KTOPRUEF

Anwenderhandbuch

Inhaltsverzeichnis

Sachstand: 15. Januar 2006

Druckvorschlag:

Stellen Sie im Acrobat-Druckdialog das Feld „Seitenanpassung“ auf „An Papierformat anpassen“ und wählen Sie bei den „Eigenschaften“ des Druckers (je nach Druckertreiber) Duplexdruck sowie den sog. „Heft-Modus“. Sie erhalten dann ein seitenrichtiges DIN-A-5-Heft, das Sie mit Heftklammern zu einer gebundenen Broschüre zusammenheften können.

1. Einleitung

Vorbemerkung: *Nichts ist so alt wie die Zeitung von gestern* – dieses Sprichwort gilt auch und gerade in der Computerbranche, in der eine Software-Dokumentation zum Zeitpunkt ihres Drucks schon wieder veraltet ist. Die in diesem Handbuch enthaltenen Informationen entsprechen also dem Softwarestand zum Zeitpunkt der Drucklegung; für den aktuellen Stand sollten Sie jedoch unbedingt die Website http://www.ktopruef.de zu Rate ziehen, damit Sie über alle inzwischen evtl. eingetretenen Neuerungen informiert sind.

1.1 Tätigkeit von KTOPRUEF

Das KTOPRUEF-System prüft Bankverbindungen (Bankleitzahl, Kontonummer) sowie Adressen (Postleitzahl, Ort, ggf. Straße und Hausnummer) auf Plausibilität. Als Datenbasis werden die stets aktuellen Unterlagen der Deutschen Bundesbank bzw. der Deutschen Post AG herangezogen.

Plausibilität bedeutet in diesem Zusammenhang, daß geprüft wird,

- ob die angegebene Bankleitzahl existiert und ob die Kontonummer dazu paßt (Kontrolle der Prüfziffer gemäß dem Verfahren der jeweiligen Bank)
- ob die angegebene Postleitzahl existiert und zum angegeben Ort paßt
- ob in den 209 Orten mit unterschiedlichen Zustellpostleitzahlen die Straße (und bei unterteilter Straße auch die Hausnummer) zur angegebenen Postleitzahl-/Ort-Kombination paßt.

Es findet also *keine* Prüfung auf die reale Existenz der Bankverbindung oder der Adresse statt. Studien haben jedoch gezeigt, daß alleine durch den Wegfall von Flüchtigkeitsfehlern bei der Eingabe (die ja durch KTOPRUEF eliminiert werden) bis zu 90 Prozent der vormals fehlerhaften Daten vermieden werden können!

Zusätzlich zu den o.a. Gut-/Schlecht-Aussagen werden ggf. weitere Ergebnisse geliefert wie z.B. eine Nachfolge-Bankleitzahl bei weggefallener Bankleitzahl, oder der Ortsteil zur Ort-/Straße-Kombination und derlei mehr.

1.2 Nutzen von KTOPRUEF

In erster Linie eignet sich KTOPRUEF für eine Bankverbindungsprüfung vor einem beabsichtigten Lastschrifteinzug, da für Rücklastschriften (Wiederbelastung des Kontos des Einziehenden) von den beteiligten Banken ein Entgelt erhoben wird. Dieses Entgelt können Sie durch den Einsatz von KTOPRUEF in den meisten Fällen vermeiden. Aber auch für Rücküberweisungen (die bei nicht existentem Empfängerkonto momentan noch kostenlos erfolgen) ist KTOPRUEF geeignet. Denn in beiden Fällen (Lastschrift und Überweisung) ist zusätzlich zu den Bankentgelten noch die Arbeitszeit zu rechnen, die anfällt, um die korrekte Bankverbindung zu ermitteln (und ggf. der Aufwand für einen erneuten Lastschrifteinzug oder eine erneute Überweisung).

Gleiches gilt für die Adreßprüfung – auch wenn unzustellbare Briefe momentan noch kostenlos zurückgesandt werden, fällt immer noch der Nachforschungsaufwand bezüglich der korrekten Adresse an.

1.3 Anwendungsfälle von KTOPRUEF

KTOPRUEF eignet sich für alle Anwendungen, in denen Kundendaten verarbeitet und/oder gespeichert werden, insbesondere natürlich für Webshops, Callcenter, Auftragsannahmen usw. Bei der Datenerfassung durch den Kunden selbst (z.B. Eingabe der Bankverbindung in einem Web-Formular) passieren immer wieder kleine Flüchtigkeitsfehler, so daß der Kunde z.B. 7601085 statt richtig 76010085 als Bankleitzahl eingibt, oder 731495 statt richtig 734195 als Kontonummer, oder 100003791 als Kontonummer statt richtig 1000003791. Wird eine solche Lastschrift ungeprüft bei der Bank eingereicht, wird der gutgeschriebene Betrag rückbelastet, zusammen mit dem Rücklastschriftentgelt. (Das „berühmte“ BGH-Urteil, daß Rücklastschriften nichts kosten dürfen, betrifft ja nur den Kunden als Zahlungspflichtigen, nicht jedoch Sie als Zahlungsempfänger!)

KTOPRUEF sollte daher „direkt an der Quelle“ eingesetzt werden: In einem Webshop darf der Kunde die Bestellseite gar nicht erst absenden können, wenn die Bankverbindung falsch ist. In einem Callcenter sollte KTOPRUEF unmittelbar in die Bestellsoftware des Agenten integriert werden, damit falsche Bankverbindungen gar nicht erst eingegeben werden können. Die Bankverbindungsprüfung sollte also immer stattfinden, solange der Kunde noch „greifbar“ ist und seine falschen Angaben noch korrigieren kann.

KTOPRUEF gibt es als „Standalone“-Lösung für Windows, Linux und im Web (mit Eingabeformular und Ergebnisanzeige); üblicher und sinnvoller ist jedoch die Einbettung unmittelbar in die vorhandene Anwendersoftware. Dazu stehen zahlreiche Schnittstellen und Bibliotheken für Windows, Linux, ASP, IIS, .NET, PHP, Perl, WDDX oder WebServices bereit, so daß KTOPRUEF in nahezu jeder Umgebung problemlos eingesetzt werden kann.

1.4 Versionen von KTOPRUEF

Normalerweise wird die *Online-Version* von KTOPRUEF eingesetzt: Das Anwendungsprogramm sendet die zu prüfenden Daten über das Internet (verschlüsselt per SSL) an den Prüfserver und erhält unmittelbar darauf die Antwort. Da die Kommunikation gemäß dem weitverbreiteten HTTP-Protokoll abläuft (das auch für den Abruf von simplen WWW-Seiten eingesetzt wird), läßt sich KTOPRUEF auch in Firmennetzen mit Firewalls und/oder Proxy-Servern i.d.R. ohne jegliche Netzwerk-Änderungen einsetzen.

Sollte eine Verbindung ins Internet unter keinen Umständen möglich sein, gibt es auch noch eine *Offline-Version* von KTOPRUEF, so daß Bankverbindungen (nicht jedoch Adressen!) lokal auf dem Anwenderrechner geprüft werden können. Nachteil dieser Methode ist jedoch, daß eine lokale Installation der KTOPRUEF-Software erforderlich ist und daß mindestens alle drei Monate ein Update installiert werden muß, um die Bankleitzahlentabelle und die verschiedenen Prüfverfahren aktuell zu halten.

Daher ist die *Offline-Version* von KTOPRUEF nur dann anzuraten, wenn absolut keine Internet-Verbindung verfügbar ist. Außerdem ist die anwenderseitige Einbindung der *Online-Version* unter PHP oder .NET / SOAP / WebServices besonders einfach; hier genügen einige wenige zusätzliche Programmzeilen im vorhandenen System.

2. Online-Funktionen

Die Online-Funktionen sind funktionell in zwei große Bereiche gegliedert:
- Funktionen im Bankenbereich (Kontonummernprüfung, Bankensuche etc.)
- Funktionen im Adreßbereich (Prüfung von Postleitzahlen, Orten etc.)

Technisch gesehen kann auf den Server über zwei Wege zugegriffen werden:
- WDDX (siehe Abschnitt 2.3.1 auf Seite 14)
- WebServices (siehe Abschnitt 2.3.2 auf Seite 15)

Beide Zugriffsarten bestehen im Prinzip aus einem HTTP-Request wie bei einem Webseitenabruf („GET“ bei WDDX bzw. „POST“ bei WebServices), d.h. es sind keine besonderen Vorkehrungen in bezug auf Firewalls, Proxy-Server o.ä. nötig – wenn vom Client-Rechner aus ein „normaler“ Zugriff aufs Internet besteht (d.h. Webseiten im Browser abgerufen werden können), ist der Zugriff auf den Prüfserver i.d.R. ebenso gewährleistet.

Beide Zugangsverfahren können mit und ohne SSL („Secure-Socket-Layer“-Verschlüsselungsverfahren im Internet) benutzt werden.

Bei allen Online-Funktionen sind die Parameter `un` (UserName = Benutzername) und `pw` (PassWort = geheimes Kennwort) mit zu übermitteln, um Zugang zum Prüfserver zu erhalten. Ohne diese Angaben steht eine Demo-Version zur Verfügung, um die technischen Schnittstellen bezüglich der Zusammenarbeit mit der Anwendersoftware (ohne reale Daten) testen zu können (siehe Abschnitt 2.6 auf Seite 20).

In den folgenden Abschnitten werden zunächst die verschiedenen Funktionen mit ihren Ein- und Ausgabeparametern beschrieben; danach folgt die technische Realisierung mit WDDX oder als WebService.

2.1 Bankverbindungen („KTOPRUEF“)

Im Bankverbindungsbereich stehen drei Funktionen zur Verfügung: eine Bankverbindungsprüfung, eine Kreditkartenprüfung sowie eine Bankensuche.

2.1.1 Bankverbindungsprüfung

Die Bankverbindungsprüfung bildet das Herz des KTOPRUEF-Systems. Es wird geprüft, ob die angegebene Bankleitzahl real existiert und ob die Kontonummer (bei dieser Bankleitzahl) existieren kann.

Eingabeparameter:

`un`	Benutzername
`pw`	Kennwort
`blz`	Bankleitzahl (stets achtstellig, rein numerisch, also ohne Zwischenräume zu übermitteln)
`kto`	Kontonummer (maximal zehnstellig, rein numerisch, also ohne Zwischenräume, Bindestriche etc. zu übermitteln)

Ausgabeparameter:

`result` Numerisches Ergebnis zur Weiterverarbeitung wie folgt:
0 = Ok, Bankverbindung plausibel
-1 = Bankleitzahl existiert nicht
1 = Unbekannte Prüfmethode (tritt i.d.R. nicht auf)
2 = Prüfmethode nicht implementiert (-"- -"-)
3 = Diese Bank verwendet keine Prüfmethode
4 = Diese Kontonummer kann nicht geprüft werden
11 = Kontonummer ist länger als zehn Stellen
12 = Kontonummer enthält nicht nur Ziffern
13 = Kontonummer ist falsch (Prüfziffer stimmt nicht)
14 = Kontonummer ist 0 und damit unzulässig
15 = Kontoartenziffer ist unzulässig
16 = Kontonummer muß genau 8 Stellen lang sein
17 = Kontonummer muß genau 9 Stellen lang sein
18 = Kontonummer muß genau 7 Stellen lang sein
19 = Kontonummer muß 8 oder 10 Stellen lang sein
20 = Kontonummer muß genau 10 Stellen lang sein
21 = Kontonummer muß 7 oder 8 Stellen lang sein
Hinweise zu diesen Ergebnissen:
– Werte zwischen 0 und 10 müssen von Ihnen als „Ok“ interpretiert werden; lediglich Werte kleiner als 0 oder ab 11 aufwärts dürfen zu einer Ablehnung der Bankverbindung führen.
– Da eine Kontonummer eine simple *Zahl* ist, spielen führende Nullen keine Rolle; d.h. „000123“ ist dieselbe Kontonummer wie „123“ und damit *drei*stellig.

`resulttext` Ergebnis im (deutschen) Klartext; Sie sollten diesen Text (und nicht einfach nur „Bankverbindung falsch“ o.ä.) dem Anwender anzeigen, die weitere Bearbeitung in Ihrer Software jedoch nicht von diesem Text abhängig machen (da er sich u.U. auch einmal ändern kann).

`bankname` Name der Bank, falls die geprüfte Bankleitzahl gültig ist. I.d.R. inkl. Ort, in Groß- und Kleinschreibung und mit Umlauten. Maximal 27 Zeichen lang.
Leer, falls die geprüfte Bankleitzahl ungültig ist.

`newblz` Ggf. Hinweis auf veraltete oder neue Bankleitzahl, falls die geprüfte Bankleitzahl in Kürze ungültig wird oder vor kurzem ungültig wurde (sonst leer). Dieses Antwortfeld wird wie folgt befüllt:
Bei gültigen Bankleitzahlen (*result* >= 0):
– Feld ist leer: Normale gültige Bankleitzahl
– Feld ist *00000000*: Bankleitzahl wird demnächst ungültig; eine Nachfolge-Bankleitzahl ist nicht bekannt
– Feld ist *xxxxxxxx*: Bankleitzahl wird demnächst ungültig; die Nachfolge-Bankleitzahl ist *xxxxxxxx*

Bei ungültigen Bankleitzahlen (*result* = -1):
– Feld ist leer: Bankleitzahl ist völlig unbekannt
– Feld ist *00000000*: Bankleitzahl wurde vor kurzem ungültig; eine Nachfolge-Bankleitzahl ist nicht bekannt
– Feld ist *xxxxxxxx*: Bankleitzahl wurde vor kurzem ungültig; die Nachfolge-Bankleitzahl ist *xxxxxxxx*.

2.1.2 Kreditkartenprüfung

Vor der Autorisierung beim Kreditkartenunternehmen kann mit dieser Funktion geprüft werden, ob die Kartennummer überhaupt gültig sein kann (Prüfziffer wird errechnet und verglichen).

Eingabeparameter:

`un`	Benutzername
`pw`	Kennwort
`cc`	Kreditkartennummer (rein numerisch, also ohne Zwischenräume zu übermitteln)

Ausgabeparameter:

`result`	Numerisches Ergebnis zur Weiterverarbeitung wie folgt: 0 = Ok, Kreditkartennummer plausibel 1 = Unbekannter Kartentyp 12 = Kartennummer enthält nicht nur Ziffern 13 = Kartennummer ist falsch (Prüfziffer stimmt nicht)
`resulttext`	Ergebnis im (deutschen) Klartext; Sie sollten diesen Text (und nicht einfach nur „Kartennummer falsch“ o.ä.) dem Anwender anzeigen, die weitere Bearbeitung in Ihrer Software jedoch nicht von diesem Text abhängig machen (da er sich u.U. auch einmal ändern kann).

2.1.3 Bankensuche

Mit dieser Funktion können Banken gesucht werden, wenn ihr Name und/oder Ort (oder Teile davon) bekannt sind.

Eingabeparameter:

`un`	Benutzername
`pw`	Kennwort
`search`	Bankname und/oder Ort (oder Teile davon). Falls mehrere Suchbegriffe angegeben werden, müssen sie durch Zwischenräume getrennt werden. Der Server liefert dann nur Banken zurück, in denen *alle* Suchbegriffe vorkommen (d.h. *UND*-Verknüpfung).
`fragments`	Wird dieser Parameter weggelassen (oder mit 0 übergeben), werden die Suchbegriffe als ganze Wörter in-

terpretiert (d.h. „dresd“ als Suchbegriff findet weder die „Dresdner Bank“ noch die „Sparkasse Dresden“).
Mit dem Parameter *fragments=1* wird dagegen auch nach *Fragmenten* gesucht, d.h. in diesem Fall würden mit dem obigen Beispiel alle Banken zurückgeliefert, in deren Bezeichnung *irgendwo* das Fragment „dresd“ vorkommt.

Ausgabeparameter:

`banken` — Liste der gefundenen Banken (kann auch leer sein, falls die Suchkriterien auf keine einzige Bank zutreffen). Jede Bank ist im Format
`Bankleitzahl|Bankname`
gelistet (Bankleitzahl stets 8 Ziffern; Bankname i.d.R. inkl. Ort, in Groß- und Kleinschreibung und mit Umlauten. Maximal 27 Zeichen lang.

2.2 Adressen („ADDRESS“)

Die Adreßprüfung ist etwas anspruchsvoller als die Bankverbindungsprüfung, sowohl was die übergebenen Abfrageparameter als auch das zurückgelieferte Ergebnis anbelangt.

Die einfachste Prüfung ist das Zusammenpassen von Postleitzahl und Ort (dann brauchen auch nur diese Parameter übergeben werden).

Schon bei dieser einfachsten Prüfung werden, falls die Postleitzahl nicht zum Ort paßt, alle gegenwärtigen und ehemaligen Ortsnamen zurückgeliefert, zu denen diese Postleitzahl momentan paßt, damit dem Anwender eine Auswahlliste angeboten werden kann.

Eine Stufe höher kann dann auch noch die Straße mit übergeben werden; dann wird in den 209 Orten mit unterschiedlichen Zustell-Postleitzahlen auch die Straße geprüft.

Eine weitere Stufe höher kann – zusätzlich zur Straße – auch noch eine Hausnummer mit übergeben werden; dann kann in unterteilten Straßen (die sich über mehrere Ortsteile und/oder Zustell-Postleitzahlen erstrecken) auch noch die Hausnummer geprüft werden.

Die Prüfungen werden nicht nacheinander, sondern gleichzeitig durchgeführt, d.h. anhand der Art der übergebenen Parameter wird automatisch die passende Prüfung durchgeführt (wird eine Straße und/oder Hausnummer mit übergeben, wird sie geprüft; falls nicht, wird nur die Postleitzahl-/Ort-Prüfung durchgeführt).

Bei den Straßen- und Hausnummernprüfungen werden außerdem, falls sie ermittelt werden können, der Ortsteil und die korrekte Postleitzahl zurückgeliefert (falls die übergebene falsch war).

Sowohl bei Orts- als auch bei Straßennamen wird die postalisch korrekte Schreibweise zurückgeliefert, so daß vor dem Speichern eine „schlampige“ Eingabe des Anwenders korrigiert werden kann; so wird z.B. „nuern-berg“ in

die ordnungsgemäße Schreibweise „Nürnberg“ (mit Groß- und Kleinschreibung und Umlauten) konvertiert.

Parallel zur Straßenprüfung gibt es auch noch eine Postfachprüfung (also ob die Postfachnummer zur angegebenen Postfach-Postleitzahl paßt).

Und schließlich besteht die Möglichkeit, den Typ einer Postleitzahl herauszufinden (also Zustellung, Postfach, Großempfänger usw.).

Im folgenden werden die einzelnen Abfragemöglichkeiten näher beschrieben.

2.2.1 Prüfung von Postleitzahl und Ort

Die einfachste Variante ist die Prüfung, ob Postleitzahl und Ort zusammenpassen. Dies funktioniert für alle Orte (also auch jene, die nur eine einzige Zustell-Postleitzahl im gesamten Ort besitzen).

Die möglichen Prüfungsergebnisse bauen aufeinander auf; d.h. die Ergebnisse *dieser* Prüfung können auch in den *erweiterten* Prüfungen (mit Straße / Hausnummer etc.) auftreten.

Eingabeparameter:

`un`	Benutzername
`pw`	Kennwort
`func`	Immer die Konstante `check`
`plz`	Postleitzahl (stets fünfstellig, also ggf. auch mit führender Null!)
`ort`	Ortsname, der geprüft werden soll
`rewrite`	Wird dieser Parameter weggelassen (oder mit *0* übergeben), so führt die Prüfung nur dann zum Erfolg, wenn der Ortsname exakt in der ordnungsgemäßen postalischen Schreibweise übergeben wurde! In diesem Fall würde z.B. „Nuernberg“ oder „nürnberg“ abgelehnt, da die richtige Schreibweise „Nürnberg“ (also mit Groß- und Kleinschreibung und Umlaut) lautet. Es wird daher dringend empfohlen, *rewrite=1* zu übergeben, da in diesem Fall ein „schlampig“ geschriebener Ortsname dennoch als gültig erkannt und obendrein noch dessen postalisch korrekte Schreibweise im Ergebnisfeld *rewritten* (s.u.) zurückgeliefert wird!

Ausgabeparameter:

`result`	Numerisches Ergebnis zur Weiterverarbeitung wie folgt: 0 = Ok, Postleitzahl paßt zum Ort (ggf. Antwortfeld *rewritten* auswerten, s.u.) 1 = Unbekannte Funktion (*func* nicht *check*) 2 = Postleitzahl formal ungültig (also nicht fünf Ziffern) 3 = Postleitzahl formal gültig, aber nicht existent

4 = Postleitzahl gilt nur für andere Orte
(siehe Ergebnisfelder *orte* bzw. *alteorte*, s.u.)

`resulttext` Ergebnis im (deutschen) Klartext; Sie sollten diesen Text (und nicht einfach nur „Adresse falsch“ o.ä.) dem Anwender anzeigen, die weitere Bearbeitung in Ihrer Software jedoch nicht von diesem Text abhängig machen (da er sich u.U. auch einmal ändern kann).

`rewritten` Ortsname im postalisch korrekten Format (mit Groß- und Kleinschreibung und Umlauten), falls er ermittelt werden kann (ansonsten leer).

`orte` Wird nur bei *result = 4* zurückgeliefert!
Liste der momentan gültigen Orte zur angegebenen Postleitzahl. Diese Liste sollte dem Anwender präsentiert werden, damit er die korrekte Ortsbezeichnung auswählen kann.

`alteorte` Wird nur bei *result = 4* zurückgeliefert!
Liste der ehemals gültigen Orte zur angegebenen Postleitzahl, im Format:
`Gefunden|AlterOrt|AktuellerOrt`
wobei „Gefunden“ das Zeichen „+“ oder „-“ sein kann: „+“ wird in den Fällen zurückgeliefert, in denen eine Übereinstimmung des zu prüfenden Ortsnamen mit einem ehemaligen Ortsnamen festgestellt wird, ansonsten „-“.
Beispiel:
Für eine Abfrage mit *plz=90425* und *ort=Almoshof* erhalten Sie *result=4*; *orte* enthält nur „Nürnberg“, und *alteorte* enthält die Zeile `+|Almoshof|Nürnberg` (und, mit „-“, alle anderen ehemaligen Orte, für die die Postleitzahl 90425 aktuell gilt, z.B. `-|Boxdorf|Nürnberg`).

Diese Ergebnisfelder werden auch bei den darauf aufbauenden Prüfungen (mit Straße / Hausnummer etc.) benutzt!

2.2.2 Prüfung von Postleitzahl, Ort und Straße

Diese Prüfung baut auf die obige Prüfung von Postleitzahl und Ort auf (siehe Abschnitt 2.2.1 auf Seite 9), so daß hier nur die zusätzlichen Felder erläutert werden:

Eingabeparameter:

`str` Straßenname, der geprüft werden soll (kann mit „...str.“ oder auch mit „...straße“ übergeben werden, jedoch ohne Hausnummer! Für Hausnummernprüfungen siehe Abschnitt 2.2.3 auf Seite 11.)

Ausgabeparameter:

`result` Numerisches Ergebnis zur Weiterverarbeitung wie in Abschnitt 2.2.1 auf Seite 9), mit folgenden zusätzlich möglichen Ergebnissen:
5 = Straße und Postfach dürfen nicht gleichzeitig angegeben werden (falls Sie auch noch *pof=...* mitgeschickt haben)
8 = Postleitzahl und Ort passen zwar zusammen; die Postleitzahl ist jedoch keine Zustell-Postleitzahl (was sie für eine Straßenangabe sein müßte)
9 = Postleitzahl und Ort passen zusammen; über die Straße ist keine Aussage möglich, da die Postleitzahl im gesamten Ort gilt (muß von Ihrer Software als „Ok“ gewertet werden!)
10 = Postleitzahl und Ort passen zwar zusammen; die angegebene Straße gibt es in diesem Ort aber nicht
11 = Postleitzahl und Ort passen zwar zusammen; die angegebene Straße hat aber eine (oder mehrere) andere Postleitzahl(en) (siehe Feld *plzout*)

`str46` Straßenname im postalisch korrekten Format (mit Groß- und Kleinschreibung und Umlauten), falls er ermittelt werden kann (ansonsten leer). Maximale Länge 46 Zeichen. Immer mit „...str.“.

`str22` Straßenname im postalisch korrekten Format (mit Groß- und Kleinschreibung und Umlauten), falls er ermittelt werden kann (ansonsten leer). Maximale Länge 22 Zeichen. Immer mit „...str.“.

`plzout` Korrekte Postleitzahl für die angegebene Straße, falls sie ermittelt werden kann und die übergebene Postleitzahl falsch war (ansonsten leer).

`ortsteil` Ortsteil im postalisch korrekten Format (mit Groß- und Kleinschreibung und Umlauten), falls er ermittelt werden kann (ansonsten leer).

Beachten Sie bitte, daß bei dieser Prüfung die Ergebnisse *result=0* und *result=9* als „Ok“ gewertet werden müssen!

2.2.3 Prüfung von Postleitzahl, Ort, Straße und Hausnummer

Diese Prüfung baut auf die obige Prüfung von Postleitzahl, Ort und Straße auf (siehe Abschnitt 2.2.2 auf Seite 10), so daß hier nur die zusätzlichen Felder erläutert werden:

Eingabeparameter:

`hnr` Hausnummer, die geprüft werden soll (bitte lassen Sie Zusätze wie „a“, „bis ...“ oder „1/2“ jeweils weg!). Die Angabe der Hausnummer ist zwingend mit der Anga-

be einer Straße verbunden (siehe Parameter *str* in Abschnitt 2.2.2 auf Seite 10).

Ausgabeparameter:

`result` Numerisches Ergebnis zur Weiterverarbeitung wie in Abschnitt 2.2.2 auf Seite 10), mit folgenden zusätzlich möglichen Ergebnissen:
12 = Postleitzahl, Ort und Straße passen zusammen; über die Hausnummer ist keine Aussage möglich, da die Postleitzahl in der gesamten Straße gilt (muß von Ihrer Software als „Ok“ gewertet werden!)
13 = Postleitzahl, Ort und Straße passen zwar zusammen; die angegebene Hausnummer gibt es in dieser Straße jedoch nicht
14 = Postleitzahl, Ort und Straße passen zwar zusammen; für die angegebene Hausnummer gilt jedoch eine andere Postleitzahl (siehe Feld *plzout*)

Beachten Sie bitte, daß bei dieser Prüfung die Ergebnisse *result=0*, *result=9* und *result=12* als „Ok“ gewertet werden müssen!

2.2.4 Prüfung von Postleitzahl, Ort und Postfach

Diese Prüfung baut auf die Prüfung von Postleitzahl und Ort auf (siehe Abschnitt 2.2.1 auf Seite 9), so daß hier nur die zusätzlichen Felder erläutert werden:

Eingabeparameter:

`pof` Postfachnummer, die geprüft werden soll (bitte als reine Zahl ohne Zwischenräume oder sonstige Sonderzeichen übergeben, also z.B. nicht „12 34-56“, sondern „123456“!)

Ausgabeparameter:

`result` Numerisches Ergebnis zur Weiterverarbeitung wie in Abschnitt 2.2.1 auf Seite 9), mit folgenden zusätzlich möglichen Ergebnissen:
5 = Straße und Postfach dürfen nicht gleichzeitig angegeben werden (falls Sie auch noch *str=...* mitgeschickt haben)
6 = Postleitzahl und Ort passen zwar zusammen; die Postleitzahl ist jedoch keine Postfach-Postleitzahl (was sie für eine Postfachangabe sein müßte)
7 = Postleitzahl und Ort passen zwar zusammen; die angegebene Postleitzahl gilt jedoch nicht für diese Postfachnummer

2.2.5 Herausfinden des Typs einer Postleitzahl

Mit dieser Funktion können Sie herausfinden, von welchem Typ eine Postleitzahl ist (Zustellung, Postfach, Großempfänger...).

Eingabeparameter:

`un`	Benutzername
`pw`	Kennwort
`func`	Immer die Konstante `type`
`plz`	Postleitzahl (stets fünfstellig, also ggf. auch mit führender Null!)

Ausgabeparameter:

`result`	Numerisches Ergebnis zur Weiterverarbeitung wie folgt: 1 = Unbekannte Funktion (*func* nicht *type*) 2 = Postleitzahl formal ungültig (also nicht fünf Ziffern) 3 = Postleitzahl formal gültig, aber nicht existent 101 = Postleitzahl gilt für Postfächer 102 = Postleitzahl gilt für Schalterausgabe 103 = Postleitzahl gilt für Gruppen-Großempfänger 104 = Postleitzahl gilt für Einzel-Großempfänger 105 = Postleitzahl ist eine Aktions-Postleitzahl 106 = Postleitzahl gilt für Zustellung 107 = Postleitzahl gilt für Zustellung *und* Postfächer
`resulttext`	Ergebnis im (deutschen) Klartext; Sie sollten diesen Text (und nicht einfach nur „Postleitzahl falsch“ o.ä.) dem Anwender anzeigen, die weitere Bearbeitung in Ihrer Software jedoch nicht von diesem Text abhängig machen (da er sich u.U. auch einmal ändern kann).

2.3 Technische Zugriffsmöglichkeiten auf den Prüfserver

Momentan gibt es zwei verschiedene Protokolle, um per Internet auf den Prüfserver zuzugreifen: WDDX und WebServices („SOAP“ – „Simple Object Access Protocol“).

WDDX ist ein relativ simples Protokoll, das einfach zu programmieren ist und für das es vorgefertige Bibliotheken in vielen Programmiersprachen gibt (z.B. Perl oder PHP).

WebServices/SOAP ist etwas anspruchsvoller, was die Formatierung der Anfrage-/Antwort-Datenpakete anbelangt, wird aber ebenfalls von einer Anzahl Entwicklungssysteme unterstützt (z.B. *Microsoft .NET*). Allerdings sind dabei erfahrungsgemäß oft einige Protokolldetails von Hersteller zu Hersteller unterschiedlich implementiert, so daß es gelegentlich zu Inkompatibilitäten zwischen WebService-Clients und dem Prüfserver kommt. Daher sollte vor einem produktiven Einsatz von KTOPRUEF als WebService die technische Zusammenarbeit zwischen Client und Server anhand des Demo-Zugangs (siehe Abschnitt 2.6 auf Seite 20) geprüft werden. Die KTOPRUEF-WebServices wurden bisher mit den WSDL-Importern von Borland und Microsoft.NET erfolgreich getestet. Versuche mit Java werden zur Zeit durchgeführt.

Beide Protokolle verwenden HTTP („HyperText Transfer Protocol“) als Basis, das auch für den Abruf von „normalen“ Webseiten genutzt wird. Eine spezielle Netzwerk- oder Firewall-Konfiguration ist daher i.d.R. nicht nötig, wenn vom Client-Rechner aus auch bisher schon auf Internet-Websites zugegriffen werden kann. Auch der Einsatz von HTTP-Proxy-Servern steht der Verwendung von KTOPRUEF nicht entgegen.

Der Zugriff kann mit („https://...“) oder ohne („http://...“) SSL („Secure Socket Layer“ = Verschlüsselung der Daten über das Internet) erfolgen.

2.3.1 Zugriff über WDDX

WDDX („Web Distributed Data Exchange“, siehe auch http://www.openwddx.org) ist ein sehr einfaches Protokoll, das für die Eingabeparameter das gleiche Format verwendet wie ein Formular auf einer Webseite (mit *method="get"*), d.h. die Eingabeparameter werden einfach an die Web-Adresse wie folgt angehängt:

```
GET http(s)://wddx.hanft.de/funktion?par1=value1&par2=value2
```

wobei `funktion` für `ktopruef` oder `address` steht und die Eingabeparameter mit Parameter-/Wert-Paaren `par=value` übergeben und durch `&` getrennt werden.

Die Ausgabeparameter werden in einer einfachen XML-Seite zurückgeliefert, die z.B. für die Anfrage

```
http://wddx.hanft.de/ktopruef?un=x&pw=x&blz=76010085&kto=1856
```

wie folgt aussehen kann:

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!DOCTYPE wddxPacket SYSTEM 'wddx_0100.dtd'>
<wddxPacket version='1.0'>
 <header/>
 <data>
  <struct>
   <var name='result'>
    <string>0</string>
   </var>
   <var name='resulttext'>
    <string>Ok</string>
   </var>
   <var name='bankname'>
    <string>Postbank Nürnberg</string>
   </var>
   <var name='newblz'>
    <string></string>
   </var>
  </struct>
 </data>
</wddxPacket>
```

Wie man sieht, werden die Ausgabeparameter des Servers zwischen <*var*> und `</var>` geklammert, wobei innerhalb dieser Variable auch noch der Typ (z.B. `<string>...</string>`) angegeben wird. Man kann also die Server-Ergebnisse ziemlich einfach selbst auswerten, oder man bedient sich der

einschlägigen Funktionen seines Entwicklungssystems (z.B. „wddx_deserialize“ von CURL in PHP).

Ein spezieller Ergebnistyp ist eine Liste bzw. „Array“ (der nur in den Ausgabeparametern „Banken“ der Bankensuche, Seite 8, sowie „Orte“ und „Alte-Orte“ der Adreßprüfung, Seite 10, vorkommen kann) und wie folgt aussieht:

```
[...]
<var name='alteorte'>
  <array length='3'>
    <string>Demodorf</string>
    <string>Demohausen</string>
    <string>Demoheim</string>
  </array>
</var>
[...]
```

2.3.2 Zugriff über WebServices

Das genaue Format der WebServices-Datenpakete soll hier gar nicht näher erläutert werden, da in den Entwicklungssystemen, die WebServices verwenden, die nötigen Funktionen bereits enthalten sind.

Auf der Website http://webservices.hanft.de finden Sie eine Übersicht der angebotenen WebServices in verschiedenen Varianten. Dort befinden sich auch Links zu den WSDL-Dokumenten, die die einzelnen WebServices beschreiben und die von Ihnen importiert werden müssen, damit Ihr Client-Rechner mit den nötigen Informationen versorgt wird.

Hinweis: Für die Adreßprüfung unter Microsoft.NET scheint nur die Variante *IAddress3* korrekt zu funktionieren.

Ein konkretes Beispiel finden Sie im Abschnitt 2.5.2 („Einbindung in Anwendungen“) auf Seite 17.

2.4 Handeingabe

Sie können den Prüfserver „von Hand“ (Sie tippen die zu prüfenden Daten ein und erhalten das Prüfergebnis angezeigt) auf zwei Arten nutzen:

- Für die Bankverbindungsprüfung steht Ihnen auf der Webseite http://www.ktopruef.de/bank.htm ein Web-Eingabeformular zur Verfügung.

- Unter Windows können Sie die (kostenlose) Software http://www.ktopruef.de/HanftWddx.exe verwenden, mit der Sie per gewohnter Windows-Benutzeroberfläche alle Funktionen des Prüfservers ausnutzen können.

2.5 Einbindung in Anwendungen

Der größte Vorteil des KTOPRUEF-Systems ist die einfache Einbindung in eigene Anwendungen. Im folgenden werden einige Beispiele aufgeführt, wie KTOPRUEF zusammen mit gängigen Programmiersprachen, Entwicklungssystemen und Anwendungen eingesetzt werden kann.

2.5.1 WDDX mit PHP

Im folgenden wird ein PHP-Skript aufgeführt, das eine Adreßprüfung durchführt. Die zu prüfenden Werte sind in diesem Beispiel als Konstante angegeben; in der Realität würden sie z.B. aus den Eingabefeldern eines Webseitenformulars stammen.

```
<?
$un = "demo";
$pw = "demo";
$plz = "99999";
$ort = "Demostadt";
?>

<?php

/* Hier wird der String für den GET per URL aufgebaut */
$address_get =
"func=check&rewrite=1&un=$un&pw=$pw&plz=$plz&ort=$ort";
$url = "http://wddx.hanft.de/address?$address_get";

echo "<br>Prüfe Adresse mit URL: $url<br> " ;

/* Zur Verbindung mit dem Server wird CURL verwendet */
/* mit curl_init wird zunächst die URL festgelegt */
$ch = curl_init ($url);

/* Hier werden noch einige Parameter für CURL gesetzt */
curl_setopt ($ch, CURLOPT_HEADER, 0); /* Header nicht
in die Ausgabe */
curl_setopt ($ch, CURLOPT_NOBODY, 0); /* Ausgabe nicht
in die HTML-Seite */
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1); /*
Umleitung der Ausgabe in eine Variable ermöglichen */

/* Aufruf von CURL und Ausgabe mit WDDX deserialisieren
*/
$result = curl_exec($ch);
if (curl_errno($ch))
{
  echo "<br>cURL-Fehler: " . curl_error($ch);
}
else
{
  $des_out = wddx_deserialize($result);
  curl_close ($ch);

  /* Die deserialisierte Ausgabe in ein Array schreiben
*/
  while (list($key, $val) = each($des_out))
  {
    $ergebnis[$key] = $val;
  }

  /* und das Ergebnis ausgeben */
  echo "<br>Ergebnis: [" . $ergebnis['result'] . "]";
  echo "<br>Ergebnistext: [" . $ergebnis['resulttext']
```

```
. "]";
}
?>
```

2.5.2 WebServices mit Microsoft.NET

Hier finden Sie ein Beispiel für die KTOPRUEF-Implementierung mit Microsoft.NET:

- Importieren Sie die WSDLs mit dem .NET-Programm *wsdl.exe* wie folgt:

```
wsdl /language:vb http://webservices.hanft.de/wsdl/IKtoPruef
wsdl /language:vb http://webservices.hanft.de/wsdl/IAddress3
```

Sie erhalten dadurch die Dateien *IKtoPruefService.vb* und *IAddress3Service.vb*.

- Schreiben Sie ein VB.NET-Programm *test.vb* ungefähr wie folgt (Sie finden dieses Programm auch unter http://www.ktopruef.de/webserv.htm – damit Sie es nicht abtippen müssen; Text-Zeilenumbrüche in diesem Listing müssen eliminiert werden):

```
Imports System
Class Test
   Public Shared Sub Main()
      Dim Result as Integer
      Dim ResultText as String
      Dim Rewritten as String
      Dim BankName as String
      Dim NewBlz as String
      Dim Banken() as String
      Dim Orte() as String
      Dim AlteOrte() as String
      Dim I as Integer

      Dim myKtoPruef as New IKtoPruefService()

      ' Bankverbindungsprüfung

      Console.WriteLine("Bankverbindungsprüfung:")
      Result = myKtoPruef.TestBlzKto("demo", "demo", _
        "89999999", "1234", _
        ResultText, BankName, NewBlz)
      Console.WriteLine("  Ergebnis: " & _
        Result.ToString())
      Console.WriteLine("  ResultText: " & ResultText)
      Console.WriteLine("  BankName: " & BankName)
      Console.WriteLine("  NewBlz: " & NewBlz)

      ' Kreditkartenprüfung

      Console.WriteLine("Kreditkartenprüfung:")
      Result = myKtoPruef.TestCc("demo", "demo", _
        "4556000400074992", ResultText)
      Console.WriteLine("  Ergebnis: " & _
        Result.ToString())
      Console.WriteLine("  ResultText: " & ResultText
```

```
        ' Bankensuche

        Console.WriteLine("Bankensuche:")
        Banken = myKtoPruef.Search("demo", "demo", _
          "Demobank Demostadt", 0)
        Select Case Banken.Length
          Case 0: Console.WriteLine("  Ergebnis: Keine
Banken gefunden.")
          Case 1: Console.WriteLine("  Ergebnis: Eine
einzige Bank gefunden:")
          Case Else
            Console.WriteLine("  Ergebnis: " & _
Banken.Length.ToString() & " Banken gefunden:")
        End Select
        For I=0 to Banken.Length-1
          Console.WriteLine("     " & Banken(I))
        Next

        ' Adreßprüfung

        Dim myAddress as New IAddress3Service()
        Console.WriteLine("Adreßprüfung:")
        Result = myAddress.CheckPlzOrt("demo", "demo", _
          "99999", "Demodorf", 1, _
          ResultText, Rewritten, Orte, AlteOrte)
        Console.WriteLine("  Ergebnis: " & _
          Result.ToString())
        Console.WriteLine("  ResultText: " & ResultText)
        For I=0 to Orte.Length-1
          Console.WriteLine("  Orte: " & Orte(I))
        Next
        For I=0 to AlteOrte.Length-1
          Console.WriteLine("  AlteOrte: " & AlteOrte(I))
        Next

    End Sub
End Class
```

- Übersetzen Sie obiges Programm mit (als einzeiligem Befehl):

```
vbc /t:exe /r:System.dll, System.Web.dll, System.XML.dll,
   System.Web.Services.dll test.vb IKtoPruefService.vb
                  IAddress3Service.vb
```

(oder wie immer Sie in Ihrem Entwicklungssystem ein VB.NET-Programm übersetzen).

- Sie erhalten dann beim Programmlauf folgendes Ergebnis:

```
Bankverbindungsprüfung:
  Ergebnis: 0
  ResultText: Ok
  BankName: Demo-Bank
  NewBlz:
Kreditkartenprüfung:
  Ergebnis: 0
```

```
   ResultText: Ok
Bankensuche:
   Ergebnis: Eine einzige Bank gefunden:
     89999999|Demobank Demostadt
Adreßprüfung:
   Ergebnis: 4
   ResultText: Postleitzahl 99999 gilt nur für andere
Orte
   Orte: Demostadt
   AlteOrte: +|Demodorf|Demostadt
   AlteOrte: -|Demohausen|Demostadt
   AlteOrte: -|Demoheim|Demostadt
```

2.5.3 MS-Office / ASP / Visual Basic / VBSkript

Für den Einsatz unter Windows mit Microsoft-Produkten empfiehlt es sich, das sog. *COM-Objekt* „HanftWddx“ zu installieren. Sie können es herunterladen unter http://www.ktopruef.de/HanftWddx.exe und installieren es (mit Administrator-Rechten!), indem Sie es einfach einmal aufrufen. Sie sehen dabei auf den ersten Blick nur die in Abschnitt 2.4 beschriebene „gewöhnliche“ Windows-Benutzeroberfläche; aber quasi „im Hintergrund“ hat sich durch den Programmstart ein sog. *COM-Server* installiert, der von allen Windows-Programmen aufgerufen werden kann.

Grundsätzlich funktioniert die Einbindung immer so ähnlich wie folgt:

```
Dim KtoPruef as Variant
Set KtoPruef = CreateObject("HanftWddx.KtoPruef")
KtoPruef.SetProxy("mein.proxy.de", 3128) ' (evtl.)
KtoPruef.SSLMode = 1 ' nur falls gewünscht
KtoPruef.TestBlzKto("Benutzername","Passwort",_
   "Bankleitzahl","Kontonummer")
ErgebnisAlsZahl = KtoPruef.Result
ErgebnisAlsText = KtoPruef.Resulttext
ErgebnisBankName = KtoPruef.BankName
```

(Das *COM-Objekt* für die Adreßprüfung heißt „HanftWddx.Address“); die Ein- und Ausgabeparameter finden Sie in Abschnitt 2.2 ab Seite 8.)

Es empfiehlt sich, die *HanftWddx*-COM-Objekte in die jeweiligen Microsoft-Anwendungen einzubinden, da dann alle Schnittstellen und Parameter nahtlos in die jeweilige Anwendung integriert sind. Dies funktioniert z.B. bei MS-Excel wie folgt:

- Wählen Sie *Extras* - *Makro* - *Visual Basic-Editor*.
- In diesem VB-Editor wählen Sie *Extras* - *Verweise*.
- Kreuzen Sie in dem dort erscheinenden Fenster *Hanft WDDX-Bibliothek für KtoPruef und Address* an und bestätigen Sie mit *Ok*.
- Erzeugen Sie ein neues Modul mit z.B. folgendem Inhalt:

```
Dim KtoPruef As New HanftWddx.KtoPruef
Public Function TestBlzKto(Blz, Kto As String) As
Integer
   Call KtoPruef.TestBlzKto("un", "pw", Blz, Kto)
   TestBlzKto = KtoPruef.Result
End Function
```

- Wechseln Sie vom VBA-Editor wieder in die Office-Applikation zurück. Sie können dort jetzt die hier definierte Funktion *TestBlzKto* als „normale“ Excel-Funktionen in der Tabelle verwenden (als Feldfunktion *=TestBlz-Kto(A2;B2)*.)

2.5.4 Delphi

Für die Einbindung in Delphi gibt es, wenn das COM-Objekt „HanftWddx“ installiert ist (siehe Abschnitt 2.5.3 auf Seite 19), zwei Möglichkeiten: „Late Binding“ und „Early Binding“. „Late Binding“ ist zwar schneller programmiert und compiliert, allerdings machen sich Fehler bei den Funktionsnamen (z.B. „TextBlzKto“ statt „TestBlzKto“) erst zur Laufzeit bemerkbar. Mit „Early Binding“ finden Sie solche Fehler bereits beim Compilieren. Im folgenden werden beide Möglichkeiten angegeben:

- Late Binding:

```
uses
   ComObj;
var
   myKtoPruef: OleVariant;
   myResult: Integer;
begin
   myKtoPruef:=CreateOleObject('HanftWddx.KtoPruef');
   myKtoPruef.TestBlzKto('un', 'pw', '76010085', '1856')
   myResult:=myKtoPruef.Result
end;
```

- Early Binding:

Importieren Sie zunächst die Typbibliothek von *HanftWddx*, und zwar mit „*Projekt - Typbibliothek importieren*“. Sie finden *HanftWddx* unter „*Hanft WDDX Bibliothek für KtoPruef und Address*“. Dann sieht der Code wie folgt aus:

```
uses
   HanftWddx_TLB;
var
   myKtoPruef: IKtoPruef;
   myResult: Integer;
begin
   myKtoPruef:=CoKtoPruef.Create;
   myKtoPruef.TestBlzKto('un', 'pw', '76010085', '1856')
   myResult:=myKtoPruef.Result
end;
```

2.6 Demo-Möglichkeiten

Sie können – ohne Anmeldung und (abgesehen von Ihrer IP-Adresse) vollkommen anonym – eine Online-Verbindung zu meinem Prüfserver herstellen und als Benutzername und Kennwort jeweils *demo* verwenden. Damit können Sie die unten beschriebenen Demo-Abfragen durchführen, um zu testen, ob Ihre Software mit dem Prüfserver zusammenarbeiten kann.

2.6.1 Bankverbindungsprüfung

Geben Sie bei Benutzung des Demo-Zugangs stets die (real nicht existierende) Bankleitzahl *89999999* an. Gerade Konto- oder Kreditkartennummern (z.B. *1234*) liefern dann das Ergebnis *0* („Ok“); ungerade Konto- oder Kreditkartennummern (z.B. *1235*) das Ergebnis *13* („falsche Prüfziffer“).

Falls Sie eine andere Bankleitzahl als *8999999* angeben, erhalten Sie das Ergebnis *-6* („Bank existiert in Demo-Version nicht“).

Eine leere Kontonummer (oder *0*) liefert das Ergebnis *14* („Kontonummer 0 gibt es nicht“).

Bei der Bankensuche erhalten Sie – unabhängig von den Suchbegriffen – stets die *Demobank Demostadt* mit der (real nicht existierenden) Bankleitzahl *89999999*).

Wenn Sie diese Ergebnisse erhalten, haben Sie die Online-Schnittstelle funktionsfähig installiert!

2.6.2 Adreßprüfung

Verwenden Sie bitte immer die (real nicht existierende) Postleitzahl *99999*. Bei allen anderen Postleitzahlen erhalten Sie stets das Ergebnis *3* („Postleitzahl existiert nicht“).

Der „korrekte“ Ort für die Postleitzahl *99999* lautet *Demostadt*. Mit dieser Kombination erhalten Sie das Ergebnis *0* („Ok“).

Mit allen anderen Ortsnamen als *Demostadt* erhalten Sie das Ergebnis *4* („Postleitzahl gilt nur für andere Orte“), zusammen mit der Liste *orte*, in der nur *Demostadt* steht, und der Liste *alteorte*, in der die „alten Orte“ *Demodorf*, *Demohausen* und *Demoheim* stehen.

Wenn Sie einen dieser drei „alten Orte“ angeben, wird das Feld *Gefunden* in der *alteorte*-Liste bei diesem Ort auf + gesetzt (siehe Seite 10).

Bei der Abfrage des Typs einer Postleitzahl können Sie wiederum nur *99999* angeben; Sie erhalten dann das Ergebnis *106* („Zustellung“). Bei allen anderen Postleitzahlen wird wieder *3* („Postleitzahl existiert nicht“) zurückgeliefert.

Wenn Sie diese Ergebnisse erhalten, haben Sie die Online-Schnittstelle funktionsfähig installiert!

3. Offline-Funktionen

Dieses Kapitel wird später nachgeliefert.

Dieses Kapitel wird später nachgeliefert.

4. Argumente für KTOPRUEF

Warum sollten Sie KTOPRUEF (und kein anderes, ähnliches Produkt) einsetzen? Hier finden Sie einige Argumente, die für den Einsatz on KTOPRUEF sprechen:

- *Die langjährige Erfahrung.* Falls Sie sich schon gefragt haben, wo der ja eigentlich ziemlich unaussprechliche Name KTOPRUEF herkommt: Die allererste Version der Software (die damals noch Teil eines größeren Abrechnungssystems war) wurde bereits in DOS-Zeiten entwickelt, als Dateinamen noch gemäß dem „8.3“-Schema aufgebaut waren und daher keine Software einen längeren Namen als acht Buchstaben haben konnnte. Als eigenständig vermarktetes Produkt gibt es KTOPRUEF bereits seit 1998; seitdem wurde es ständig verbessert und weiterentwickelt.

- *Die Update-Sicherheit.* Können Ihnen andere Anbieter garantieren, daß ihre Daten immer aktuell sind? Oft finden Sie auf deren Websites noch nicht einmal Aussagen darüber, wie aktuell die Daten momentan gerade sind. Wenn Sie mit veralteten Daten arbeiten, kann u.U. Ihr Webshop-Kunde nicht bei Ihnen bestellen, weil seine neue Bankleitzahl noch nicht als gültig akzeptiert wird! Durch den Wartungsvertrag erhalten Sie bei der Offline-Version die Sicherheit, jederzeit mit den aktuellen Bankdaten zu arbeiten; die Online-Version ist gar ohne Ihr Zutun stets aktuell.

- *Die Plattformunabhängigkeit.* Die Offline-Version von KTOPRUEF gibt es für Windows und für Linux, und Ihre Softwarelizenz gilt für jede der beiden Plattformen, so daß Sie bei einem Wechsel die Software nicht neu kaufen müssen – Sie müssen mir dazu noch nicht einmal Bescheid geben, sondern Sie laden sich einfach die Datendatei fr die gewünschte Plattform herunter! Natürlich kann auch die Online-Version von beiden Plattformen aus (auch gemischt) genutzt werden.

- *Die Integration in eigene Software.* Bei vielen anderen Anbietern müssen Sie eine vorgegebene Software installieren, in diese Software z.B. Ihre Kundendatei im CSV-Format einlesen und erhalten dann die Ergebnisse in einem Ausgabefenster, das Sie vielleicht wieder in ihre eigene Software exportieren knnen. Das ist sehr umständlich! KTOPRUEF dagegen kann zu einem integralen Bestandteil Ihrer eigenen Software werden – so können Sie z.B. die Bankverbindungsprüfung in MS-Excel als ganz normale Spaltenformel "=TestBlzKto(A2;B2)" angeben, und das sowohl mit der Offline- als auch mit der Online-Version!

- *Die Vielseitigkeit.* Wo sonst finden Sie ein Produkt, das es offline und online gibt, für Windows und für Linux, als COM-Server, als DLL-Bibliothek, als WDDX-Schnittstelle, als WebService, mit und ohne SSL? Sollten sich Ihre Anforderungen eines Tages ändern, paßt sich KTOPRUEF Ihren neuen Erfordernissen einfach an – so haben Sie Investitionsschutz!

- *Die Nutzung von Standards.* Bei der Programmierung von KTOPRUEF wurde bewußt darauf geachtet, auf vorhandene Standards aufzusetzen und Speziallösungen zu vermeiden: Bei der Offline-Version können Sie

unter Windows entweder das COM-Objekt installieren (Komponentenmodell von Microsoft, die bevorzugte Variante) oder eine DLL-Bibliothek zu Ihrer Software dazubinden; unter Linux sind die KTOPRUEF-Bibliotheken einfach aus Ihrem *gcc*-Programm heraus aufzurufen (*.h-Includes* werden mitgeliefert). Bei der Online-Version können Sie den WDDX-Standard (Web Distributed Data Exchange) verwenden, oder Sie nutzen KTOPRUEF als WebService (was insbesondere aus *.NET*-Anwendungen heraus extrem einfach ist).

- *Günstige Preise.* Bei anderen Anbietern müssen Sie bei der Bankverbindungsprüfung oft teures Geld für zweifelhafte Bonitätsprüfungen oder fragwürdige Score-Werte mitbezahlen, obwohl Sie solche Auskünfte vielleicht gar nicht wollen oder brauchen. Oder es besteht eine Mindestabnahmepflicht von einer Million Bankverbindungsprüfungen pro Jahr. Vergessen Sie das alles – durch die Reduktion von KTOPRUEF auf das Wesentliche zahlen Sie keinen Cent mehr als nötig!

- *Vorteilhafte Vertragsbedingungen.* Bei KTOPRUEF gibt es nicht viel „Kleingedrucktes“ und insbesondere keine lange Vertragsbindung: Einen Jahres-Wartungsvertrag für die Offline-Version können Sie zum Ablauf eines jeden Jahres kündigen; den Zugang zur Online-Version sogar zum Ablauf eines jeden Kalendervierteljahres!

- *Einfache Abwicklung.* Bei KTOPRUEF gibt es kein umständliches Hin und Her mit „Fragen Sie uns nach den Preisen“ oder „Schicken Sie uns ein Angebot“. Alle Unterlagen befinden sich auf dem KTOPRUEF-Webserver, inklusive Web- bzw. PDF-Formularen für Bestellungen (Offline-Version) bzw. Anmeldungen (Online-Version).
  Die Bestellung der Offline-Version erfolgt einfach durch das Ausfüllen und Absenden eines Web-Bestellformulars. Wenige Sekunden nach Ihrer Bestellung erhalten Sie bereits die Rechnung im PDF-Format. Sobald Ihre Überweisung eingegangen ist, erhalten Sie per E-Mail die Zugangsdaten für den KTOPRUEF-Update-Server, von dem Sie die vollständige Datendatei herunterladen können.
  Falls Sie die Online-Version bevorzugen, können Sie das Anmeldeformular unmittelbar am Bildschirm ausfüllen und mir danach per Fax oder Post zusenden. Sobald Ihre Anmeldung bei mir eingegangen ist, erhalten Sie per E-Mail die Zugangsdaten für den Prüfserver.
  Ich garantiere den Zugang zur Online-Version 24 Stunden nach dem Erhalt der Anmeldung; in der Realität geht das aber meist viel schneller. Wenn Sie mir die Anmeldung faxen, haben Sie in der Regel bereits nach zehn Minuten Ihre Zugangskennung und können sofort loslegen!

- *Gratis-Support per E-Mail.* Wenn Sie Fragen oder Probleme haben, schreiben Sie einfach eine E-Mail an *support@ktopruef.de*. Bei KTOPRUEF gibt es keine teure telefonische 0190/0900-Support-„Hotline“!

Stichwortverzeichnis