die neu hinzugefügte Spalte die die Indexposition der nachfolgenden Spalte. Die Anzeigeposition wird über die Eigenschaft ClmOrder definiert.



Die Angabe eines nachfolgenden Spalten-Objektes ist nicht zulässig, wenn die Liste bereits einen Inhalt besitzt (Rückgabewert ErrUnavailable). Es können jedoch Spalten am Ende eingefügt werden (WinAdd() ohne nachfolgendes Objekt).

Nachdem ein Spalten-Objekt einer DataList hinzugefügt wurde, hat diese zunächst keinen Inhalt. Mit dem Befehl WinLstCellSet() kann der Spalteninhalt entsprechend des durch ClmType definierten Spalten-Typs gesetzt werden. Der Befehl WinLstCellGet() liefert den Wert false, wenn die Zelle der Spalte noch keinen Inhalt besitzt.

### **Hinweise für GroupColumn-Objekte**

Dynamische GroupColumn-Objekte können auch einem nicht dynamisch erstellten RecView- oder einem nicht dynamisch erstellten, übergeordneten GroupColumn-Objekt hinzugefügt werden.

Ein GroupColumn-Objekt besitzt eine Anzeige- und eine Indexposition. Die Anzeigeposition definiert, wo das GroupColumn-Objekt im RecView bzw. im

übergeordneten GroupColumn-Objekt angezeigt wird. Die Anzeigeposition kann durch die Eigenschaft VisibleOrder gesetzt und abgefragt werden.

Die Indexposition definiert eine eindeutige fortlaufende Nummer für die Eigenschaft SelectorItem bzw. SelectorSubItem.

Die Angabe eines nachfolgenden GroupColumn-Objektes ist nicht zulässig, wenn das RecView bereits einen Inhalt besitzt. Es können jedoch GroupColumn-Objekte am Ende eingefügt werden (WinAdd() ohne nachfolgendes Objekt).

Wird das RecView-Objekt angezeigt, während WinAdd() für ein GroupColumn-Objekt durchgeführt wird, dann hat dies zur Folge, dass das Ereignis EvtLstGroupInit aufgerufen wird.

### **Hinweise zum PopupList-Objekt**

Das Objekt kann den Eingabeobjekten (Edit, IntEdit, ...) hinzugefügt werden. Dem PopupList-Objekt kann wiederum ein DataListPopup, RecListPopup- oder StoListPopup-Objekt hinzugefügt werden.


Einem GroupColumn-Objekt kann ein weiteres GroupColumn-Objekt untergeordnet werden. Dies ist jedoch nur zulässig, wenn das übergeordnete Objekt nicht bereits einem GroupColumn-Objekt untergeordnet ist und das unterzuordnende GroupColumn-Objekt seinerseits keine untergeordneten GroupColumn-Objekte enthält.

### **Beispiele:**

```
// MDI-Frame 'Addresses' laden tMdiFrame # WinOpen('Addresses');// Wenn hinzufügen des MDI-Fensters
```

Mögliche Laufzeitfehler:

<u>_ErrHdlInvalid</u>	Elternobjekt (obj) oder hinzuzufügendes Oberflächenobjekt (handle1) ungültig oder das Oberflächenobjekt (handle1) ist nicht dynamisch erstellt worden.
<u>_ErrMemExhausted</u>	Fenstererstellung ist fehlgeschlagen.
<u>_ErrIllegalOp</u>	Das nachfolgende Objekt (handle3) ist angegeben, jedoch kein Oberflächenobjekt oder kein Kindobjekt von dem angegebenen Elternobjekt (obj). Das hinzuzufügende Objekt (handle1) ist ein <u>Frame</u> , <u>MdiFrame</u> oder <u>AppFrame</u> und ein nachfolgendes Objekt (handle3) ist angegeben.

obj -> WinCopy([int1]) : 

handle  
Oberflächenobjekt kopieren  
obj Oberflächenobjekt  
Optionen (optional)  
    WinCopyDefault Alle Eigenschaften  
                            und Ereignisse  
                            werden kopiert  
                            (Standard).  
int1  
    WinCopyNoEvents Ereignisfunktionen  
                            werden nicht  
                            kopiert.  
                            Deskriptor des neuen  
                            Oberflächenobjektes oder  
                            Fehlerwert:  
                            ErrType Das  
Resultat handle Oberflächenobjekt,  
                            oder eines der  
                            untergeordneten  
                            Objekte kann nicht  
                            kopiert werden.

Siehe Verwandte Befehle, WinCreate(),  
      WinAdd()

Mit diesem Befehl wird das Oberflächenobjekt (obj) inklusive der untergeordneten Oberflächenobjekte kopiert. Hierbei werden alle Eigenschaften und Ereignisse ebenfalls übernommen.

Bei (obj) kann es sich sowohl um ein dynamisch erstelltes (WinCreate()) als auch um ein aus der Datenbank geladenes Objekt handeln (WinOpen()). Sofern es sich um ein Objekt aus der Datenbank handelt, muss es sich bei (obj) und all seinen untergeordneten Objekten um dynamisch erstellbare Objekte handeln, also um solche Objekte, die auch mit WinCreate() erstellt werden können.

Folgende Optionen können im optionalen Argument (int1) angegeben werden:

WinCopyDefault Alle Eigenschaften und Ereignisse werden kopiert (Standard).  
WinCopyNoEvents Ereignisfunktionen werden nicht kopiert.  
Um das kopierte Oberflächenobjekt anzuzeigen, muss es einem sichtbaren Oberflächenobjekt mit WinAdd() hinzugefügt werden.

Konnte das Objekt kopiert werden, wird der Deskriptor auf das neue Objekt zurückgegeben, andernfalls der Fehlerwert ErrType.

### Beispiele:

```
// Groupbox inklusive Ereignisse kopieren
tNewObj # $grpGroup->WinCopy();
if (tNewObj > 0) tFrame-
```

Mögliche Laufzeitfehler:

## Kontakt

- ErrHdlInvalid Oberflächenobjekt (obj) enthält keinen gültigen Deskriptor oder verweist nicht auf ein dynamisch erstellbares Objekt.
- ErrValueInvalid Bei (int1) wurde eine ungültige Option angegeben.

```
WinCreate(int1[, alpha2[,  
alpha3[, handle4[,  
handle5]]]]) : handle
```



Oberflächenobjekt erzeugen

int1 Objekttyp

alpha2 Name des Objektes (optional)

alpha3 Caption des Objektes (optional)

handle4 Deskriptor des Elternobjektes  
(optional)


handle5 Deskriptor des nachfolgenden  
Objektes (optional)

Resultat handle Objekt-Deskriptor oder  
Fehlerwert

Siehe Verwandte Befehle, WinAdd(),  
WinRemove(), WinDestroy()

Der Befehl erstellt ein Oberflächenobjekt vom Typ (int1) und liefert den Deskriptor auf dieses zurück. Optional können sowohl der Name im Parameter (alpha2), als auch die Caption im Parameter (alpha3) angegeben werden. Die Caption (alpha3) wird nur ausgewertet, wenn das entsprechende Objekt diese Eigenschaft besitzt.

Wird das Elternobjekt (handle4) angegeben, wird das erstellte Objekt nach dem Setzen der Eigenschaften Name und Caption unmittelbar an WinAdd() übergeben. Das nachfolgende Objekt (handle5) wird nur ausgewertet, wenn das Elternobjekt (handle4) angegeben ist.

 Um das erstellte Objekt anzuzeigen muss es einem sichtbaren Oberflächenobjekt durch Angabe eines Elternobjektes (handle4) oder durch Übergabe an WinAdd() hinzugefügt werden.

Das neu erstellte Objekt kann nicht über das with-Statement angesprochen werden. Die Suche über WinSearch() ist nur möglich, wenn das Objekt einem anderen Oberflächenobjekt untergeordnet ist. Vorzugsweise sollte der zurückgelieferte Deskriptor verwendet werden.

Für die Ausführung von dynamisch erzeugten Frame- und AppFrame-Objekte muss weiterhin der Befehl WinDialogRun() verwendet werden. Zum Schließen eines angezeigten Fensters kann wahlweise WinClose(), WinRemove() oder WinDestroy() benutzt werden. Um das nicht mehr angezeigte Fenster zu zerstören, stehen WinClose() und WinDestroy() zur Verfügung.

Als Resultat kann der Fehlerwert \_ErrType zurückgegeben werden, wenn ein Objekt von dem Typ (int1) nicht angelegt, oder nicht in das Elternobjekt (handle4) eingefügt werden kann. Ist kein Fehler aufgetreten, wird der Deskriptor des angelegten Objektes zurückgegeben.

### Beispiele:

```
// NotebookPage zu einem Notebook-Objekt hinzufügen WinCreate(_WinTypeNotebookPage, 'nbpView', 'Ar
```

### Unicode



Dynamisch erstellte Objekte können auch Unicode-Objekte sein. Die Festlegung, ob ein Objekt Unicode ist oder nicht, wird bei WinCreate bzw. WinAdd() getroffen.

Sofern der Deskriptor des Elternobjektes bei WinCreate angegeben ist, wird ein Unicode-Objekt erstellt, sofern es sich beim Elternobjekt um ein Unicode-Objekt handelt.

Wird ein dynamisch erstelltes Objekt per WinAdd() zu einem Unicode-Objekt hinzugefügt, dann wird das dynamisch erstellte Objekt ebenfalls Unicode.

Bestimmte Eigenschaften eines Objektes können Unicode-Eigenschaften sein. Bei dynamisch erstellten Unicode-Objekten wird der UTF-8-Zeichensatz erwartet.

### Beispiele:

```
myLabel # WinCreate(_WinTypeLabel);myLabel->wpCaption # CaptionString; // Objekt nicht Unicode. D
```

```
myLabel # WinCreate(_WinTypeLabel,'myLabel',CaptionString,UnicodeFrame); // myLabel wird als Unic
```

### Folgende Objekte können dynamisch erzeugt werden:

<b>Palette</b>	<b>Objekt</b>	<b>Objekttyp</b>
Form	<u>Frame</u>	<u>_WinTypeFrame</u> / <u>_WinTypeDialog</u>
Form	<u>AppFrame</u>	<u>_WinTypeAppFrame</u>
Form	<u>MdiFrame</u>	<u>_WinTypeMdiFrame</u>
Eingabe	<u>Edit</u>	<u>_WinTypeEdit</u>
Eingabe	<u>IntEdit</u>	<u>_WinTypeIntEdit</u>
Eingabe	<u>BigIntEdit</u>	<u>_WinTypeBigIntEdit</u>
Eingabe	<u>FloatEdit</u>	<u>_WinTypeFloatEdit</u>
Eingabe	<u>DecimalEdit</u>	<u>_WinTypeDecimalEdit</u>
Eingabe	<u>TimeEdit</u>	<u>_WinTypeTimeEdit</u>
Eingabe	<u>DateEdit</u>	<u>_WinTypeDateEdit</u>
Eingabe	<u>ColorEdit</u>	<u>_WinTypeColorEdit</u>
Eingabe	<u>FontNameEdit</u>	<u>_WinTypeFontNameEdit</u>
Eingabe	<u>FontSizeEdit</u>	<u>_WinTypeFontSizeEdit</u>
Eingabe	<u>TextEdit</u>	<u>_WinTypeTextEdit</u>
Eingabe	<u>RtfEdit</u>	<u>_WinTypeRTFEdit</u>
Schaltflächen	<u>Button</u>	<u>_WinTypeButton</u>
Schaltflächen	<u>MenuButton</u>	<u>_WinTypeMenuButton</u>
Schaltflächen	<u>ColorButton</u>	<u>_WinTypeColorButton</u>
Schaltflächen	<u>Radiobutton</u>	<u>_WinTypeRadioButton</u>
Schaltflächen	<u>Checkbox</u>	<u>_WinTypeCheckBox</u>
Ansicht	<u>DataList</u>	<u>_WinTypeDataList</u>

		Kontakt
Ansicht	<u>DataListPopup</u>	<u>WinTypeDataListPopup</u>
Ansicht	<u>StoList</u>	<u>WinTypeStoList</u>
Ansicht	<u>StoListPopup</u>	<u>WinTypeStoListPopup</u>
Ansicht	<u>RecList</u>	<u>WinTypeRecList</u>
Ansicht	<u>RecListPopup</u>	<u>WinTypeRecListPopup</u>
Ansicht	<u>RecView</u>	<u>WinTypeRecView</u>
Ansicht	<u>TreeView</u>	<u>WinTypeTreeView</u>
Ausgabe	<u>Label</u>	<u>WinTypeLabel</u>
Ausgabe	<u>Icon</u>	<u>WinTypeIcon</u>
Ausgabe	<u>Picture</u>	<u>WinTypePicture</u>
Ausgabe	Overlay ( <u>Picture</u> )	<u>WinTypePicture</u>
Ausgabe	<u>MetaPicture</u>	<u>WinTypeMetaPicture</u>
Ausgabe	Overlay ( <u>MetaPicture</u> )	<u>WinTypeMetaPicture</u>
Ausgabe	<u>Animation</u>	<u>WinTypeAnimation</u>
Ausgabe	<u>DocView</u>	<u>WinTypeDocView</u>
Ausgabe	<u>WebNavigator</u>	<u>WinTypeWebNavigator</u>
Ausgabe	<u>Chromium</u>	<u>WinTypeChromium</u>
Ausgabe	<u>PrtJobPreview</u>	<u>WinTypePrtJobPreview</u>
Toolbar	<u>Toolbar-Button</u>	<u>WinTypeToolbarButton</u>
Toolbar	<u>Statusbar-Button</u>	<u>WinTypeStatusbarButton</u>
Anordnung	<u>Groupbox</u>	<u>WinTypeGroupBox</u>
Anordnung	<u>Notebook</u>	<u>WinTypeNotebook</u>
Anordnung	<u>NotebookPage</u>	<u>WinTypeNotebookPage</u>
Anordnung	<u>GroupSplit</u>	<u>WinTypeGroupSplit</u>
Anordnung	<u>GroupTile</u>	<u>WinTypeGroupTile</u>
Anordnung	<u>Scrollbox</u>	<u>WinTypeScrollbox</u>
Anordnung	<u>Divider</u>	<u>WinTypeDivider</u>
COM	<u>CtxOffice</u>	<u>WinTypeCtxOffice</u>
COM	<u>CtxAdobeReader</u>	<u>WinTypeCtxAdobeReader</u>
COM	<u>CtxDocEdit</u>	<u>WinTypeCtxDocEdit</u>
Weitere	<u>Hyperlink</u>	<u>WinTypeHyperLink</u>
Weitere	<u>Calendar</u>	<u>WinTypeCalendar</u>
Weitere	<u>Progress</u>	<u>WinTypeProgress</u>
Weitere	<u>RecNavigator</u>	<u>WinTypeRecNavigator</u>
	<u>PopupList</u>	<u>WinTypePopupList</u>
	<u>Column</u>	<u>WinTypeListColumn</u>

GroupColumn

WinTypeGroupColumn

GroupTile-Button

WinTypeGroupTileButton

### **Besonderheiten bei der Erstellung dynamischer Spalten- und GroupColumn-Objekte:**

Eine dynamisch angelegte Spalte (WinTypeListColumn) hat standardmäßig folgende Eigenschaften:

- Sie ist vom Typ alpha (ClmType = TypeAlpha)
- Sie ist sichtbar (Visible = true)
- Spaltenbreite ist 70 Pixel (ClmWidth = 70)
- ClmOrder steht auf dem höchstmöglichen Wert (MaxInt), so dass ohne weitere Veränderung der Eigenschaft ein Hinzufügen der Spalte am Ende erfolgt (Anzeigeposition).

Die Eigenschaft ClmType kann nur gesetzt werden, wenn die Spalte noch keiner Liste hinzugefügt wurde, oder die Liste leer ist.

Die Spalte kann nur vor einer bestehenden Spalte (handle5) eingefügt werden, wenn die Liste leer ist. Andernfalls ist der Rückgabewert ErrUnavailable. Am Ende einer Liste können Spalten immer eingefügt werden.

Eine dynamisch angelegte GroupColumn (WinTypeGroupColumn) hat standardmäßig folgende Eigenschaften:

- ContentType ist WinContentTypeText
- ContentSource ist WinContentSourceField
- ContentHeightMax = 0 (keine Höhenbegrenzung)
- WordBreak = false
- AreaMarginLeft, AreaMarginTop, AreaMarginRight und AreaMarginBottom sind auf 4 Pixel eingestellt.
- BorderWidth ist 1 und BorderType ist WinBorderTypeNone
- Sie ist sichtbar (Visible = true)
- Spaltenbreite ist 70 Pixel (ClmWidth = 70)
- VisibleOrder steht auf 0. Die Eigenschaft kann erst nach dem Hinzufügen der Spalte zum RecView bzw. zur übergeordneten Spalte gesetzt werden.

Mögliche Laufzeitfehler:

<u>ErrHdlInvalid</u>	Deskriptor des nächsten Objektes (handle5) ist angegeben, jedoch kein Elternobjekt (handle4).
<u>ErrMemExhausted</u>	Anlegen des Objektes ist fehlgeschlagen. Dies kann zum Beispiel auftreten, wenn die maximale Anzahl der Oberflächenobjekte von 10.000 erreicht ist.
<u>ErrValueInvalid</u>	Name (alpha2) ist nicht leer und enthält keinen gültigen Objektnamen. (Beschränkungen des Namens siehe <u>Name</u> )
<u>ErrIllegalOp</u>	Das nachfolgende Objekt (handle5) ist angegeben, jedoch kein Oberflächenobjekt oder kein Kindobjekt von dem angegebenen Elternobjekt (handle4).

## Kontakt

Das angelegte Objekt (int1) ist ein Frame, MdiFrame oder AppFrame und ein nachfolgendes Objekt (handle5) ist angegeben.

### \_ErrPropInvalid

Die Caption (alpha3) des neuen Objektes konnte nicht gesetzt werden. Der Fehler tritt auf, wenn es zwar eine Caption-Eigenschaft beim Objekt gibt, diese jedoch nicht gesetzt werden kann (z. B. weil die Eigenschaft read-only ist).



obj -> WinDestroy([logic1])

Frame- oder Oberflächenobjekt zerstören

obj zu zerstörendes  
Oberflächenobjekt  
nur untergeordnete

logic1 Objekte zerstören  
(optional)

Verwandte Befehle,

Siehe WinCreate(),

WinAdd(),

WinRemove()

Der Befehl zerstört ein durch WinCreate() erstelltes Oberflächenobjekt. Das Objekt wird im Argument (obj) angegeben. Sollte das Objekt noch nicht mit WinRemove() vom Elternobjekt gelöst worden sein, geschieht dies hier automatisch.

Der Deskriptor (obj) ist nach der Zerstörung nicht mehr gültig.

Enthält das Objekt (obj) untergeordnete Objekte, werden diese ebenfalls zerstört.

Dynamisch erstellte Frame-Objekte können alternativ auch mit WinClose() geschlossen und zerstört werden. Dabei ist zu beachten, dass ein angezeigtes Fenster beim ersten Aufruf von WinClose() nur geschlossen wird. Der Befehl WinDestroy() muss hingegen nur einmal aufgerufen werden.

Wird im Argument (logic1) true übergeben, werden nur die untergeordneten dynamisch erstellten Oberflächenobjekte entfernt und zerstört. In diesem Fall wird das Objekt (obj) weder entfernt noch zerstört.



Der übergebene Deskriptor (obj) muss mit WinCreate() erstellt worden sein.

### **Hinweise zum Entfernen von Spalten-Objekten:**

Wenn die Liste Daten (Zeilen) enthält, kann immer nur die zuletzt hinzugefügte Spalte (mit der größten Indexposition) entfernt bzw. zerstört werden. Die Angabe einer anderen Spalte führt zu dem Laufzeitfehler ErrIllegalOp. Sofern die Liste keine Daten enthält, kann eine beliebige dynamisch erstellte Spalte entfernt werden.

### **Hinweise zum Entfernen von GroupColumn-Objekten:**

Wenn das RecView Gruppen enthält, kann immer nur das zuletzt hinzugefügte GroupColumn-Objekt (mit der größten Indexposition) entfernt bzw. zerstört werden. Dies gilt auch bei untergeordneten GroupColumn-Objekten. Sofern das RecView leer ist (also keine Gruppen enthält), kann ein beliebiges GroupColumn-Objekt entfernt werden.

### **Beispiel:**

```
sub EvtClicked( aEvt                : event;      // Ereignis): logic; local {   tDlg
```

Mögliche Laufzeitfehler:

\_\_ErrHdllInvalid Das Oberflächenobjekt (obj) ist ungültig oder wurde nicht mit WinCreate() dynamisch erstellt.

obj -> 

WinRemove()

Objekt entfernen

obj zu entfernendes  
Oberflächenobjekt

Verwandte Befehle,

Siehe WinCreate(),

WinAdd(),

WinDestroy()

Dieser Befehl entfernt ein mit WinAdd() hinzugefügtes Oberflächenobjekt (obj) aus seinem Elternobjekt.

Enthält das entfernte Objekt untergeordnete Objekte, werden diese mit entfernt.

Ein entferntes Objekt kann mit WinAdd() wieder einem Oberflächenobjekt hinzugefügt, oder mit WinDestroy() zerstört werden.

Wird als (obj) ein Fenster-Objekt angegeben, wird es geschlossen. Es kann anschließend mit WinClose() oder WinDestroy() zerstört werden.



Der übergebene Deskriptor (obj) muss mit WinCreate() erstellt worden sein.

### **Hinweise zum Entfernen von Spalten-Objekten:**

Wenn die Liste Daten (Zeilen) enthält, kann immer nur die zuletzt hinzugefügte Spalte (mit der größten Indexposition) entfernt bzw. zerstört werden. Die Angabe einer anderen Spalte führt zu dem Laufzeitfehler ErrIllegalOp. Sofern die Liste keine Daten enthält, kann eine beliebige dynamisch erstellte Spalte entfernt werden.

### **Hinweise zum Entfernen von GroupColumn-Objekten:**

Wenn das RecView Gruppen enthält, kann immer nur das zuletzt hinzugefügte GroupColumn-Objekt (mit der größten Indexposition) entfernt bzw. zerstört werden. Dies gilt auch bei untergeordneten GroupColumn-Objekten. Sofern das RecView leer ist (also keine Gruppen enthält), kann ein beliebiges GroupColumn-Objekt entfernt werden.

### **Beispiel:**

```
sub EvtClicked( aEvt : event; // Ereignis): logic; local { tDlg
```

Mögliche Laufzeitfehler:

ErrHdlInvalid Das Oberflächenobjekt (obj) ist ungültig oder wurde nicht mit WinCreate() dynamisch erstellt.

ErrIllegalOp Das Objekt (obj) besitzt kein Elternobjekt oder das Objekt ist eine Spalte, die sich nicht am Ende einer gefüllten Liste befindet.

## Chromium-Befehle

Befehle und Konstanten für Chromium-Objekte

Liste sortiert

nach

Siehe Gruppen,  
Alphabetische  
Liste aller  
Befehle

### **Befehle**

- WinCroNavigate
- WinCroPrint
- WinCroReload
- WinCroSelection

### **Konstanten**

- WinCroDevToolsSideBottom
- WinCroDevToolsSideLeft
- WinCroDevToolsSideNone
- WinCroDevToolsSideRight
- WinCroDevToolsSideTop
- WinCroNavBack
- WinCroNavForward
- WinCroReloadIgnoreCache
- WinCroReloadNormal
- WinCroSelCopy
- WinCroSelCut
- WinCroSelDelete
- WinCroSelPaste
- WinCroSelSelectAll



obj ->



WinCroNavigate(int1)

Navigiert zu einer URL

obj    Objekt

      (Chromium-Objekt)

int1    Art der Navigation

Siehe Verwandte Befehle

Navigiert zu einer URL, deren Inhalt im Verlauf der Navigation bereits angezeigt wurde, falls eine solche Navigation stattgefunden hat.

Im Parameter (int1) können folgende Optionen übergeben werden:

- **WinCroNavBack (1)**

      Navigiert zur vorhergehenden Seite.

- **WinCroNavForward (2)**

      Navigiert zur nächsten Seite.

Mögliche Laufzeitfehler

ErrHdlInvalid    Bei (obj) handelt es sich nicht um ein Chromium-Objekt.

ErrValueInvalid    In Argument (int1) wurde keiner der gültigen Werte für die Navigation angegeben.

obj ->

WinCroPrint()

Inhalt drucken

obj Objekt

(Chromium-Objekt)

Siehe Verwandte Befehle

Der Befehl ruft einen Dialog zum Drucken des Inhaltes der aktuell angezeigten Seite aus.

Mögliche Laufzeitfehler

\_ErrHdlInvalid Bei (obj) handelt es sich nicht um ein Chromium-Objekt.





obj -> WinCroReload(int1)

Lädt den Inhalt der aktuellen URL erneut

obj    Objekt

      (Chromium-Objekt)

int1   Optionen

Siehe Verwandte Befehle

Der Befehl lädt den Inhalt der aktuell angezeigten Seite erneut. Über Optionen kann gesteuert werden, ob die Seite neu angefordert oder aus dem Cache geladen werden soll.

Im Parameter (int1) können folgende Optionen übergeben werden:

- **WinCroReloadIgnoreCache (1)**

      Zwischengespeicherte Inhalte werden erneut angefordert.

- **WinCroReloadNormal (0)**

      Zwischengespeicherte Inhalte werden nicht erneut angefordert (default).

Mögliche Laufzeitfehler

ErrHdlInvalid    Bei (obj) handelt es sich nicht um ein Chromium-Objekt.

ErrValueInvalid In Argument (int1) wurde keine der gültigen Optionen angegeben.



obj -> WinCroSelection(int1)

Aktionen für den ausgewählten Inhalt

obj    Objekt

obj    (Chromium-Objekt)

int1   Aktion

Siehe Verwandte Befehle

Der Befehl führt Aktionen auf den aktuell angezeigten Inhalt aus, sofern möglich.

Im Parameter (int1) können folgende Optionen übergeben werden:

- **\_WinCroSelCopy (1)**

Kopiert den ausgewählten Inhalte in die Zwischenablage.

- **\_WinCroSelCut (2)**

Schneidet den ausgewählten Inhalt in die Zwischenablage aus.

- **\_WinCroSelDelete (4)**

Löscht den ausgewählten Inhalt.

- **\_WinCroSelPaste (3)**

Ersetzt den ausgewählten Inhalt durch den Inhalt in der Zwischenablage.

- **\_WinCroSelSelectAll (5)**

Wählt den gesamten Inhalt aus.

Mögliche Laufzeitfehler

\_ErrHdlInvalid    Bei (obj) handelt es sich nicht um ein Chromium-Objekt.

\_ErrValueInvalid In Argument (int1) wurde keine der gültigen Optionen angegeben.

## CodeEdit-Befehle

Befehle zum Verwenden eines CodeEdit-Objektes

Verwandte

Befehle,

Siehe CodeEdit,

Befehlsgruppen,

Befehlsliste

## **Befehle**

- WinEditorBookmarkGet
- WinEditorBookmarkToggle
- WinEditorGetSelectedText
- WinEditorGetSelection
- WinEditorGoTo
- WinEditorHighlight
- WinEditorKeywordsAdd
- WinEditorKeywordsRemove
- WinEditorKeywordsUpdate
- WinEditorLoad
- WinEditorReplaceSelectedText
- WinEditorSave
- WinEditorSearch
- WinEditorSetSelection

obj -> WinEditorBookmarkGet(int1[, var alpha2[,  
int3]]) : logic



Lesezeichen im CodeEdit-Objekt setzen und entfernen

obj            Deskriptor des  
              CodeEdit-Objektes

int1           Zeile

var alpha2 Beschreibung (optional)

int4           View-Nummer (optional)

Resultat     logic Lesezeichen gesetzt?

Siehe         Verwandte Befehle,  
              WinEditorBookmarkToggle()

Mit dieser Funktion kann im CodeEdit-Objekt (obj) ermittelt werden, ob in Zeile (int1) ein Lesezeichen gesetzt ist.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Optional kann die Beschreibung (alpha2) des Lesezeichens ermittelt werden.

In (int3) kann optional die Nummer des Views angegeben werden, in dem das Lesezeichen ermittelt werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

## Resultat

Als Resultat wird zurückgegeben, ob in der Zeile (int1) ein Lesezeichen gesetzt ist.


## Beispiel:

```
// Lesezeichen ermitteltSet # $CodeEdit->WinEditorBookmarkGet(42); // Lesezeichen und Beschreibung
```

Mögliche Laufzeitfehler:


\_ErrHdlInvalid Der Deskriptor des CodeEdit (obj) ist ungültig.

\_ErrValueInvalid Ungültige oder nicht existierende View-Nummer (int3) angegeben.

obj -> WinEditorBookmarkToggle(int1[, alpha2[, int3[,  logic4]]]) : logic

Lesezeichen im CodeEdit-Objekt setzen und entfernen

obj      Deskriptor des  
          CodeEdit-Objektes  
int1      Zeile  
alpha2   Beschreibung (optional)  
int4      View-Nummer (optional)  
logic4    Explizit setzen / entfernen  
          (optional)

Resultat logic Lesezeichen gesetzt? 


Siehe Verwandte Befehle,  
WinEditorBookmarkGet()

Mit dieser Funktion können Lesezeichen im CodeEdit-Objekt (obj) gesetzt oder entfernt in der Zeile (int1) werden. Ist noch kein Lesezeichen gesetzt, wird es in der Zeile gesetzt, ansonsten das bestehende Lesezeichen entfernt.

 Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Optional kann eine Beschreibung (alpha2) für das Lesezeichen angegeben werden.

In (int3) kann optional die Nummer des Views angegeben werden, in dem das Lesezeichen gesetzt bzw. entfernt werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

 Die gesetzten Lesezeichen werden für alle Views übernommen, bei denen die Eigenschaften FileName und EditorTextType auf den gleichen Wert gesetzt sind. Ist Argument (logic4) angegeben, wird in der angegebenen Zeile (int1) bei true explizit ein Lesezeichen gesetzt, auch wenn bereits ein Lesezeichen vorhanden ist. Eine eventuell vorhandene Beschreibung wird hierbei durch (alpha2) überschrieben. Ist hier false oder NULL angegeben, wird ein vorhandenes Lesezeichen entfernt, jedoch nicht hinzugefügt, wenn keines existiert.

## Resultat

Als Resultat wird zurückgegeben, ob in der Zeile (int1) ein Lesezeichen gesetzt ist.

## Beispiel:

```
// Lesezeichen switchentSet # $CodeEdit->WinEditorBookmarkToggle(42); // Lesezeichen explizit setzen
```

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Deskriptor des CodeEdit (obj) ist ungültig.

ErrValueInvalid Ungültige oder nicht existierende View-Nummer (int3) angegeben.

obj -> WinEditorGetSelectedText([handle1[,  
int2]]) : alpha



Selektierten Text im CodeEdit-Objekt ermitteln

obj      Deskriptor des CodeEdit-Objektes

handle1    Deskriptor eines  
            Memory-Objektes (optional)

int2      View-Nummer (optional)

Resultat alpha    Selektierter Text



Verwandte Befehle,

Siehe WinEditorGetSelection(),

WinEditorReplaceSelectedText()

Mit dieser Funktion wird im CodeEdit-Objekt (obj) der selektierte Text ermittelt. Wird ein Memory-Objekt (handle1) wird der selektierte Text in dieses geschrieben. Andernfalls wird der Text zurückgegeben.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

In (int2) kann optional die Nummer des Views angegeben werden, in dem der selektierte Text ermittelt werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

### Beispiel:

```
// Selektierten Text ermittelntMem # MemAllocate(_MemAutoSize);$CodeEdit->WinEditorGetSelectedText
```


Mögliche Laufzeitfehler:

\_ErrHdlInvalid    Der Deskriptor des CodeEdit (obj) oder des Memory-Objektes (handle1) ist ungültig.

\_ErrValueInvalid Ungültige oder nicht existierende View-Nummer (int2) angegeben.



## Kontakt

obj -> WinEditorGetSelection(int1[, int2]) : point 

Selektionsbereich im CodeEdit-Objekt ermitteln

obj      Deskriptor des CodeEdit-Objektes

Optionen

\_WinEditorSelGetCaretPos      Cursorposition  
ermitteln

int1      \_WinEditorSelGetAnchorPos      Ankerposition  
ermitteln

\_WinEditorSelGetRange      Selektionslänge  
ermitteln

int2      View-Nummer (optional)

Resultat point      Ermittelte Koordinate / Länge 

Verwandte Befehle,

Siehe WinEditorSetSelection(),

WinEditorGetSelectedText(),

WinEditorReplaceSelectedText()

Mit dieser Funktion werden im CodeEdit-Objekt (obj) Informationen über den selektierten Bereich ermittelt.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Mit dem Argument (int1) wird der zu ermittelnde Wert angegeben. Folgende Konstanten können angegeben werden:

\_WinEditorSelGetCaretPos      Cursorposition ermitteln

\_WinEditorSelGetAnchorPos      Ankerposition ermitteln

\_WinEditorSelGetRange      Selektionslänge ermitteln

In (int2) kann optional die Nummer des Views angegeben werden, in dem der Selektionsbereich ermittelt werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

## Resultat

Als Resultat wird eine Koordinate im Text zurückgegeben. Bei den Positionsangaben bestimmt die :x-Koordinate die Zeile und die :y-Koordinate die Spalte. Das erste Zeichen hat die Koordinate 1,1. Bei der Option \_WinEditorSelGetRange bestimmt die :x-Koordinate die Länge des selektierten Textes. Die :y-Koordinate ist 0.

## Beispiel:

```
// Selektionsanfang und -Ende ermitteltCaret # $CodeEdit->WinEditorGetSelection(_WinEditorSelGet
```

Mögliche Laufzeitfehler:

\_ErrHdlInvalid      Der Deskriptor des CodeEdit (obj) ist ungültig.

\_ErrValueInvalid      Ungültige oder nicht existierende View-Nummer (int2) angegeben.

obj -> WinEditorGoTo(point1[, logic2[,  
int3]])



Im CodeEdit-Objekt zur Position scrollen

obj      Deskriptor des  
         CodeEdit-Objektes

point1 Position

logic2 Nur scrollen (optional)

int3      View-Nummer  
         (optional)

Verwandte Befehle,

Siehe WinEditorSetSelection(),  
WinEditorHighlight()

Mit dieser Funktion wird im CodeEdit-Objekt (obj) zur angegebenen Position gescrollt.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Die Position wird mit (point1) definiert.



Bei der Positionsangaben bestimmt die :x-Koordinate die Zeile und die :y-Koordinate die Spalte. Das erste Zeichen hat die Koordinate 1,1.

Wird im optionalen Argument (logic2) true (Default) angegeben, wird nur zu den Koordinaten gescrollt. Durch Angabe von false wird zusätzlich die Cursor-Position auf die angegebene Position gesetzt.

In (int3) kann optional die Nummer des Views angegeben werden, in dem selektiert werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

### Beispiel:

```
// Zu Zeile 4, Spalte 2 scrollen$CodeEdit->WinEditorGoTo(PointMake(2, 4));// Cursorposition in Z
```

Mögliche Laufzeitfehler:

\_ErrHdlInvalid

Der Deskriptor des CodeEdit (obj) ist ungültig.

\_ErrValueInvalid

Ungültige oder nicht existierende View-Nummer (int3) angegeben.

obj -> WinEditorHighlight(point1[,  
logic2[, point3[, int4]]) : rect  
Text im CodeEdit-Objekt hervorheben



obj     Deskriptor des  
       CodeEdit-Objektes  
point1 Startposition  
logic2 Hervorhebung löschen  
       (optional)  
point3 Endposition (optional)  
int4    View-Nummer (optional)  
Siehe Verwandte Befehle,  
       WinEditorSetSelection()

Mit dieser Funktion kann ein Bereich im CodeEdit-Objekt (obj) hervorgehoben werden. Die Hervorhebung findet in der Farbe ThemeColEditorCustomHighlight statt.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Die Startposition wird mit (point1) definiert.

Die Endposition wird mit (point3) angegeben. Wird (point3) nicht angegeben, geht die Hervorhebung bis zum Ende des Textes.



Bei den Positionsangaben bestimmt die :x-Koordinate die Zeile und die :y-Koordinate die Spalte. Das erste Zeichen hat die Koordinate 1,1. Negative :x-Koordinaten werden auf das Ende des Textes gesetzt.

Parameter (logic2) definiert, ob eine bisher vorhandene Hervorhebung entfernt werden soll. Wird der Parameter nicht angegeben, wird er auf true gesetzt.

In (int4) kann optional die Nummer des Views angegeben werden, in dem der Selektionsbereich ermittelt werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo( \_WinCount) ermittelt werden.



Die Hervorhebung wird für alle Views übernommen, bei denen die Eigenschaften FileName und EditorTextType auf den gleichen Wert gesetzt sind.

### Beispiel:

```
// Bereich hervorheben$CodeEdit->WinEditorHighlight(PointMake(5, 5), false, PointMake(5, 10));//
```

Mögliche Laufzeitfehler:

ErrHdlInvalid    Der Deskriptor des CodeEdit (obj) ist ungültig.

ErrValueInvalid Ungültige oder nicht existierende View-Nummer (int2) angegeben.



WinEditorKeywordsAdd(alpha1, int2) : int  
Selbstdefinierte Schlüsselwörter  
zu den CodeEdit-Objekten  
hinzufügen

alpha1 Schlüsselwörter mit '|' (Pipe) getrennt

Typ der Schlüsselwörter

WinEditorKeyDatatype Datentyp

WinEditorKeyKeyword Schlüsselwort

int2

WinEditorKeyFunction Funktion

WinEditorKeyConstant Konstante



WinEditorKeyCustom kundenspezifisch

Fehlercode

Resultat int ErrOk Kein Fehler aufgetreten

ErrUnavailable Kein CodeEdit-Objekt erzeugt

Siehe Verwandte Befehle, WinEditorKeywordsRemove(),  
WinEditorKeywordsUpdate()

Mit dieser Funktion können eigene Schlüsselwörter zu allen CodeEdit-Objekten hinzugefügt werden. Damit diese im CodeEdit-Objekt hervorgehoben werden, muss die Eigenschaft EditorTextType auf WinEditorTextTypePrc gesetzt sein. Hinzugefügte Schlüsselwörter werden zusätzlich in die Auto-Vervollständigungsliste ( + ) aufgenommen.



Diese Methode kann frühestens im EvtCreated des Elternfensters des ersten CodeEdit-Objektes verwendet werden.



Die Liste der Schlüsselwörter wird automatisch zurückgesetzt, wenn die letzte laufende Prozedur verlassen wird (Prozedur nach Start oder Testprozedur im Standard-Client bzw. Rückkehr in den Designer).



Die Schlüsselwörter werden nicht auf vorhandensein geprüft. Wird ein Schlüsselwort mehrfach hinzugefügt, erscheint es mehrfach in der Auto-Vervollständigungsliste.

Im Argument (alpha1) müssen ein oder mehrere hinzuzufügende Schlüsselwörter angegeben werden. Mehrere Schlüsselwörter werden durch '|' (Pipe) getrennt.



Das erste Zeichen der Schlüsselwörter muss ein Unterstrich oder ein Buchstabe sein. Ab dem zweiten Zeichen dürfen zusätzlich Punkte und Ziffern verwendet werden.

In (int2) wird der Typ der Schlüsselwörter angegeben. Der entsprechende Typ definiert die Hervorhebung im CodeEdit-Objekt. Folgende Typen können angegeben werden:

Konstante	Typ	Stil
<u>WinEditorKeyDatatype</u>	Datentyp	Fett mit Farbe <u>ThemeColEditorDataTypes</u>
<u>WinEditorKeyKeyword</u>	Schlüsselwort	Fett mit Farbe <u>ThemeColEditorKeywords</u>
<u>WinEditorKeyFunction</u>	Funktion	Farbe <u>ThemeColEditorFunctions</u>
<u>WinEditorKeyConstant</u>	Konstante	Kursiv mit Farbe <u>ThemeColEditorConstants</u>
<u>WinEditorKeyCustom</u>	kundenspezifisch	Fett mit Farbe <u>ThemeColEditorCKeywords</u>

**Resultat**

Das Resultat ist \_\_ErrOk, wenn das Hinzufügen der Schlüsselwörter erfolgreich war. Ist noch kein CodeEdit-Objekt fertig initialisiert, wird \_\_ErrUnavailable zurückgegeben.




Die Liste der Schlüsselwörter muss für jedes CodeEdit-Objekt separat mittels WinEditorKeywordsUpdate() aktualisiert werden. Diese Notwendigkeit entfällt, wenn die Schlüsselwörter hinzugefügt wurden, bevor eine Datei im CodeEdit geöffnet wird.

**Beispiel:**

```
// Schlüsselwörter hinzufügenErr # WinEditorKeywordsAdd('MyFunction1|MyFunction2', _WinEditorKey
```

Mögliche Laufzeitfehler:

\_\_ErrValueInvalid Leeres Schlüsselwort oder ungültiges Zeichen im Schlüsselwort (alpha1) oder ungültiger Typ (int2) angegeben.

WinEditorKeywordsRemove([int1[, alpha2]]) : int   
 Schlüsselwörter der CodeEdit-Objekte entfernen  
 int1      Modus (optional)  
 alpha2    Zu entfernende Schlüsselwörter


Fehlercode

Resultat int ErrOk      Kein Fehler aufgetreten 

ErrUnavailable Kein CodeEdit-Objekt erzeugt

Siehe Verwandte Befehle, WinEditorKeywordsAdd(),  
WinEditorKeywordsUpdate()

Mit dieser Funktion werden die Schlüsselwörter aller CodeEdit-Objekte zurückgesetzt. Damit werden alle mit WinEditorKeywordsAdd() hinzugefügten Wörter nicht mehr hervorgehoben.

 Diese Methode kann frühestens im EvtCreated des Elternfensters des ersten CodeEdit-Objektes verwendet werden.

Mit dem optionalen Argument (int1) kann definiert werden, welche Schlüsselwörter entfernt werden. Folgende Konstanten können angegeben werden:


WinEditorKeyRemoveCustom Entfernt alle Schlüsselwörter, die mit  
WinEditorKeywordsAdd() hinzugefügt wurden  
 (Default)

WinEditorKeyRemoveAll Entfernt alle Schlüsselwörter, inklusive der  
 CONZEPT 16 Schlüsselwörter

WinEditorKeyRemoveList Entfernt die in (alpha2)angegebenen Schlüsselwörter  
 In dem optionalen Argument (alpha2) werden die zu entfernenden Schlüsselwörter bei dem Modus WinEditorKeyRemoveList angegeben. Mehrere Schlüsselwörter werden durch '|' (Pipe) getrennt.

## Resultat

Das Resultat ist ErrOk, wenn das Löschen der Schlüsselwörter erfolgreich war. Ist noch kein CodeEdit-Objekt fertig initialisiert, wird ErrUnavailable zurückgegeben.

 Die Liste der Schlüsselwörter muss für jedes CodeEdit-Objekt separat mittels WinEditorKeywordsUpdate() aktualisiert werden. Diese Notwendigkeit entfällt, wenn die Schlüsselwörter zurückgesetzt wurden, bevor eine Datei im CodeEdit geöffnet wird.

## Beispiel:

```
// Selbst hinzugefügte Schlüsselwörter entfernenErr # WinEditorKeywordsRemove();// CodeEdit aktu
```

obj -> WinEditorKeywordsUpdate([int1]) : int  
 Schlüsselwörter im CodeEdit-Objekt aktualisieren  
 obj Deskriptor des CodeEdit-Objektes  
 int4 View-Nummer (optional)



Fehlercode

Resultat logic ErrOk Kein Fehler aufgetreten  
ErrUnavailable EditorTextType !=  
WinEditorTextTypePrc



Siehe Verwandte Befehle, WinEditorKeywordsAdd(),  
WinEditorKeywordsRemove()

Mit dieser Funktion werden die mittels WinEditorKeywordsAdd() hinzugefügten bzw. mit WinEditorKeywordsRemove() entfernten Schlüsselwörter im CodeEdit-Objekt (obj) angewendet. Die Eigenschaft EditorTextType muss hierzu auf WinEditorTextTypePrc gesetzt sein.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

In (int1) kann optional die Nummer des Views angegeben werden, in dem die Schlüsselwörter aktualisiert werden sollen. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) bewirkt eine Aktualisierung in allen Views. Die Anzahl der Views kann mit \$CodeEdit->WinInfo( \_WinCount) ermittelt werden.

## Resultat



Das Resultat ist ErrOk, wenn die Schlüsselwörter erfolgreich aktualisiert wurden. Ist die Eigenschaft EditorTextType nicht auf WinEditorTextTypePrc gesetzt, wird ErrUnavailable zurückgegeben.

## Beispiel:

```
// Schlüsselwörter hinzufügenWinEditorKeywordsAdd('MyFunction1|MyFunction2', _WinEditorKeyFunction1)
```

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Deskriptor des CodeEdit (obj) ist ungültig oder das View mit der Nummer (int1) ist nicht vorhanden.

obj -> WinEditorLoad([alpha1[,  int2[, int3[, int4]]]]) : int  
 Text im CodeEdit-Objekt laden  
 obj      Deskriptor des CodeEdit-Objektes  
 alpha1   Name der Prozedur / des Textes  
           (optional)  
           Quelle (optional)  
           \_WinStreamNameText Interne(n)  
                                  Prozedur / Text  
 int2                                   laden  
           \_WinStreamNameFile Externen Text  
                                  laden  
           Typ (optional)  
           \_WinEditorTextTypeAuto Texttyp anhand  
                                  der Dateieindung  
                                  ermitteln  
           \_WinEditorTextTypePrc Text als  
                                  Prozedur laden  
           \_WinEditorTextTypeText Text als  
 int3                                   normalen Text  
                                  laden  
           \_WinEditorTextTypeXml Text als  
                                  XML-Dokument  
                                  laden  
           \_WinEditorTextTypeJson Text als  
                                  JSON-Dokument  
                                  laden  
 int4      View-Nummer (optional)  
           Fehlercode  
           \_rOk              Speichern erfolgreich  
           \_rNoRec        Prozedur / Text  
                                  (alpha1)nicht  
 Resultat int               vorhanden   
           \_rNoRights Benutzerberechtigung  
                                  nicht ausreichend  
           \_ErrFsi... Fehler für externe  
                                  Dateioperationen

Siehe Verwandte Befehle, WinEditorSave()

Mit dieser Funktion wird der Inhalt einer Prozedur / eines Textes in das CodeEdit-Objekt (obj) geladen.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Optional kann in (alpha1) der Name angegeben werden. Ist der Text bisher noch nicht geladen, muss hier ein Name definiert werden.

Als Quelle (int2) können folgende Konstanten angegeben werden:



## Kontakt

WinStreamNameText Interne(n) Prozedur / Text laden

WinStreamNameFile Externen Text laden

Wird das Argument (int2) nicht angegeben, wird die Quelle anhand des Namens (alpha1) bzw. der Eigenschaft FileName ermittelt.

Folgende Typen (int3) können angegeben werden:

WinEditorTextTypeAuto Texttyp anhand der Dateiendung ermitteln

WinEditorTextTypePrc Text als Prozedur laden

WinEditorTextTypeText Text als normalen Text laden

WinEditorTextTypeXml Text als XML-Dokument laden

WinEditorTextTypeJson Text als JSON-Dokument laden

In (int4) kann optional die Nummer des Views angegeben werden, in dem der Text geladen werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. Wird als View-Nummer 0 (oder nicht angegeben) angegeben, werden alle dargestellten Texte geladen. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

## Resultat

Das Resultat ist ErrOk, wenn alle Texte geladen werden konnten. Neben den Fehlerwerten für externe Dateioperationen kann für interne Dokumente einer der folgenden Fehlerwerte zurückgegeben werden:

rNoRec Prozedur / Text (alpha1) nicht vorhanden

rNoRights Benutzerberechtigung nicht ausreichend.

## Beispiel:

```
// Geladenes Dokument mit gleichen Einstellungen erneut ladentResult # $CodeEdit->WinEditorLoad()
```

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Deskriptor des CodeEdit (obj) ist ungültig.

ErrValueInvalid Ungültige Quelle (int2), ungültiger Typ (int3) oder ungültige bzw. nicht existierende View-Nummer (int4) angegeben.

obj -> WinEditorReplaceSelectedText(alpha1[,  
handle2[, int3]])



Selektierten Text im CodeEdit-Objekt ersetzen

obj      Deskriptor des  
          CodeEdit-Objektes  
alpha1   Ersetzungstext  
handle2   Deskriptor eines  
          Memory-Objektes (optional)  
int3      View-Nummer (optional)  
          Verwandte Befehle,  
Siehe     WinEditorGetSelection(),  
          WinEditorGetSelectedText()

Mit dieser Funktion wird im CodeEdit-Objekt (obj) der selektierte Text ersetzt. Wird ein Memory-Objekt (handle2) wird der selektierte Text den den Inhalt dieses ersetzt. Andernfalls wird der Ersetzungstext (alpha1) verwendet.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

In (int3) kann optional die Nummer des Views angegeben werden, in dem der selektierte Text ersetzt werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

### Beispiel:

```
// Selektierten Text ersetzen $CodeEdit->WinEditorReplaceSelectedText('replaced content');
```

Mögliche Laufzeitfehler:

\_ErrHdlInvalid      Der Deskriptor des CodeEdit (obj) oder des Memory-Objektes (handle2) ist ungültig.  
\_ErrValueInvalid   Ungültige oder nicht existierende View-Nummer (int3) angegeben.





Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Optional kann in (alpha1) der Name angegeben werden. Ist der Text bisher noch nicht gespeichert, muss hier ein Name definiert werden.

Soll ein bestehendes internes Dokument überschrieben werden, das zuvor nicht geladen wurde, muss die Option (int4) \_WinEditorSaveOverwrite angegeben werden.

Als Ziel (int2) können folgende Konstanten angegeben werden:

\_WinStreamNameText Interne(n) Prozedur / Text laden

\_WinStreamNameFile Externen Text laden

Wird das Argument (int2) nicht angegeben, wird das Ziel die Quelle aus WinEditorLoad() verwendet.

Folgende Typen (int3) können angegeben werden:

\_WinEditorTextTypeAuto Texttyp beibehalten

\_WinEditorTextTypePrc Text als Prozedur speichern

\_WinEditorTextTypeText Text als normalen Text speichern

\_WinEditorTextTypeXml Text als XML-Dokument speichern

\_WinEditorTextTypeJson Text als JSON-Dokument speichern

In (int5) kann optional die Nummer des Views angegeben werden, in dem der Text gespeichert werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. Wird als View-Nummer 0 (oder nicht angegeben) angegeben, werden alle dargestellten Texte gespeichert. Die Anzahl der Views kann mit \$CodeEdit->WinInfo( \_WinCount) ermittelt werden.

## Resultat

Das Resultat ist \_ErrOk, wenn alle Texte gespeichert werden konnten. Neben den Fehlerwerten für externe Dateioperationen kann für interne Dokumente einer der folgenden Fehlerwerte zurückgegeben werden:

\_rLocked Text / Prozedur (alpha1) gesperrt.

\_rExists Text / Prozedur (alpha1) existiert bereits. Option (int4)

\_WinEditorSaveOverwrite muss zum Überschreiben angegeben werden.

\_rNoRights Benutzerberechtigung nicht ausreichend.

\_rDeadlock Verklemmung aufgetreten.

## Beispiel:

```
// Geladenes Dokument mit gleichen Einstellungen speichertResult # $CodeEdit->WinEditorSave();//
```

Mögliche Laufzeitfehler:

\_ErrHdlInvalid Der Deskriptor des CodeEdit (obj) ist ungültig.

\_ErrValueInvalid Ungültiges Ziel (int2), ungültiger Typ (int3), ungültige Optionen (int4) oder ungültige bzw. nicht existierende View-Nummer (int5)

Kontakt

angegeben.

## Kontakt

obj -> WinEditorSearch(alpha1[, int2[, point3[,  
alpha4[, int5]]]]) : rect

Text im CodeEdit-Objekt suchen und ersetzen

obj Deskriptor des CodeEdit-Objektes

alpha1 Suchbegriff

Optionen (optional)

WinEditorSearchRev

Rückwärts suchen

WinEditorSearchInSel

Im selektierten  
Bereich suchen

WinEditorSearchGoTo

Zu Suchtreffer  
springen

WinEditorSearchRegExp

Regulären Ausdruck  
suchen

WinEditorSearchWholeWord

Ganzes Wort suchen

int2 WinEditorSearchCI

Groß-/Kleinschreibung  
ignorieren

WinEditorSearchHighlightWord

Aktuellen Suchtreffer  
hervorheben

WinEditorSearchHighlightOccurrences

Andere Suchtreffer  
hervorheben

WinEditorSearchReplace

Aktuellen Suchtreffer  
ersetzen

WinEditorSearchReplaceAll

Alle Suchtreffer  
ersetzen

point3 Startposition (optional)

alpha4 Ersetzungsbegriff (optional)

int5 View-Nummer (optional)

Resultat rect Position des Suchtreffers



Siehe Verwandte Befehle, WinEditorGetSelection()

Mit dieser Funktion wird der Inhalt des CodeEdit-Objektes (obj) nach der Zeichenfolge (alpha1) durchsucht.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Als Suchoption (int2) können optional folgende Konstanten angegeben werden:

WinEditorSearchRev

Rückwärts suchen

WinEditorSearchInSel

Im selektierten Bereich suchen

WinEditorSearchGoTo

Zu Suchtreffer springen

WinEditorSearchRegExp

Regulären Ausdruck suchen

WinEditorSearchWholeWord

Ganzes Wort suchen

WinEditorSearchCI

Groß-/Kleinschreibung ignorieren

WinEditorSearchHighlightWord

Aktuellen Suchtreffer hervorheben

WinEditorSearchHighlightOccurrences

Andere Suchtreffer hervorheben


WinEditorSearchReplace

Aktuellen Suchtreffer ersetzen

WinEditorSearchReplaceAll Alle Suchtreffer ersetzen  
Die Optionen können miteinander kombiniert werden.

Mit den Optionen WinEditorSearchReplace und WinEditorSearchReplaceAll werden die Suchtreffer durch den Ersetzungsbegriff (alpha4) ersetzt. Ist dieser leer, werden die Suchtreffer entfernt.

Die Startposition kann optional mit (point3) definiert werden.

 Bei den Positionsangaben bestimmt die :x-Koordinate die Zeile und die :y-Koordinate die Spalte. Das erste Zeichen hat die Koordinate 1,1. Wird der Punkt -1,-1 angegeben, wird die Positionsangabe ignoriert. Abhängig davon, ob die Suchoption WinEditorSearchRev angegeben ist, wird vom Anfang oder vom Ende des Textes gesucht.

In (int5) kann optional die Nummer des Views angegeben werden, in dem gesucht werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden. View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo( \_WinCount) ermittelt werden.

## Resultat

Als Resultat wird die Position des Suchtreffers zurückgegeben. Hierbei bestimmt die :left-Koordinate die Anfangszeile, die :top-Koordinate die Anfangsspalte, die :right-Koordinate die Endzeile und die :bottom-Koordinate die Endspalte des Treffers.

## Beispiel:

```
// Wort suchen ohne MarkierungtRect # $CodeEdit->WinEditorSearch('Hallo');// Wort ab bestimmter P
```

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Deskriptor des CodeEdit (obj) ist ungültig.  
ErrValueInvalid Ungültige Option (int2) oder ungültige oder nicht existierende View-Nummer (int5) angegeben.



obj -> WinEditorSetSelection(point1[, point2[, int3[, int4]])  
 Selektionsbereich im CodeEdit-Objekt setzen oder hinzufügen

obj Deskriptor des CodeEdit-Objektes

point1 Startpunkt / Cursor-Position

point2 Endpunkt (optional)

Optionen (optional)

WinEditorSelSetDefault Normale Selektion  
 setzen (Default)

int3 WinEditorSelSetBlock Blockselektion  
 setzen

WinEditorSelSetMulti Normalen  
 Selektionsbereich  
 hinzufügen

int4 View-Nummer (optional)

Verwandte Befehle,

Siehe WinEditorGetSelection(),

WinEditorHighlight(), WinEditorGoTo()

Mit dieser Funktion wird im CodeEdit-Objekt (obj) ein Bereich selektiert.



Diese Methode kann frühestens im EvtCreated des Elternfensters verwendet werden.

Die Startposition wird mit (point1) definiert.

Die Endposition wird mit (point2) angegeben. Wird (point2) nicht angegeben, wird nur der Cursor an die Startposition (point1) gesetzt.



Bei den Positionsangaben bestimmt die :x-Koordinate die Zeile und die :y-Koordinate die Spalte. Das erste Zeichen hat die Koordinate 1,1. Negative :x-Koordinaten werden auf das Ende des Textes gesetzt.

Mit dem Argument (int3) kann optional der Typ der Markierung definiert werden. Folgende Konstanten können angegeben werden:

WinEditorSelSetDefault Normale Selektion setzen (Default)

WinEditorSelSetBlock Blockselektion setzen

WinEditorSelSetMulti Normalen Selektionsbereich hinzufügen

In (int4) kann optional die Nummer des Views angegeben werden, in dem selektiert werden soll. Die Views können mit den Nummern 1 bis 4 angesprochen werden.

View-Nummer 0 (oder nicht angegeben) ist gleichbedeutend mit 1. Die Anzahl der Views kann mit \$CodeEdit->WinInfo(\_WinCount) ermittelt werden.

### Beispiel:

```
// Kompletten Text markieren$CodeEdit->WinEditorSetSelection(PointMake(1, 1), PointMake(-1, 0));
```

Mögliche Laufzeitfehler:

ErrHdlInvalid Der Deskriptor des CodeEdit (obj) ist ungültig.

ErrValueInvalid



## Kontakt

Ungültige Option (int3) oder ungültige oder nicht existierende View-Nummer (int4) angegeben.

Konstanten für CodeEdit-Befehle  
Konstanten für CodeEdit-Befehle  
Siehe CodeEdit-Befehle

- WinEditorKeyConstant
- WinEditorKeyCustom
- WinEditorKeyDatatype
- WinEditorKeyFunction
- WinEditorKeyKeyword
- WinEditorKeyRemoveAll
- WinEditorKeyRemoveCustom
- WinEditorKeyRemoveList
- WinEditorSaveOverwrite
- WinEditorSearchCI
- WinEditorSearchGoTo
- WinEditorSearchHighlightOccurrences
- WinEditorSearchHighlightWord
- WinEditorSearchInSel
- WinEditorSearchRegExp
- WinEditorSearchReplace
- WinEditorSearchReplaceAll
- WinEditorSearchRev
- WinEditorSearchWholeWord
- WinEditorSelGetAnchorPos
- WinEditorSelGetCaretPos
- WinEditorSelGetRange
- WinEditorSelSetBlock
- WinEditorSelSetDefault
- WinEditorSelSetMulti
- WinEditorTextTypeAuto
- WinEditorTextTypeJson
- WinEditorTextTypePrc
- WinEditorTextTypeText
- WinEditorTextTypeXml

\_WinEditorKeyConstant

Schlüsselwort ist eine Konstante

Wert 3

Siehe Verwandte Befehle,

WinEditorKeywordsAdd()

Option bei WinEditorKeywordsAdd(), durch die die hinzugefügten Schlüsselwörter als Konstante (Kursiv mit Farbe ThemeColEditorConstants) dargestellt werden.

\_WinEditorKeyCustom

Schlüsselwort ist kundenspezifisch

Wert 5

Siehe Verwandte Befehle,

WinEditorKeywordsAdd()

Option bei WinEditorKeywordsAdd(), durch die die hinzugefügten Schlüsselwörter als kundenspezifisches Schlüsselwort (Fett mit Farbe ThemeColEditorCKeywords) dargestellt werden.

\_WinEditorKeyDatatype

Schlüsselwort ist ein Datentyp

Wert 0

Siehe Verwandte Befehle,

WinEditorKeywordsAdd()

Option bei WinEditorKeywordsAdd(), durch die die hinzugefügten Schlüsselwörter als Datentyp (Fett mit Farbe ThemeColEditorDataTypes) dargestellt werden.

\_WinEditorKeyFunction

Schlüsselwort ist eine Funktion

Wert 2

Siehe Verwandte Befehle,

WinEditorKeywordsAdd()

Option bei WinEditorKeywordsAdd(), durch die die hinzugefügten Schlüsselwörter als Funktion (Farbe ThemeColEditorFunctions) dargestellt werden.

\_WinEditorKeyKeyword

Schlüsselwort ist ein Schlüsselwort

Wert 1

Siehe Verwandte Befehle,

WinEditorKeywordsAdd()

Option bei WinEditorKeywordsAdd(), durch die die hinzugefügten Schlüsselwörter als Schlüsselwort (Fett mit Farbe ThemeColEditorKeywords) dargestellt werden.

\_WinEditorKeyRemoveAll

Alle Schlüsselwörter entfernen

Wert 1

Siehe WinEditorKeywordsRemove()

Option bei WinEditorKeywordsRemove(), durch die alle Schlüsselwörter entfernt werden können.



\_WinEditorKeyRemoveCustom

Selbst hinzugefügte Schlüsselwörter entfernen

Wert 0

Siehe WinEditorKeywordsRemove()

Option bei WinEditorKeywordsRemove(), durch die alle mit WinEditorKeywordsAdd() hinzugefügten Schlüsselwörter entfernt werden können.

\_WinEditorKeyRemoveList

Ausgewählte Schlüsselwörter entfernen

Wert 2

Siehe WinEditorKeywordsRemove()

Option bei WinEditorKeywordsRemove(), durch die ausgewählte Schlüsselwörter entfernt werden können.

WinEditorSaveOverwrite  
Internes Dokument überschreiben  
Wert 1 / 0x0001

Verwandte

Siehe Befehle,

WinEditorSave()

Option bei WinEditorSave() durch die ein bestehendes Dokument, welches nicht aktuell im CodeEdit-Objekt dargestellt wird, überschrieben werden kann.

\_WinEditorSearchCI  
Groß-/Kleinschreibung ignorieren  
Wert 32 / 0x0020

Siehe Verwandte Befehle,  
WinEditorSearch()

Option bei WinEditorSearch() durch die die Groß-/Kleinschreibung beim Suchen ignoriert werden kann.

\_WinEditorSearchGoTo

In Trefferzeile springen

Wert 4 / 0x0004

Siehe Verwandte Befehle,

WinEditorSearch()

Option bei WinEditorSearch() durch die in die Zeile des Suchtreffers gesprungen wird.

WinEditorSearchHighlightOccurences

Weitere Suchtreffer hervorheben

Wert 128 / 0x0080

Siehe Verwandte Befehle,

WinEditorSearch()

Option bei WinEditorSearch() durch die alle weiteren außer dem aktuellen Suchtreffer hervorgehoben werden.

WinEditorSearchHighlightWord  
Aktuellen Suchtreffer hervorheben  
Wert 64 / 0x0040

Siehe Verwandte Befehle,  
WinEditorSearch()

Option bei WinEditorSearch() durch die der aktuelle Suchtreffer hervorgehoben wird.

WinEditorSearchInSel

In selektiertem Bereich suchen

Wert 2 / 0x0002

Siehe Verwandte Befehle,  
WinEditorSearch()

Option bei WinEditorSearch() durch die nur im selektierten Bereich gesucht wird.



WinEditorSearchRegExp  
Regulären Ausdruck verwenden  
Wert 8 / 0x0008

Siehe Verwandte Befehle,  
WinEditorSearch()

Option bei WinEditorSearch() durch die der Suchbegriff als regulärer Ausdruck interpretiert wird.

WinEditorSearchReplace  
Aktuellen Suchtreffer ersetzen  
Wert 256 / 0x0100

Siehe Verwandte Befehle,  
WinEditorSearch()

Option bei WinEditorSearch() durch die der aktuelle Suchtreffer ersetzt wird.

WinEditorSearchReplaceAll

Alle Suchtreffer ersetzen

Wert 512 / 0x0200

Siehe Verwandte Befehle,

WinEditorSearch()

Option bei WinEditorSearch() durch die alle Suchtreffer ersetzt werden.

WinEditorSearchRev

Rückwärts suchen

Wert 1 / 0x0001

Siehe Verwandte Befehle,  
WinEditorSearch()

Option bei WinEditorSearch() durch die rückwärts gesucht wird.

WinEditorSearchWholeWord  
Nach ganzen Wörtern suchen  
Wert 16 / 0x0010

Siehe Verwandte Befehle,  
WinEditorSearch()

Option bei WinEditorSearch() durch die die Suchergebnisse auf ganze Wörter beschränkt werden kann.

Der Suchbegriff wird nur dann gefunden, wenn vor und nach ihm ein Worttrennzeichen steht. Worttrennzeichen sind alle Zeichen mit Ausnahme von Buchstaben oder Zahlen. Das Zeilenende trennt ebenfalls Wörter voneinander.

WinEditorSelGetAnchorPos  
Ankerposition aus Selektion ermitteln  
Wert 1

WinEditorGetSelection(),  
Siehe WinEditorSelGetCaretPos,  
WinEditorSelGetRange

Option bei WinEditorGetSelection(), mit der die Ankerposition der aktuellen Selektion im CodeEdit-Objekt ermittelt wird.

WinEditorSelGetCaretPos  
Cursorposition aus Selektion ermitteln  
Wert 0

WinEditorGetSelection(),  
Siehe WinEditorSelGetAnchorPos,  
WinEditorSelGetRange

Option bei WinEditorGetSelection(), mit der die Cursorposition der aktuellen Selektion im CodeEdit-Objekt ermittelt wird.

WinEditorSelGetRange

Länge der Selektion ermitteln

Wert 2

WinEditorGetSelection(),

Siehe WinEditorSelGetCaretPos,

WinEditorSelGetAnchorPos

Option bei WinEditorGetSelection(), mit der die Länge der aktuellen Selektion im CodeEdit-Objekt ermittelt wird.



WinEditorSelSetBlock

Text mit Blockseletion selektieren

Wert 1

WinEditorSetSelection(),

Siehe WinEditorSelSetDefault,

WinEditorSelSetMulti

Option bei WinEditorSetSelection(), mit der eine Blockselektion gesetzt wird.

WinEditorSelSetDefault

Text selektieren

Wert 0

WinEditorSetSelection(),

Siehe WinEditorSelSetBlock,

WinEditorSelSetMulti

Option bei WinEditorSetSelection(), mit der eine normale Selektion gesetzt wird.

WinEditorSelSetMulti

Selektion hinzufügen

Wert 2

WinEditorSetSelection(),

Siehe WinEditorSelSetDefault,

WinEditorSelSetBlock

Option bei WinEditorSetSelection(), mit die bestehenden Selektionen erhalten bleiben und ein weiterer Bereich selektiert wird.

WinEditorTextTypeAuto  
Texttyp ermitteln oder beibehalten  
Wert -1

Siehe WinEditorLoad(),  
WinEditorSave()

Diese Option kann bei den Befehlen WinEditorLoad() und WinEditorSave() angegeben werden, um den Typ des zu ladenden Dokumentes automatisch zu ermitteln bzw. zu speichernden Dokumentes beizubehalten. Beim Laden wird automatisch die Eigenschaft EditorTextType auf den ermittelten Typ gesetzt.

\_WinEditorTextTypeJson  
Darzustellender Text ist ein JSON-Dokument  
Wert 3

EditorTextType,  
Siehe WinEditorLoad(),  
WinEditorSave()

Wird die Eigenschaft EditorTextType auf den Wert \_WinEditorTextTypeJson gesetzt,  
wird im CodeEdit-Objekt ein JSON-Dokument angezeigt.

Diese Option kann zusätzlich bei den Befehlen WinEditorLoad() und WinEditorSave()  
angegeben werden, um den Typ des zu ladenden bzw. zu speichernden Dokumentes  
festzulegen. Hierbei wird automatisch die Eigenschaft EditorTextType gesetzt.

\_WinEditorTextTypePrc

Darzustellender Text ist eine Prozedur

Wert 0

EditorTextType,

Siehe WinEditorLoad(),

WinEditorSave()

Wird die Eigenschaft EditorTextType auf den Wert \_WinEditorTextTypePrc gesetzt, wird im CodeEdit-Objekt eine Prozedur angezeigt.

Diese Option kann zusätzlich bei den Befehlen WinEditorLoad() und WinEditorSave() angegeben werden, um den Typ des zu ladenden bzw. zu speichernden Dokumentes festzulegen. Hierbei wird automatisch die Eigenschaft EditorTextType gesetzt.

\_WinEditorTextTypeText  
Darzustellender Text ist ein Text  
Wert 1

EditorTextType,  
Siehe WinEditorLoad(),  
WinEditorSave()

Wird die Eigenschaft EditorTextType auf den Wert \_WinEditorTextTypeText gesetzt, wird im CodeEdit-Objekt ein Text angezeigt.

Diese Option kann zusätzlich bei den Befehlen WinEditorLoad() und WinEditorSave() angegeben werden, um den Typ des zu ladenden bzw. zu speichernden Dokumentes festzulegen. Hierbei wird automatisch die Eigenschaft EditorTextType gesetzt.

\_WinEditorTextTypeXml

Darzustellender Text ist ein XML-Dokument

Wert 2

EditorTextType,

Siehe WinEditorLoad(),

WinEditorSave()

Wird die Eigenschaft EditorTextType auf den Wert \_WinEditorTextTypeXml gesetzt, wird im CodeEdit-Objekt ein XML-Dokument angezeigt.

Diese Option kann zusätzlich bei den Befehlen WinEditorLoad() und WinEditorSave() angegeben werden, um den Typ des zu ladenden bzw. zu speichernden Dokumentes festzulegen. Hierbei wird automatisch die Eigenschaft EditorTextType gesetzt.



CtxDocEdit-Befehle

Befehle und Konstanten für CtxDocEdit-Objekt

Verwandte

Befehle,

Siehe CtxDocEdit,

Befehlsgruppen,

Befehlsliste

## **Befehle**

- WinDocLoadBin
- WinDocLoadName
- WinDocPrint
- WinDocSaveBin
- WinDocSaveName
- WinDocUserDictAddName
- WinDocUserDictRemoveName

## **Konstanten**

- \_WinDocLoadAscii
- \_WinDocLoadAuto
- \_WinDocLoadDoc
- \_WinDocLoadDocX
- \_WinDocLoadHtml
- \_WinDocLoadInsert
- \_WinDocLoadMix
- \_WinDocLoadOEM
- \_WinDocLoadRtf
- \_WinDocSaveAscii
- \_WinDocSaveAuto
- \_WinDocSaveDoc
- \_WinDocSaveDocX
- \_WinDocSaveHtml
- \_WinDocSaveMark
- \_WinDocSaveMix
- \_WinDocSaveOEM
- \_WinDocSavePdf
- \_WinDocSaveRtf
- \_WinStreamNameFile
- \_WinStreamNameText

obj -> WinDocLoadBin(handle1, int2[,  
alpha3]) : int



Binäres Objekt in CtxDocEdit-Objekt laden

obj Objekt (CtxDocEdit-Objekt)

handle1 Deskriptor des binären Objekts

Modus

WinDocLoadAscii ASCII-Text  
laden

WinDocLoadRtf RTF-Text  
laden

WinDocLoadHtml HTML-Format  
laden

WinDocLoadDoc Word-Dateien  
im .doc  
Format laden

int2 WinDocLoadDocX Word-Dateien  
im .docx  
Format laden

WinDocLoadOEM OEM-Text  
laden

WinDocLoadMix Text mit  
Daten  
mischen

WinDocLoadInsert Text in den  
bestehenden  
Text einfügen

alpha3 Verschlüsselungscode (optional)

Resultat int Fehlerwert

Verwandte Befehle,

Siehe WinDocLoadName(),

WinDocSaveBin(), BinOpen()

Mit diesem Befehl wird der Inhalt eines binären Objekts in ein CtxDocEdit-Objekt geladen. Der Deskriptor des CtxDocEdit-Objektes wird in (obj), der Deskriptor des binären Objekts in (handle1) übergeben.

Der Parameter (int2) bestimmt das zu lesende Format. Folgende Konstanten können angegeben werden:

- **WinDocLoadAscii**

Laden von Text im ASCII-Format.

- **WinDocLoadRtf**

Laden von Text im RTF-Format.

- **WinDocLoadHtml**

Laden von Text im HTML-Format.

- **WinDocLoadDoc**

Word-Dateien im .doc Format laden.

- **WinDocLoadDocX**

Word-Dateien im .docx Format laden.

- **WinDocLoadOEM**

Laden von Text im OEM-Format.

- **WinDocLoadMix**

Text mit Daten mischen.

- **WinDocLoadInsert**

Der zu ladende Text wird in einen bestehenden Text eingefügt.

Die Parameter zum Quellenformat können mit WinDocLoadInsert kombiniert werden, um in einen bestehenden Text den angegebenen Text einzufügen.

Der Text ersetzt den Text, der mit der Markierung (\$ctxDocEdit->cpiSelLength) selektiert ist. Ist keine Markierung vorhanden, wird der Text an der aktuellen Cursorposition \$ctxDocEdit->cpiSelStart eingefügt.

Bei der Kombination mit der Option WinDocLoadMix werden beim Laden des Textes die Platzhalter durch die entsprechenden Daten ersetzt. Weitere Informationen befinden sich im Abschnitt Text und Daten mischen.



WinDocLoadMix kann nicht zusammen mit WinDocLoadDoc oder WinDocLoadDocX angegeben werden.

Als Rückgabewert kann neben den Fehlerkonstanten aus dem Bereich der binären Objekte der Wert ErrGeneric zurückgegeben werden, wenn ein interner Fehler aufgetreten ist. Bei der Rückgabe von ErrOk ist kein Fehler aufgetreten.



Intern wird die Funktion LoadFromMemory der Text-Control-Bibliothek aufgerufen. Dabei wird die Eigenschaft LoadSaveAttribute der Bibliothek beachtet. Die Eigenschaft kann mit \$ctxDocEdit->cpiLoadSaveAttribute gelesen und gesetzt werden. Nähere Informationen finden Sie auf der Hersteller-Seite des Moduls.

### **Beispiel:**

```
// Word-Dokument aus binärem Objekt laden $ctxDocEdit->WinDocLoadBin(tBinFileHdl, _WinDocLoadDoc);
```


Mögliche Laufzeitfehler

ErrHdlInvalid

Bei (obj) handelt es sich nicht um ein CtxDocEdit-Objekt bzw. bei (handle1) nicht um ein BLOb-Objekt.

ErrValueInvalid

Argument (int2) enthält ungültige Werte.

obj -> WinDocLoadName(int1, , int2[, alpha3]) : int

Text in CtxDocEdit-Objekt laden

obj Objekt (CtxDocEdit-Objekt)

Quelle des Textes

WinStreamNameText internen  
int1 Text laden  
WinStreamNameFile externen  
Text laden

Modus für interne und externe  
Texte

WinDocLoadAscii ASCII-Text  
laden  
WinDocLoadRtf RTF-Text  
laden  
WinDocLoadHtml HTML-Format  
laden  
WinDocLoadOEM OEM-Text  
laden  
WinDocLoadMix Text mit  
Daten  
mischen  
int2 WinDocLoadInsert Text in den  
bestehenden  
Text einfügen

zusätzlicher Modus für externe  
Texte

WinDocLoadDoc Word-Dateien  
im .doc  
Format laden  
WinDocLoadDocX Word-Dateien  
im .docx  
Format laden  
WinDocLoadAuto Textformat  
anhand der  
Dateiendung  
automatisch  
erkennen

alpha3 Name des Textes (optional)

Resultat int Fehlerwert

Verwandte Befehle,

Siehe WinDocLoadBin(),  
WinDocSaveName()

Mit diesem Befehl wird ein Text in ein CtxDocEdit-Objekt geladen. Der Deskriptor des Objektes wird in (obj) übergeben.

In (int1) wird die Quelle des zu ladenden Textes definiert. Der Name der Quelle wird in (alpha3) angegeben.

Folgende Quellen können angegeben werden:

- **\_WinStreamNameText**

Der Text steht in einem internen Text zur Verfügung. Der Name des Textes wird in (alpha3) übergeben.

- **\_WinStreamNameFile**

Der Text steht in einer externen Datei zur Verfügung. Der Name des Textes kann in (alpha3) übergeben werden. Ist er nicht angegeben, wird er der Eigenschaft FileName entnommen.

Der Parameter (int2) bestimmt das Format der Quelle. Folgende Konstanten können übergeben werden:

### **Modus für interne und externe Texte**

- **\_WinDocLoadAscii**

Laden von Text im ASCII-Format.

- **\_WinDocLoadRtf**

Laden von Text im RTF-Format.

- **\_WinDocLoadHtml**

Laden von Text im HTML-Format.

- **\_WinDocLoadOEM**

Laden von Text im OEM-Format.

- **\_WinDocLoadMix**

Text mit Daten mischen.

- **\_WinDocLoadInsert**

Der zu ladende Text wird in einen bestehenden Text eingefügt.

### **zusätzlicher Modus für externe Texte**

- **\_WinDocLoadDoc**

Word-Dateien im .doc Format laden.

- **\_WinDocLoadDocX**

Word-Dateien im .docx Format laden.

- **\_WinDocLoadAuto**

Textformat anhand der Dateiendung automatisch erkennen.

Die Parameter zum Quellenformat können mit \_WinDocLoadInsert kombiniert werden, um in einen bestehenden Text den angegebenen Text einzufügen.