Ing.-Büro B. Rath
Fröbelstraße 11 C
10405 Berlin
Fon +49 30 4053 4894
Fax +49 30 4053 4962
Mobile +49 151 2422 2966

Version: 2012.7.6.0

GAEB Textkonverter

Inhalt

Vorbemerkung

Die Aufgabe des GAEB Textkonverters ist es, Lang- und Kurztexte, sowie andere formatierte Texte (AddText etc.) aus verschiedenen Formaten in das GAEB DA XML konforme GAEB XHTML Format zu überführen, bzw. aus diesem zu konvertieren. Die so in GAEB XML konvertierten Texte können unmittelbar an den Editor übergeben werden oder von diesem als Input verwendet werden. Diese so konvertierten Texte können auch unmittelbar an die GAEB Toolbox übergeben werden.

Zur Zeit unterstützt werden folgende Formate: **RTF, ASCII, GAEB XML**

Der GAEB Textkonverter wird als DLL bereit gestellt. Er eignet sich für alle Schnittstellen Funktionen im Zusammenhang mit dem Im- und Export von GAEB DA XML Dateien, insbesondere im Zusammenhang mit der GAEB Toolbox.

Konvertierung zwischen GAEB XML und ASCII

Beim ASCII Export (Konvertieren von GXML nach ASCII) ist folgendes zu beachten:

- Bilder und Tabellen werden ignoriert. Stattdessen wird ein Hinweistext ausgegeben ("Hinweis: die vorhandene Tabelle /Graphik wurde entfernt")
- In nicht-numerierten Listen (Bulleting) wird jeder Listeneintrag mit vorangestelltem '-' geliefert (der tatsächliche Bullet Style wird nicht berücksichtigt).
- In allen Listen werden Hierarchien nicht dargestellt. Es wird als kein Tab und dergleichen eingefügt.
- Wenn die Anzahl der Zeichen beim Holen des Ascii Textes (gtcGetAsciiGetLineL/K/AL/AK/F/ML) angeben wird, versucht der Konverter, den zu holenden Text nach einem Leerzeichen oder einem Bindestrich ("-") umzubrechen. Eine Bindestrich mit unmittelbar folgender Ziffer (z.B. negative Zahlen) wird hierbei ignoriert. Ist kein Leerzeichen oder Bindestrich vorhanden, wird der Text "hart" umgebrochen (mitten im Wort). Wird ein Leerzeichen als erstes Zeichen einer Zeile gefunden wird es hier nicht berücksichtigt.

Konvertierung zwischen GAEB XML und RTF

Bei RTF Import (Konvertieren RTF nach GXML) ist folgendes zu beachten:

- Importiert wird nur, was in GAEB XML zugelassen ist. Dies betrifft Formatierungen und Elemente bzw. Objekte (Tabellen, Listen, etc.)
- Listen innerhalb Tabellen sind nicht zugelassen und werden dann in einfache Absätze (innerhalb der Zelle) gewandelt
- Bilder innerhalb Tabellen sind nicht zugelassen und werden dann ignoriert
- Bilder innerhalb von Listen sind nicht zugelassen und werden dann ignoriert
- Bilder innerhalb von Absätzen sind nicht zugelassen. Der aktuelle Absatz wird geteilt.
- Es werden nur Bildtypen importiert, die vom GAEB zugelassen sind (zur Zeit jpeg,gif,png)
- Tabellen innerhalb von Listen sind nicht zugelassen und werden dann ignoriert
- optionale Trennstriche ("\-" = optional hyphen) werden ignoriert
- non-breaking hyphens ("_") werden als normal Trennstriche ("-") eingefügt
- Die vertikale Standard Ausrichtung in Zellen unterscheidet sich. In RTF ist die TOP, in GXML ist dies Center. Eine Konvertierung erfolgt nicht.

Bei RTF Export (Konvertieren GXML nach RTF) ist folgendes zu beachten:

- Alle Zeichen außerhalb des ASCII Bereiches werden als Unicode-RTF abgelegt. Der Grund liegt u.a. darin, dass z.B. die griechischen Zeichen (z.B. Lambda) nur in dieser Weise sicher kodiert werden können.
- Graphiken können im RTF in unterschiedlicher Weise abgelegt werden. Als PNG / JPEG ByteStrom, oder als WMF8 ByteStrom. Die erste Variante wird nicht von allen RT Readern unterstützt. Die zweite ist die ältere und sollte von allen beherrscht werden. WMF8 sind um das mind. 15-fach größer, als die anderen Formate (GIF wird in RTF immer als WMF8 transportiert). Es ist also dringend angeraten, zu prüfen,

welches Format von Ihrem Reader unterstützt wird. In den Konvertierungsroutineen gtcGxml2RtfL und gtcGxml2RtfF kann mit dem Paramter bMakeWmf angegeben werden, ob WMF8 erzeugt werden sollen.

In diesem Kapitel werden die einzelnen Funktionen des Konverters erläutert. Code Beispiele folgen am Ende. Die Typen der nachfolgenden Funktionen sind in C++ Nomenklatur gehalten.

Fehlerobjekt

Fehler werden in den jeweiligen Puffern als XML Elemente übergeben. Details zum Fehlerformat finden Sie in der Shemadatei 'gtcerrors.xsd'.

int WINAPI gtcConvCreate(int iCode);

Diese Funktion erstellt eine Instanz des Konverters und liefert die ReferenzID als Rückgabewert. Ist iCode falsch, wird der Konverter im Demomodus betrieben.

int WINAPI gtcConvFree(int iRef);

Diese Funktion gibt eine zuvor mit gtcConvCreate erzeugte Instanze wieder frei. Diese Funktion muss nach Beendigung der Konvertierung unbedingt aufgerufen werden um das Objekt aus dem Speicher sauber zu entfernen.

int WINAPI gtcGetVersion();

Diese Funktion liefert die Versionsnummer der Konverterschnittstelle. Die Versionsnummer wird als Rückgabewert zurückgegeben und ist wie folgt aufgebaut:
Hunderterstelle und Größer = Major-Number,
Zehnerstelle = Minor-Number,
Einerstelle = Sub-Minor-Number
Beispiel: 120 entspricht der Version 1.2.0

int WINAPI gtcGetBuildVersion(int iRef, LPSTR *pConverterVersion);

Diese Funktion liefert die Versionsnummer der Konverterschnittstelle im Textformat. Die Versionsnummer wird über den Parameter pConverterVersion zurückgegeben und ist wie folgt aufgebaut:
{Major-Number}.{Minor-Number}.{Sub-Minor-Number}.{Build-Number}
Beispiel: '1.3.1.11'

Der Speicher für pConverterVersion wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcConvSetGaebDaXmlVersion(int iRef, LPCSTR pszGaebDaXmlVersion);

Diese Funktion setzt die Versionsnummer, welche bei der Erzeugung des GAEB-DAXML Textes durch den Konverter verwendet werden soll.

mögliche Werte für den Parameter pszGaebDaXmlVersion:
"3.0" : Bei der Konvertierung wird die Version 3.0 verwendet
"3.1" : Bei der Konvertierung wird die Version 3.1 verwendet

int WINAPI gtcConvSetRtfFontNameF0(int iRef, LPCSTR pszRtfFontNameF0);

Diese Funktion setzt den Fontname für die Standardschriftart *f0,* welche bei der Konvertierung in das RTF-Format verwendet werden soll.

int WINAPI gtcRtfAddL(int iRef, LPCSTR rtfL);

Diese Funktion fügt einen RTF Text als Langtext hinzu. Der Aufruf ersetzt jeden voher hinzugefügten Langtext.

int WINAPI gtcRtfAddK(int iRef, LPCSTR rtfK);

Diese Funktion fügt einen RTF Text als Kurztext hinzu. Der Aufruf ersetzt jeden voher hinzugefügten Kurztext.

int WINAPI gtcRtfAddA(int iRef, LPCSTR rtfL);

Diese Funktion fügt einen RTF Text als Langtext-AddText hinzu. Der Aufruf ersetzt jeden voher hinzugefügten Langtext-AddText.

int WINAPI gtcRtfAddAK(int iRef, LPCSTR rtfAK);

Diese Funktion fügt einen RTF Text als Kurztext-AddText hinzu. Der Aufruf ersetzt jeden voher hinzugefügten Kurztext-AddText.

int WINAPI gtcRtfAddF(int iRef, LPCSTR rtfF, LPCWSTR targetTag);

Diese Funktion fügt einen RTF Text als formatierten Text hinzu. Der Aufruf ersetzt jeden voher hinzugefügten formatierten Text. Der Parameter targetTag gibt den Namen des zu erzeugenden Textobjektes an, dieser muss einer der folgenden Werte sein:
COReas, Descrip.

int WINAPI gtcRtfAddML(int iRef, LPCSTR rtfML, LPCWSTR targetTag);

Diese Funktion fügt einen RTF Text als mehrzeiligen Text hinzu. Der Aufruf ersetzt jeden voher hinzugefügten mehrzeiligen Text. Der Parameter targetTag gibt den Namen des zu erzeugenden Textobjektes an, dieser muss einer der folgenden Werte sein:
ArtOutline, LblTx, BidComm.

int WINAPI gtcRtf2gxmlSetInner(int iRef, BOOL fGetInnerXml);

Gibt (optional) an, ob bei rtf 2 gxml Konvertierungen gxml als InnerXml geliefert werden soll. Der gesetzte Wert bleibt erhalten bis zum nächsten Aufruf bzw. zum Aufruf von gtcConvFree.

int WINAPI gtcRtf2gxml(int iRef, LPWSTR *pwBuf, LPWSTR *pwError, BOOL fDummy);

Diese Funktion führt die Wandlung von zuvor per gtcRtfAddL(/K/A/AK/F/ML) hinzugefügtem RTF nach GXML durch. Der GXML-Text wird in dem Parameter pwBuf abgelegt.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

Änderung: Ab der Version 1.2.0 wird der ursprüngliche vierte Parameter nicht mehr verwendet. Das zu erzeugende Zielformat wird nun durch die verwendeten gtcRtfAdd-Funktionen bestimmt. Der vierte Parameter bleibt allerdings aus Kompatibilitätsgründen als Dummy-Wert bestehen und wird bei der Konvertierung nicht berücksichtigt.

int WINAPI gtcRtf2gxmlL(int iRef, LPWSTR *pwBuf, BOOL *fHasTA, BOOL *fHasTB, LPWSTR *pwError);

Diese Funktion führt die Wandlung von zuvor per gtcRtfAddL/A hinzugefügtem RTF-Langtext in das entsprechende GXML-Element DetailTxt bzw. DetailAddText durch. Der GXML-Text wird in dem Parameter pwBuf abgelegt. Das Ergebnis kann unmittelbar an gtcUI.gxmlLtx bzw. gtcUI.gxmlAddTextLtx übergeben werden. Über den Parameter fHasTA wird zurückgegeben ob der Gxml-Text Textergänzungen des Ausschreibenden enthält. Über den Parameter fHasTB wird zurückgegeben ob der Gxml-Text Textergänzungen des Bieters enthält.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcRtf2gxmlK(int iRef, LPWSTR *pwBuf, BOOL *fHasTA, LPWSTR *pwError);

Diese Funktion führt die Wandlung von zuvor per gtcRtfAddK/AK hinzugefügtem RTF-Kurztext in das entsprechende GXML-Element OutlineText bzw. OutlineAddText durch. Der GXML-Text wird in dem Parameter pwBuf abgelegt. Das Ergebnis kann unmittelbar an gtcUI.gxmlKt bzw. gtcUI.gxmlAddTextKt übergeben werden. Über den Parameter fHasTA wird zurückgegeben ob der Gxml-Text Textergänzungen des Ausschreibenden enthält.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcGxml2RtfL(int iRef, LPCWSTR gxml, LPSTR *pBuf, BOOL bMakeWmf, LPWSTR *pwError);

Diese Funktion liefert in pBuf einen Langtext als RTF.

Der Wert für gxml kann unmittelbar aus gtcUI.gxmlLtx verwendet werden

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcGxml2RtfK(int iRef, LPCWSTR gxml, LPSTR *pBuf, LPWSTR *pwError);

Diese Funktion liefert in pBuf einen Kurztext als RTF.

der Wert für gxml kann unmittelbar aus gtcUI.gxmlKt verwendet werden

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcGxml2RtfF(int iRef, LPCWSTR gxml, LPSTR *pBuf, BOOL bMakeWmf, LPWSTR *pwError);

Diese Funktion liefert in pBuf einen formatierten Text als RTF.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

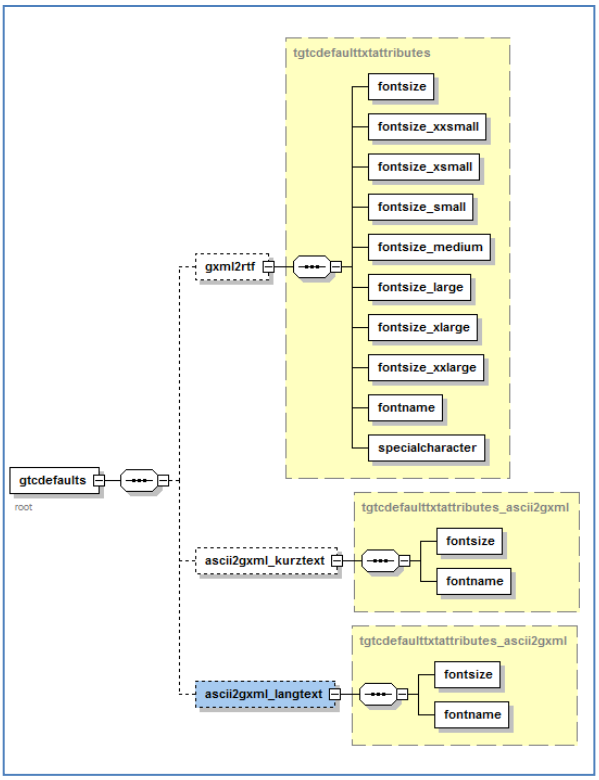
int WINAPI gtcGxml2RtfML(int iRef, LPCWSTR gxml, LPSTR *pBuf, LPWSTR *pwError);

Diese Funktion liefert in pBuf einen mehrzeiligen Text als RTF.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcGxml2RtfSetDefaults(int iRef, LPCWSTR sDefaults);

Setzt Defaultwerte gemäß gtcdefaults Schema. Die gesetzten Defaults bleiben erhalten bis zum nächsten Aufruf bzw. nächsten Free der Konverterreferenz.



gtcdefaults Schema

0-Werte für Zahlen (z.B. fontsize) und " (Leerstring) für Strings (z.B. fontname) deaktivieren die Defaultbehandlung für das jeweilige Attribut

Die Funktion ist hilfreich, um z.B. einen eine 10 Punkt Schrift als Standard im RTF zu erhalten.

int WINAPI gtcGetGxmlImgFromFile(int iRef, LPCSTR pszFilePath, LPCSTR pszFileTyp, LPCWSTR targetTag, LPWSTR *pwBuf);

Diese Funktion wandelt die Grafik-Datei mit dem absoluten Dateipfad pszFilePath in eine GAEB-DA-XML Element-Beschreibung um. Der Parameter pszFileTyp gibt den Typ der Quelldatei an, dieser muss einer der Werte 'jpeg', 'gif', 'png', 'tif', 'tiff', 'wmf', 'emf' bzw. 'bmp' sein. Der Parameter TargetTag enthält den Namen des zu erzeugenden Elementes, dieser kann **'Image'** oder **'image'** sein. Bei erfolgreicher Konvertieung wird das Resultat in dem Paramter pwBuf zurück gegeben.

Der Speicher für pwBuf wird vom Konverter bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAsciiClear(int iRef);

Diese Funktion löscht den Inhalt des ASCII Speichers.

int WINAPI gtcAsciiAddLineL(int iRef, LPCSTR ascii, char cKind, int iMarkLbl);

Diese Funktion fügt eine Zeile dem Langtext hinzu. ASCII darf kein $0A oder $0D enthalten

Bei Textergänzungen (ein oder mehrzeilig) muss cKind undiMarkLbl entsprechend gesetzt werden. Mehrzeilige Textergänzungen müssen die gleichen Werte für cKind und iMarkLbl in jeder Zeile aufweisen. die eigentliche Textergänzung (ComplBody) muss in einfachen Hochkommata eingeschlossen sein.

int WINAPI gtcAsciiAddLineK(int iRef, LPCSTR ascii, char cKind, int iMarkLbl);

Diese Funktion fügt eine Zeile dem Kurztext hinzu. ASCII darf kein $0A oder $0D enthalten.

Bei Textergänzungen (ein oder mehrzeilig) muss cKind undiMarkLbl entsprechend gesetzt werden. Mehrzeilige Textergänzungen müssen die gleichen Werte für cKind und iMarkLbl in jeder Zeile aufweisen. die eigentliche Textergänzung (ComplBody) muss in einfachen Hochkommata eingeschlossen sein.

int WINAPI gtcAsciiAddLineAK(int iRef, LPCSTR ascii);

Diese Funktion fügt eine Kurztextzeile dem zu erzeugenden AddText hinzu. Der ASCII-Text darf kein $0A oder $0D Zeichen enthalten.

int WINAPI gtcAsciiAddLineAL(int iRef, LPCSTR ascii);

Diese Funktion fügt eine Langtextzeile dem zu erzeugenden AddText hinzu. Der ASCII-Text darf kein $0A oder $0D Zeichen enthalten.

int WINAPI gtcAsciiAddLineF(int iRef, LPCSTR ascii, LPCWSTR targetTag);

Diese Funktion fügt eine Textzeile dem zu erzeugenden formatierten Text hinzu. Der Parameter targetTag gibt den Namen des zu erzeugenden Textobjektes an, dieser muss einer der folgenden Werte sein: **COReas, Descrip.**
Der ASCII-Text darf kein $0A oder $0D Zeichen enthalten.

int WINAPI gtcAsciiAddLineML(int iRef, LPCSTR ascii, LPCWSTR targetTag);

Diese Funktion fügt eine Textzeile dem zu erzeugenden mehrzeiligen Text hinzu. Der Parameter targetTag gibt den Namen des zu erzeugenden Textobjektes an, dieser muss einer der folgenden Werte sein: **ArtOutline, LblTx, BidComm.**
Der ASCII-Text darf kein $0A oder $0D Zeichen enthalten.

int WINAPI gtcAscii2gxml(int iRef, LPWSTR *pwBuf, LPWSTR *pwError);

Diese Funktion liefert in pwBuf den vollständigen gxml Text. Falls der Text in Kurz- und Langtext aufgeteilt ist so wird Kurz- und Langtext komplett formatiert gesetzt. Das Ergebnis kann unmittelbar an gtcUI.gxmlText übergeben werden.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAscii2gxmlK(int iRef, LPWSTR *pwBuf, BOOL *fHasTA, LPWSTR *pwError);

Diese Funktion liefert in pwBuf den Kurztext als gxml Text. Wurde der Kurztext mit der Methode gtcAsciiAddLineK gesetzt so wird der Text als OutlineText-Element geliefert, wurde der Kurztext mit der Methode gtcAsciiAddLineAK gesetzt so wird der Text als OutlineAddText-Element geliefert. Das Ergebnis kann unmittelbar an gtcUI.gxmlKt bzw. gtcUI.gxmlAddTextKt übergeben werden. Über den Parameter fHasTA wird zurückgegeben ob der Gxml-Text Textergänzungen des Ausschreibenden enthält.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAscii2gxmlL(int iRef, LPWSTR *pwBuf, BOOL *fHasTA, BOOL *fHasTB, LPWSTR *pwError);

Diese Funktion liefert in pwBuf den Langtext als gxml Text. Wurde der Langtext mit der Methode gtcAsciiAddLineL gesetzt so wird der Text als DetailTxt-Element geliefert, wurde der Kurztext mit der Methode gtcAsciiAddLineAL gesetzt so wird der Text als DetailAddText-Element geliefert. Das Ergebnis kann unmittelbar an gtcUI.gxmlLtx bzw. gtcUI.gxmlAddTextLtx übergeben werden. Über den Parameter fHasTA wird zurückgegeben ob der Gxml-Text Textergänzungen des Ausschreibenden enthält. Über den Parameter fHasTB wird zurückgegeben ob der Gxml-Text Textergänzungen des Bieters enthält.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcGxml2Ascii(int iRef, LPCWSTR gxml, LPWSTR* pwError, long iMaxChar);

Diese Funktion wandelt den übergebenen Text als Ascii Text. Abholen des Textes mit gtcAsciiGetLineL, gtcAsciiGetLineK, gtcAsciiGetLineAL, gtcAsciiGetLineAK, gtcAsciiGetLineF bzw. gtcAsciiGetLineML.

Siehe hierzu auch Beschreibung zu ASCII

int WINAPI gtcAsciiGetLineL(int iRef, LPSTR* pBuf, char* cKind, int* iMarkLbl);

Diese Funktion liefert in pBuf eine Zeile des Langtextes als ASCII. Jeder Aufruf dieser Funktion erhöht den Zeilenzähler. Im Result wird die Anzahl der in pBuf gelieferten Zeichen. Ist das letzte Zeichen erreicht, wird im Result -1 übergeben.

Der Parameter cKind ist als Zeiger auf einen einzelnen Character zu übergeben. Es wird nur das einzelne Zeichen für die Art der Textergänzung ohne abschließendes Nullzeichen zurück gegeben.

Hinweis: Am Zeilenende wird kein LF oder FF übergeben.

Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

Ist cKind ungleich 'T', handelt es sich um eine Zeile einer Textergänzung (ggf. mehrzeilig). Die eigentliche Textergänzung (ComplBody) sind von einfachen Hochkommata umschlossen. Möglicherweise aber eben erst in Folgezeilen.

int WINAPI gtcAsciiGetLineK(int iRef, LPSTR* pBuf, char* cKind, int* iMarkLbl);

Diese Funktion liefert in pBuf eine Zeile des Kurztextes als ASCII. Jeder Aufruf dieser Funktion erhöht den Zeilenzähler. Im Result wird die Anzahl der in pBuf gelieferten Zeichen. Ist das letzte Zeichen erreicht, wird im Result -1 übergeben.

Der Parameter cKind ist als Zeiger auf einen einzelnen Character zu übergeben. Es wird nur das einzelne Zeichen für die Art der Textergänzung ohne abschließendes Nullzeichen zurück gegeben.

Hinweis: Am Zeilenende wird kein LF oder FF übergeben.

Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

ist cKind ungleich 'T', handelt es sich um eine Zeile einer Textergänzung (ggf. mehrzeilig). Die eigentliche Textergänzung (ComplBody) sind von einfachen Hochkommata umschlossen. Möglicherweise aber eben erst in Folgezeilen.

int WINAPI gtcAsciiGetLineAL(int iRef, LPSTR* pBuf);

Diese Funktion liefert in pBuf eine Zeile des Langtextes(AddText) als ASCII. Jeder Aufruf dieser Funktion erhöht den Zeilenzähler. Im Result wird die Anzahl der in pBuf gelieferten Zeichen zurück gegeben. Ist das letzte Zeichen erreicht, wird im Result -1 übergeben.

Hinweis: Am Zeilenende wird kein LF oder FF übergeben.

Der Speicher für pBuf wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAsciiGetLineAK(int iRef, LPSTR* pBuf);

Diese Funktion liefert in pBuf eine Zeile des Kurztextes(AddText) als ASCII. Jeder Aufruf dieser Funktion erhöht den Zeilenzähler. Im Result wird die Anzahl der in pBuf gelieferten Zeichen zurück gegeben. Ist das letzte Zeichen erreicht, wird im Result -1 übergeben.

Hinweis: Am Zeilenende wird kein LF oder FF übergeben.

Der Speicher für pBuf wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAsciiGetLineF(int iRef, LPSTR* pBuf);

Diese Funktion liefert in pBuf eine Zeile des formatierten Textes als ASCII. Jeder Aufruf dieser Funktion erhöht den Zeilenzähler. Im Result wird die Anzahl der in pBuf gelieferten Zeichen zurück gegeben. Ist das letzte Zeichen erreicht, wird im Result -1 übergeben.

Hinweis: Am Zeilenende wird kein LF oder FF übergeben.

Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAsciiGetLineML(int iRef, LPSTR* pBuf);

Diese Funktion liefert in pBuf eine Zeile des mehrzeiligen Textes als ASCII. Jeder Aufruf dieser Funktion erhöht den Zeilenzähler. Im Result wird die Anzahl der in pBuf gelieferten Zeichen zurück gegeben. Ist das letzte Zeichen erreicht, wird im Result -1 übergeben.

Hinweis: Am Zeilenende wird kein LF oder FF übergeben.

Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAsciiGetStlbBauKey(int iRef, LPSTR* pBuf);

Diese Funktion liefert den STLB-Bau Schlüssel des aktuellen Textes, einschließlich Informationen zu Textergänzungen gem. dem hier erhältlichen Schema.

Der Speicher für pBuf wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAscii2RtfL(int iRef, LPSTR* pBuf, BOOL fMakeWmf, LPWSTR* pwError);

Diese Funktion wandelt ASCII Langtext nach RTF liefert diesen in pBuf. Der interne Ascii Puffer muss zuvor mit gtcAsciiClear geleert werden, der Ascii Text muss Zeilenweise mit gtcAsciiAddLineL hinzugefügt werden.

Das gleiche Ergebnis würde erzielt, wenn mit den entsprechenden Funktionen Ascii nach gxml gewandelt und dieses nach RTF gewandelt würde.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcAscii2RtfK(int iRef, LPSTR* pBuf, LPWSTR* pwError);

Diese Funktion andelt ASCII Kurztext nach RTF liefert diesen in pBuf. Der interne Ascii Puffer muss zuvor mit gtcAsciiClear geleert werden, der Ascii Text muss Zeilenweise mit gtcAsciiAddLineK hinzugefügt werden.

Das gleiche Ergebnis würde erzielt, wenn mit den entsprechenden Funktionen Ascii nach gxml gewandelt und dieses nach RTF gewandelt würde.

Der Speicher für pBuf und pwError wird von gtcConv bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcRtf2AsciiL(int iRef, LPWSTR* pwError, long iMaxChar);

Diese Funktion wandelt RTF Langtext in ASCII um. Der RTF Text muss zuvor mit gtcRtfAddL hinzugefügt sein. Der ASCII Text wird zeilenweise mit gtcAsciiGetLineL abgeholt.

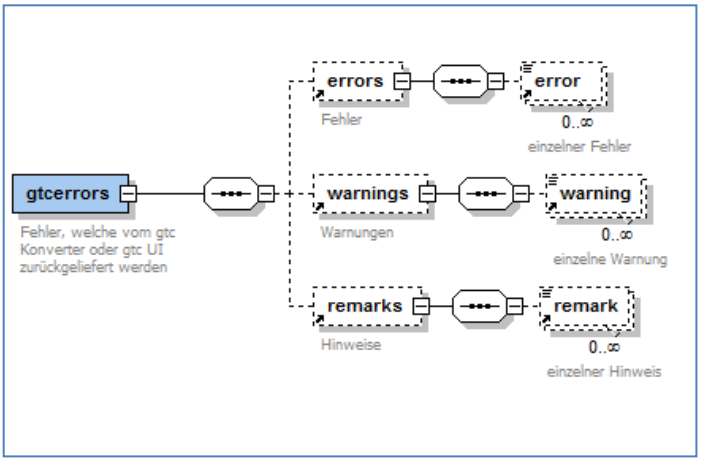
Der Speicher für pwError wird vom Konverter bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

int WINAPI gtcRtf2AsciiK(int iRef, LPWSTR* pwError, long iMaxChar);

Diese Funktion wandelt RTF Kurztext in ASCII um. Der RTF Text muss zuvor mit gtcRtfAddK hinzugefügt sein. Der ASCII Text wird zeilenweise mit g tcAsciiGetLineK abgeholt.

Der Speicher für pwError wird vom Konverter bereitgestellt und verwaltet. Notwendig hierzu ist unbedingt der abschließende Aufruf von gtcConvFree nach Beendigung der Konvertierung.

Schema Fehlermeldungen



2012.07.09 B.Rath