



Textbausteine und Makros

Andreas Viebke
Ernst-Thälmann-Straße 86 A
14532 Kleinmachnow
www.typer.de

Alle Rechte vorbehalten

Release Notes

3.6.6.2

1. Die neue Funktion `collect` zeigt einen Dialog an, mit dem die Elemente einer Liste mit Hilfe von Eingabefeldern bearbeitet werden können.
2. Die neue **Listenmethode** `list` liefert das Element mit dem als Argument übergebenen Index als Liste. Dies ist insbesondere bei Arrays nützlich, wenn eine Spalte nicht über ihren Namen, sondern ihren Index adressiert wird.
`CONTACTS.list(3)` ergibt dasselbe wie `CONTACTS[3]^`.
3. Die Funktion `regexp` beherrscht nun vollständige reguläre Ausdrücke, die ferner auch auf Listen angewendet werden können.
4. Alle regulären Ausdrücke in ShortCut werden nun als vollständige reguläre Ausdrücke nach Pearl-Standard interpretiert, auch diejenigen zur Definition von Fenstertiteln oder Programmen, in denen Kürzel erlaubt oder verboten werden. Bestehende reguläre Ausdrücke nach dem Muster `''+string+''` werden jedoch erkannt und müssen nicht angepasst werden, wenn `string` nur alphanumerische Zeichen enthält. Ein Ausdruck wie `*notepad*` müsste demnach nicht angepasst werden, `*@*.de` dagegen schon.
5. Die neue Funktion `synclist` liefert aus einer Liste diejenigen Elemente, die in den ganzzahligen Anteilen der Elemente einer anderen Liste vermerkt sind.
6. Mit der neuen **Listenmethode** `item` kann man auf ein Listen- oder Array-Element über einen berechneten Namen zugreifen.
`CONTACTS.item('last'+name')[1]` ergibt dasselbe wie `CONTACTS.lastname[1]`.
7. Der Zeichenbereichsoperator (siehe **Operatoren**) wird nun wieder richtig erkannt.

3.6.6.1

1. Die Funktion `explode()` liefert bei der Umwandlung einer Zeichenkette in eine Liste nun auch dann die richtige Anzahl Elemente, wenn das letzte Element in der Zeichenkette leer ist, die Zeichenkette also mit dem Separator endet.
2. Die **Listenmethoden** `draw` und `drop` liefern nun die Anzahl der nach der Operation verbliebenen Elemente.
3. Mit den neuen Funktionen `Utf8ToAnsi()` und `AnsiToUtf8()` können UTF-8-kodierte Zeichenketten in ANSI bzw. umgekehrt umgewandelt werden.
4. Die neue **Konstante** `UTF8BOM` enthält die Zeichenkette, mit denen Dateien im UTF-8-Format häufig beginnen.
5. Die neue Funktion `unquote()` entfernt von einer Zeichenkette oder den Elementen einer Liste die Anführungszeichen am Anfang und am Ende.
6. Mit der neuen Funktion `quoted()` lässt sich prüfen, ob eine Zeichenkette von Anführungszeichen umgeben ist.

3.6.6.0

1. Makros, deren Inhalt Richt Text ist, können nun auch per Hotkey aufgerufen werden.
2. Die Position des ShortCut-Hauptfensters wird nun korrigiert, wenn es sich außerhalb des sichtbaren Bereichs befindet..
3. Das neue **Systemmakro** `$focus` wird aufgerufen, kurz bevor ein Anwendungsfenster den Eingabefokus erhält.
4. Die neue Funktion `fileage` ermittelt das Erstellungsdatum einer Datei.
5. In Ausdrücken, mit denen zu der Länge einer Zeichenkette eine Zahl addiert wird, muss die Längenberechnung mit den Präfixen `+$` nicht mehr geklammert werden. Beispiel: Anstelle von `P:=A+(+$STR)`; kann nun `P:=A++$STR`; geschrieben werden.
6. Die neue Anweisung `until` funktioniert wie `while`, kehrt jedoch die Testbedingung um. `while !A {...}` kann also als `until A {...}` formuliert werden.
7. Die Anweisung `arrays` ist entfallen. Verwenden Sie bitte stattdessen das Synonym `lists`.
8. Wenn bei der Anweisung `list` nur ein Wert angegeben ist, so wird aus diesem nicht mehr durch Auswertung die Liste der zuzuweisenden Werte berechnet. Der Wert wird stattdessen direkt dem ersten Element der Liste zugewiesen. Um einen Wert auszuwerten und das Ergebnis als einzelne Werte der Liste zuzuweisen, kann der Auswertungsoperator verwendet werden. Beispiel:

```
L1[:=("Viele", "Köche", "verderben", "den", "Brei");
list L2 [quote(string(L1, ",", ""))];
lprint textbox(L2);
```
9. Mit den neuen **Konstanten** `SCREENWIDTH` und `SCREENHEIGHT` kann die Bildschirmauflösung ermittelt werden.
10. Bei den eingebauten (**konstanten**) Listen `CLIP`, `COMPUTERLIST`, `TIMERLIST`, und `USERLIST` stehen nun die Methoden `clear`, `delete`, `draw`, `drop`, `first`, `last` und `reverse` zur Verfügung. Nur bei `CLIP` wirken sich die Methoden `clear`, `delete`, `draw` und `drop` auch auf die eingebaute Liste aus. `CLIP.reverse` liefert eine umgekehrte Kopie von `CLIP`, kehrt den Zwischenablagenstapel jedoch nicht mehr um.
11. Bei Listen- und Zeichenketten-Variablen stehen nun die **Methoden** `delete` und `insert` zur Verfügung. `delete` erfordert mindestens ein Argument (den Index des zu löschenden Elements) und optional als zweites Argument die Anzahl der zu löschenden Elemente. `insert` erfordert mindestens zwei Argumente, den Index des Zielelements sowie mindestens ein an dieser Stelle einzufügendes Element. Die Elemente werden in umgekehrter Reihenfolge an derselben Stelle eingefügt, so dass beispielsweise `S:="abc"; lprint S.insert(2, "12", "34");` die Zeichenkette `a1234bc` ergibt. Dies gilt analog auch für Listen.
12. Listenelemente können nun mit einer negativen Indexangabe adressiert werden. In diesem Fall wird der tatsächliche Index vom Ende der Liste her bestimmt. Dies ist vorteilhaft, wenn Listen rückwärts durchlaufen werden. Beispiel:

```
list L 'alle', 'meine', 'entchen';
repeat (I,L) lprint L[-I];
```
13. Mit der neuen Anweisung `items`, lassen sich Bezeichner definieren, mit denen man auf die Elemente von Listen zugreifen kann, ohne eine Indexzahl zu verwenden. Diese Bezeichner werden wie Methoden verwendet. Beispiel:

```
list L 'eins', 'zwei', 'drei', 'vier';
items (L) one, two, three, four;
lprint L.three;
```
14. Die Anweisung `array`, die bisher synonym zu `list` war, hat eine eigene Aufgabe erhalten. `array` dient nun dazu, eine zweidimensionale Liste (eine Liste von Listen, also eine Tabelle) zu definieren und optional mit Werten zu belegen. Die Anweisung kombiniert die beiden Anweisungen `lists` und `items`.
15. Mit der neuen **Methode** `items` kann man die Bezeichner ermitteln, die bei einer Variablen verfügbar sind. Beispiel:

```
array A (one,two);
textbox(A.items);
```
16. Die neue **Methode** `where` liefert bei einem `array` die Liste derjenigen Datensätze (Zeilen), bei denen die als Argument übergebene Bedingung zutrifft. Beispiel: `CONTACTS.where(age<45 && age>25)`

17. Bei den Anweisungen `list`, `lists`, `array` und `items` kann der Auswertungsoperator verwendet werden, um den/die Namen von Variablen zu bestimmen. Beispiel:
- ```
lists [N] 'a', 'b', 'c', 'd';
```
- Dies wertet `N` aus, um den/die Namen der zu verwendenden Listenvariablen zu ermitteln. Außer bei `list` können mehrere Namensquellen angegeben werden, die auch Listen sein dürfen. Beispiel:
- ```
M[]:=('R','S'); N[]:=('T','U');
lists ([M],[N]) 'a', 'b', 'c', 'd';
```
- Dies gilt analog auch für die Werte, die den Listen zugewiesen werden. Der erste Auswertungsoperator eines Wertes expandiert diesen, um die einzelnen zuzuweisenden Werte zu ermitteln. Im folgenden Beispiel werden der Liste `A` drei Werte zugewiesen:
- ```
W:="'Wert1','Wert2'";
list A [W], "Wert3";
```
18. Den Anweisungen `list`, `lists` und `array` kann nun eine der Präfix-Anweisungen `set`, `local` und `global` vorangehen, um den Geltungsbereich der zu definierenden Variablen einzuschränken. Wird keine dieser Präfix-Anweisungen angegeben, wird wie bisher `set` angenommen.
19. Der neue Präfix-Operator `@` liefert bei Variablenreferenzen den Namen der referenzierten Variablen. Beispiel:
- ```
FOO:="abc"; &BAR:=FOO;
lprint BAR; // -> abc
lprint @BAR; // -> FOO
```
20. In ShortCuts Konfigurationsdatei `Shortcut.ini`, die sich im Programmverzeichnis befindet, gibt es die Möglichkeit, unter `[General]` den Parameter `UsageFile` einzutragen. Mit dem Wert dieses Parameters kann der Pfad der Datei festgelegt werden, in der die Statistik für Makro- und Applikationsaufrufe gespeichert wird. In diesem Pfad können nun Umgebungsvariablen wie `%username%` verwendet werden. Dadurch kann dieser Wert für alle Anwender in einem Netzwerk, deren Homeverzeichnis im selben Verzeichnis liegt, identisch sein. Dies vereinfacht die Erstellung von Statistiken vor allem in **Multi-User-Umgebungen**.
21. Vergleiche mit Zeichenketten als Operanden erzeugen nun das richtige boolesche Ergebnis.
22. Beim Downgrade zu einer Version vor 3.6.6.0 muss ShortCut erneut freigeschaltet werden.

3.6.5.3

- Die Funktion `chrscan()` überprüft nicht mehr die Zeichen in einem Bereich, sondern genau die angegebenen Zeichen.
 - Die Funktionen `strtohex()` und `hextostr()` sind nun dokumentiert.
 - Es gibt zwei neue Operatoren, `?:` und `??`. Der Operator `?:` liefert den Wert der auf seiner linken Seite angegebenen Variablen, sofern diese definiert ist. Andernfalls liefert er den Wert des auf der rechten Seite angegebenen Ausdrucks. Der Operator `??` funktioniert wie der `?`-Operator, jedoch muss auf seiner linken Seite ein Variablenname stehen, kein Ausdruck. Existiert diese Variable, wird der rechts neben `??` stehende Ausdruck ausgewertet. Existiert die Variable dagegen nicht und folgen noch ein `:` sowie ein weiterer Ausdruck, so wird dieser letzte Ausdruck ausgewertet und geliefert. Die beiden neuen Operatoren machen in den meisten Fällen die Verwendung der Funktion `defined` überflüssig.
 - Die neue Funktion `remoteexecmacro()` führt auf einem Computer ein dortiges Makro aus, sofern das Passwort für die Ausführung von Makros dafür bekannt ist.
 - Mit dem neuen Subbefehl `30` der Funktion `statistics()` kann eine Statistikdatei eingelesen und einem Computer oder Benutzer zugeordnet werden.
 - Das mitgelieferte Statistikmakro `$statistics` wurde stark erweitert. So ist es nun möglich, die Statistik von einem anderen Computer im Netzwerk berechnen zu lassen. Ferner kann die Statistik jetzt auch auf Basis der Benutzer anstatt der Computer erstellt werden. Hierfür müssen einige **Voraussetzungen** erfüllt sein.
 - Bei Textblöcken (`\[+t var]...\[-t]`) kann die zu belegende Variable nun indiziert werden. Beispiel:
- ```
\[+t var[1]]...\[-t]
```
- Kommentare in Makros können sich nun auch innerhalb mehrzeiliger Anweisungen befinden. Beispiel:
- ```
global MINIMUM:=5 // Kleinster Wert
MAXIMUM:=50; // Größter Wert
```

3.6.5.0

1. Kategorien können jetzt über das Optionen-Menü als vertraulich gekennzeichnet werden. Eine vertrauliche Kategorie und die darin enthaltenen Makros werden nicht übertragen, wenn ein anderer Computer die Makros von dem betreffenden ShortCut anfordert (mit **Anfordern von** über das Kontextmenü oder die Funktion `requestitems`). Beachten Sie, dass der Vertraulichkeitsstatus von Kategorien verloren geht, wenn die Makros von früheren ShortCut-Versionen gespeichert werden.
2. ShortCuts eingebauter Texteditor kann nun Schlüsselwörter in Makros (beispielsweise Funktionen, Anweisungen) farblich hervorheben. Es können insgesamt 10 Farben für die Bestandteile von Makros vergeben werden, eine davon für eigene Schlüsselwörter. Der entsprechende Dialog ist über das Dropdown-Menü im unteren Bereich des Editors zu erreichen. Die Farben und Schlüsselwörter werden von der `Synchronisation` erfasst.
3. Bei der Aktualisierung von `Verzeichniskategorien` wird die Makroliste nur noch dann neu aufgebaut, wenn tatsächlich solche Kategorien vorhanden sind.
4. Der Dialog, der von der Funktion `input()` geöffnet wird, wird nun dynamisch an die Textmenge des Mitteilungstextes angepasst. Ferner kann das Eingabefeld auch mehrzeilig sein. Analog zu `select()` können nun Buttons zum Dialog hinzugefügt werden.
5. Der `message()`-Dialog kann nun auch mit zusätzlichen Buttons versehen werden.
6. Bei allen Dialogen, die mit zusätzlichen Buttons versehen werden können, werden Buttons ohne Text nicht mehr angezeigt. Diese Buttons werden dennoch wie angezeigte Buttons behandelt und können daher zur Erzeugung von Abstand vor, zwischen und nach Buttons verwendet werden. Ferner macht eine negative Breite den Button zum voreingestellten Button des Dialogs.
7. Die Anweisung `case` akzeptiert nun auch eine Liste, in der die zu vergleichenden Werte angegeben werden können. Dies ist insbesondere in der Anweisung `switch` nützlich.
8. `switch` erzeugt nun keinen eigenen Variablenkontext mehr, wodurch die in dieser Anweisung definierten Variablen nach dem Verlassen der Anweisung weiter existieren.
9. Die neue Funktion `cutstr()` "subtrahiert" eine Zeichenkette von einer anderen. Sie liefert den Teil der ersten übergebenen Zeichenkette, der übrig bleibt, nachdem die mit der zweiten Zeichenkette vom Anfang her übereinstimmenden Zeichen entfernt wurden.
10. Der neue `Variablenreferenzoperator` `&` ermöglicht es, einer Funktion eine einfache Variable und nicht nur deren Wert zu übergeben. Er ergänzt den bestehenden Listenreferenzoperator (ebenfalls `&`).
11. Der Präfix `$` vor einem Ausdruck bewirkt nun zusätzlich zur Konvertierung des Ergebnisses in eine Zeichenkette, dass der Ganzzahlenanteil des Ergebnisses auf die Länge der Zeichenkette gesetzt wird.
12. Der `Operator` `==`, der bisher gleichbedeutend mit `=` war, fasst seine Operanden jetzt immer als Zahlen auf. Zusammen mit dem `$`-Präfix lässt sich daher mit `==` einfacher prüfen, ob eine Zeichenkette eine bestimmte Länge hat (`$A==5`) oder zwei Zeichenketten gleich lang sind (`$A==$B`). Die Negation dazu (`! ==`) ist ebenfalls verfügbar. Für den Längenvergleich von Zeichenketten mit den Operatoren `<`, `>`, `<=` und `>=` muss nach wie vor `length(A)` oder `+$A` verwendet werden, da ansonsten die Zeichenketten selbst verglichen werden. Beispiel: `A:="66"; if (+$A<=2) beep; if (length(A)<=2) beep;`
13. Ist ein Operand des binären Minus-Operators eine Zeichenkette, wird deren Länge verwendet. Nach `A:="Apfelbaum"` liefert `$A-4` also 5. Dies gilt analog auch für die Operatoren `*`, `/` und `%`. Der binäre Plus-Operator fasst nach wie vor beide Operanden als Zeichenketten auf, wenn mindestens einer von ihnen eine Zeichenkette ist. `"12"+34` liefert also wie bisher `"1234"`, da `34` in `"34"` umgewandelt wird (weil der erste Operand eine Zeichenkette ist).
14. `Konstanten` kann nun kein Wert mehr zugewiesen werden. Ferner liefert die neue Konstante `CONSTANTNAMES` die Namen aller Konstanten als Zeichenkette (separiert durch jeweils ein Leerzeichen), und `TRUE` und `FALSE` sind nun Konstanten, die als 1 bzw. 0 definiert sind.
15. Vor dem Schlüsselwort `else` kann das Semikolon nun entfallen, sofern der Kontext eindeutig ist. Intern wird das Semikolon jedoch in den Makrotext eingefügt und erscheint daher in Fehlermeldungen, die den Kontext ausgeben. Es sind nun Schreibweisen wie die folgende möglich:
`if (Z=1) A:=5 else A:=6;`
16. Bei der `Zusatzoption` `\ [+t] . . \ [-t]` unterscheidet ShortCut nun zwischen Groß- und Kleinschreibung (`T` vs. `t`), damit ein Textbereich einen anderen Textbereich enthalten kann. Ein Textbereich kann nur mit demselben Optionsbuchstaben geschlossen werden. Bei klein geschriebenen Optionsbuchstaben wird der Textbereich farblich wie eine Zeichenkettenkonstante dargestellt, andernfalls wie normaler Makrocode.

17. Mit dem neuen **Konfigurationsparameter** `TextBuffers` kann man einstellen, wieviele ältere Versionen der Textpuffer-Datei aufbewahrt werden sollen. Voreingestellt werden 10 Versionen aufbewahrt. Mit der Funktion `textbuffer()` kann man auf diese Versionen zugreifen.
18. ShortCut versucht jetzt, diejenigen Dialoge, die per Makro geöffnet wurden, in den Vordergrund zu holen.
19. Bei einem Versionswechsel gehen nun keine **Konfigurationsparameter** mehr verloren, die manuell in die Konfigurationsdatei `ShortCut.ini` eingetragen wurden.
20. In Makros können konstante Zeichenketten nicht mehr durch einen Backslash am Ende der Zeile auf der nächsten Zeile fortgesetzt werden.
21. Die neue Funktion `httpgetasync()` lädt ein Dokument im Hintergrund herunter. Das heruntergeladene Dokument wird einem Makro übergeben.

3.6.2.2

1. Eingegebene Kürzel werden nun besser erkannt, während kurze Makros (beispielsweise Timer-Makros) im Hintergrund laufen.
2. Die Beschreibung der Funktion `dateval()` enthält Beispiele zur Berechnung des Wochentags und der Kalenderwoche zu einem beliebigen Datum ab dem 1.1.1900.
3. Die Funktion `float()` liefert nun eine Zeichenkette, wenn ihr erstes Argument eine Fließkommazahl ist.
4. Der **Operator** `$` wandelt jetzt auch eine Fließkommazahl in eine Zeichenkette (zwei Nachkommastellen) um.
5. Die Funktion `select()` zeigt nun einen Dialog mit Auswahlkästchen an und erlaubt eine Vorauswahl der Elemente.
6. Mit der Funktion `settle()` können auch Elemente einer Liste nach Auswahl mit `select()` gelöscht werden. Ferner liefert die Funktion nun -1, wenn die Liste nach der Ausführung leer ist.
7. Die neue **Konfigurationsoption** `PopupTimeout` erlaubt es, eine Zeit vorzugeben, nach der der mit `popup()` geöffnete Auswahldialog automatisch geschlossen wird (Voreinstellung: deaktiviert). `PopupTimeout` ist auch als Parameter der Anweisung `option` verfügbar und wird nach jedem Makro auf den ursprünglichen Wert zurückgesetzt.
8. Die Funktion `message()` kann nun auch eine Tray-Icon-Nachricht anzeigen.
9. Die Listenvariable für zusätzliche Funktionsargumente, `FARGS`, ist nun auch verfügbar, wenn einer Funktion keine Argumente übergeben werden. In diesem Fall ist die Liste leer. Ferner wird `FARGS` nun wie eine lokale Variable behandelt.
10. Die Funktion `date()` liefert jetzt immer eine Zeichenkette.
11. Das neue **Systemmakro** `$idle` wird immer dann ausgeführt, wenn die Systemlast gering ist. Das Makro sollte so kurz wie möglich laufen.
12. Mit der neuen **Konfigurationsoption** `IdleInterval` lässt sich das Mindestintervall zwischen zwei aufeinander folgenden Aufrufen von `$idle` festlegen (Voreinstellung: 120s). `IdleInterval` ist auch als Parameter der Anweisung `option` verfügbar. Die Einstellung wirkt global, d.h. sie wird nach Ablauf eines Makros nicht zurückgesetzt.
13. Mit dem neuen **Zeichenbereichs-Operator** `..` können Zeichenketten erzeugt werden, die die Zeichen des angegebenen Bereichs in auf- oder absteigender Reihenfolge enthalten. Beispiel: `'A'..'D'+ 'd'..'a'` ergibt `ABCDdcba`
14. Aufgrund der optimierten Erfassung von Variablen- und Funktionsnamen wird Makro-Code jetzt schneller ausgeführt.
15. Kleinere Verbesserungen bei der Lokalisierung.
16. Das Systemmakro `$default` wird nicht mehr unterstützt und kann gelöscht werden.

3.6.2.0

1. Die neuen Funktion `crc16()` und `crc32()` berechnen die 16-Bit- bzw. 32-Bit-Prüfsumme einer Zeichenkette (wie beispielsweise einer Datei).
2. Die Funktion `filelink()` hat ein weiteres Argument erhalten, mit dem die Quelle des Icons der anzulegenden Verknüpfung festgelegt werden kann.
3. Die Statistikdaten werden jetzt im benutzerspezifischen ShortCut-Verzeichnis gespeichert, wenn das Programmverzeichnis schreibgeschützt ist.
4. Wenn einem Makro ein Hotkey zugewiesen ist, so ist der Hotkey jetzt während der Makroausführung deaktiviert. Das Makro kann daher den Hotkey senden, ohne dass das Makro erneut ausgeführt wird.
5. Wenn ein Makro, dem ein Hotkey zugewiesen ist, nicht ausgeführt werden kann, dann sendet ShortCut den Hotkey jetzt ans System. Dies bedeutet, dass der Hotkey eines Makros, das in einer bestimmten Anwendung blockiert wurde, von der aktiven Anwendung erkannt werden kann.
6. Wenn ShortCut über das Icon im Tray-Menü deaktiviert wird, werden alle Makro-Hotkeys freigegeben. Wird ShortCut wieder aktiviert, sind auch die Makro-Hotkeys wieder aktiv.
7. Mit der neuen **Konfigurationsoption** `MaxHotkeyLag` kann man festlegen, wie lange ShortCut höchstens wartet, bis ein Makro-Hotkey losgelassen wurde. Dies betrifft nur Makros, die ausführbar sind, also keine Makros, die per Fenster- oder Anwendungsausschluss blockiert wurden.
8. Bei der Makro-**Zusatzoption** `\[+t varname]...\[-t]`, mit der sich längere Texte in Makros ablegen lassen, kann nun vor dem Variablennamen ein Schlüsselwort für den Geltungsbereich der Variablen angegeben werden. Beispiel: `\[+t global meintext]...` Dadurch entfällt die Notwendigkeit, die Variable vorher zu deklarieren, wenn sie nicht nur im Kontext des Makros sichtbar sein soll.
9. ShortCut erkennt ungültige Zeichen in Makros nun besser und entfernt sie beim Import.

3.6.1.0

1. Die Funktion `windowpos()` akzeptiert nun ein weiteres Argument, mit dem die Lage des betreffenden Fensters beeinflusst werden kann. So kann beispielsweise ein Fenster dauerhaft im Vordergrund gehalten werden.
2. Wenn ein leere geöffnete Kategorie gelöscht wird, so wird die unmittelbar darüber liegende Kategorie nicht mehr geöffnet.
3. Einige Localizer (Texte in ShortCuts Dialogen) wurden korrigiert oder verbessert.
4. Wenn eine **Verzeichnis-Kategorie** über den Kontext-Menübefehl **Aktualisieren** neu geladen wird, so wird ein Fortschrittsbalken angezeigt, wenn der Vorgang voraussichtlich länger dauern wird.
5. Listen haben als Ergänzung zu `last` nun die **Listenmethode** `first`. Einfache Variablen verfügen nun auch (analog zu den entsprechenden Listenmethoden) über die **Variablenmethoden** `first`, `last`, `draw`, `drop`, und `clear`. Methoden werden prinzipbedingt schneller ausgeführt als entsprechende Funktionen oder Ausdrücke mit Indizierung. Sie können nur auf Variablen angewendet werden, nicht auf Ausdrücke.
6. Bei der **Methode** `delete` kann nun auch bei Listen als zweites Argument die Anzahl der zu löschenden Elemente angegeben werden.
7. Konstruktionen wie `global A:=(3+4);` werden nicht mehr als Funktionsdefinitionen missverstanden. `global f(a,b) {...}` ist nach wie vor eine zulässige verkürzte Schreibweise von `global function f(a,b) {...}`.
8. Die Funktionen `inc()` und `dec()` sowie die **Präfixe und Postfixe** `++` und `--` setzen jetzt wieder den Zeichenkettenanteil des Ergebnisses auf den erzeugten Zahlenwert, so dass sie beispielsweise in `fill()` sinnvoll verwendet werden können.
9. Beim automatischen Import neuer Systemmakros nach der Installation einer neuen Version sind die Optionen, mit denen alle Makros oder Applikationen gelöscht werden können, jetzt deaktiviert und nicht aktivierbar. Anwender können dadurch beim Update ihre Makros nicht mehr versehentlich löschen.
10. Mit der neuen Funktion `filelink()` kann eine Dateiverknüpfung angelegt werden.

3.6.0.0

1. Um **gemeinsam genutzte Textbausteinsammlungen** besser zu unterstützen, kann man jetzt - über das Kontextmenü der Makroliste - ein Verzeichnis im Dateisystem als Kategorie importieren (Befehl **Neues Verzeichnis**). Voreingestellt werden aus dem gewählten Verzeichnis alle Dateien importiert, die die Endung `*.txt` und `*.rtf` haben. In den **Makro-Optionen** (über die Registerkarte **Steuerung** zugänglich) lässt sich das Intervall definieren, in dem die Verzeichnis-Kategorien aktualisiert werden sollen. Ferner gibt es für Verzeichnis-Kategorien im Kontextmenü den Befehl **Aktualisieren**.
2. ShortCut installiert sich jetzt als Shell-Erweiterung, sofern man dies auf der Registerkarte **Allgemein** aktiviert hat und die erforderlichen Benutzerrechte besitzt. Ein Makro kann über das ShortCut-Menü im Kontext-Menü des Windows Explorers aufgerufen werden, wenn die entsprechende Option im **Kommentar-Dialog** gesetzt ist. Das Makro kann über **MARGS** auf die im Windows Explorer ausgewählten Dateien zugreifen.
3. ShortCut erkennt formatierten Text jetzt automatisch, unabhängig davon, wie er in ShortCut als Textbaustein angelegt wurde. Man kann formatierten Text aus einer Datei einlesen (`*.rtf`), per Drag and Drop zur Kürzelliste hinzufügen oder direkt als Ersatztext angeben. Das Systemmakro `$richtext` wird nicht mehr benötigt und sollte gelöscht werden (ShortCut leert es beim Start). Formatierter Text wird immer über die Zwischenablage eingefügt, sofern man das Kürzel nicht per Drag and Drop in die Anwendung zieht.
4. Drag and Drop von Kürzeln in andere Anwendungen wurde verbessert. ShortCut bietet der Zielanwendung nun stets das richtige Textformat an (reinen Text oder formatierten Text). Ferner kann die Textausgabe nicht mehr dadurch blockiert werden, dass vorher ein Makro ausgeführt wurde, das mit der **Option** `-o` die Textausgabe unterdrückt hat.
5. Die Funktion `dirdialog()` öffnet jetzt den Dialog zur Verzeichnisauswahl direkt (ohne Zwischendialog). Aus diesem Grund ist das dritte Argument der Funktion (der Dialogtitel) entfallen. Der zurückgegebene Pfad endet nun immer auf einen Backslash, es sei denn, der Dialog wurde abgebrochen.
6. Die Funktion `filedialog()` sowie alle Dateidialoge in ShortCut zeigen jetzt auf der linken Seite die benutzerspezifischen Verzeichnisse (wie Desktop, Arbeitsplatz) an und können in der Größe geändert werden.
7. Wenn reiner Text mit ShortCuts eingebautem Editor bearbeitet wird, ist der Button **Ausführen** jetzt deaktiviert.
8. Die Funktion `execmacro()` akzeptiert jetzt anstelle eines Kürzels auch eine Liste. Das erste Element einer solchen Liste wird als Makrokürzel aufgefasst, die restlichen Elemente werden als Argumente interpretiert, die dem aufzurufenden Makro übergeben werden sollen. Das aufgerufene Makro kann über die Variable **MARGS** auf die Argumente zugreifen.
9. Bei der Vervollständigung sowie beim Aufruf der Funktion `popup()` können nun die Funktionstasten F1 bis F24 verwendet werden, um den selektierten Eintrag in der Wörterliste auszuwählen. In diesem Fall wird bei der Vervollständigung das neue Systemmakro `$popup` aufgerufen, dem in **MARGS** [1] der selektierte Eintrag sowie in **MARGS** [2] der Code der gedrückten Taste (als Zeichenkette, `F1="112"`) übergeben werden. Der tatsächlich auszugebende Text kann in `$popup` mit `return` zurückgegeben werden. Wird die Wörterliste dagegen mit der Funktion `popup()` geöffnet, wird das Makro `$popup` nicht aufgerufen. Wird eine Funktionstaste verwendet, um den in der Wörterliste selektierten Eintrag auszuwählen, so findet keine Leerzeichenersetzung statt, unabhängig davon, ob die Wörterliste durch die Vervollständigungsfunktion oder durch die Funktion `popup()` geöffnet wurde.
10. Das Kontextmenü der Makroliste enthält jetzt den Menüpunkt **Verschieben**, mit dem das selektierte Makro an den Anfang einer anderen Kategorie verschoben werden kann. Selbstverständlich können Makros nach wie vor per Drag and Drop verschoben werden.
11. Die folgenden neuen **Konstanten** sind verfügbar: **BUILD**, **USERNAME**, **USERLIST** und **MACADDRESS**.
12. Der Dialog **Makrotext-Optionen** heißt jetzt **Makro-Optionen**. Die Option **Klick auf Tray-Icon (de)aktiviert Makros** heißt nun **Klick auf Tray-Icon (de)aktiviert Kürzelerkennung** und befindet sich jetzt im Dialog **Makro-Optionen**, anstatt auf der Registerkarte **Allgemein**.

3.5.9.9

1. Die Konfigurationsdatei ShortCut.ini wird nicht mehr nur nach einem Versionswechsel von alten Makros bereinigt, sondern auch, wenn man ein oder mehrere Makros mit dem entsprechenden **Dialog** löscht.
2. Wenn auf der Registerkarte **Allgemein** die Option **Klick auf Tray-Icon (de)aktiviert Makros** ausgewählt ist, so wird die Makroliste nun in einer anderen Farbe dargestellt, wenn die Kürzelauflösung per Klick auf das Tray-Icon deaktiviert wurde.
3. Einfache Variablen haben nun die zusätzlichen **Methoden** `compress` und `uncompress`, mit denen Zeichenketten komprimiert bzw. entpackt werden können. Den Methoden kann optional ein Dateipfad übergeben werden, um das Ergebnis in der angegebenen Datei zu speichern.
4. Die verfügbaren Methoden von **einfachen Variablen** und **Listen** sind nun besser dokumentiert.
5. Das mitgelieferte Makro `$statistics` wurde aktualisiert. Die Systemmakros `$init` und `$exit` sind wie angekündigt nicht mehr in der Importdatei `AutoImport.ini` enthalten und stehen dem Anwender voll für eigene Zwecke zur Verfügung.

6. Die Richtungstasten deaktivieren nun die Erkennung eines Kürzels, das per Leertaste, Tabulatortaste oder Eingabetaste aufgelöst werden könnte.
7. Die Option, ein Makro automatisch aufzulösen, schließt die manuelle Auflösung per Leertaste, Tabulatortaste oder Eingabetaste nun aus, da die beiden Auflösungsvarianten nicht parallel verwendet werden können. Die Häkchen im Optionen-Dropdown werden automatisch entsprechend entfernt.
8. Der Austausch von Makros und anderer Nachrichten über das Netzwerk wurde verbessert. ShortCut sendet Nachrichten nun an die Arbeitsgruppe oder Domäne. Da die Nachrichten selbst den Empfänger beinhalten, werden sie dennoch nur von diesem ausgewertet. Ferner werden Makros oder statistische Daten, die eine ShortCut-Instanz versendet, weil sie angefordert wurden, nun ab ca. 6 KB komprimiert, sofern das anfordernde ShortCut mitgeteilt hat, dass es Kompression unterstützt (was ab Version 3.5.9.9 geschieht).
9. Die neue Funktion `execresults()` ermöglicht es, die Ergebnisse von Makros, die mit `remoteexec()` oder per `Lizenzdialog` auf anderen Computern ausgeführt wurden, nachträglich zu ermitteln. Ferner können Makros mit `remoteexec()` oder per `Lizenzdialog` nicht mehr nur auf einem, sondern auf beliebig vielen Computern ausgeführt werden. Voraussetzung hierfür ist, dass die Passwörter (des sendenden und der empfangenden ShortCuts) für die Ausführung übereinstimmen.
10. Die Funktionen `index()` und `pos()` haben das weitere (optionale) Argument `start` erhalten, mit dem die Position des Listenelements bzw. Zeichens angegeben werden kann, ab der mit der Suche nach einem Element bzw. dem Suchbegriff begonnen wird.
11. Die **vordefinierte Variable** `SUFFIX` enthält in ihrem ganzzahligen Anteil nun auch dann den Makroindex, wenn das Makro über ShortCuts eingebauten Editor ausgeführt wird.
12. Mit der neuen **Konfigurationsoption** `IgnoreKeyMismatch` in der Sektion `Completion` von ShortCuts Konfigurationsdatei `ShortCut.ini` kann festgelegt werden, ob die Wörterliste, die beim Aufruf von `popup()` oder bei der **Vervollständigung** angezeigt wird, geöffnet bleibt, wenn der letzte Tastendruck zu keinem der Einträge in der Wörterliste passt. Voreingestellt wird die Liste in diesem Fall geschlossen.
13. Wenn man ein gelöscht Makro wiederherstellt, so wird es nun unterhalb des ausgewählten Makros eingefügt und nicht an die Makroliste angehängt.
14. Die Update-Prüfung findet jetzt nur noch in dem auf der Registerkarte **Info** angegebenen Intervall statt.
15. Vor der zeitversetzten Ausführung eines Makros (beispielsweise `$paste`) erzwingt ShortCut nicht mehr, dass die Steuertasten losgelassen werden. Hält man z.B. `Strg+v` gedrückt, so wird der Einfügevorgang so lange wiederholt, bis diese Tastenkombination tatsächlich losgelassen wird.
16. Wenn der Pfad des Backup-Verzeichnisses nicht auf einen Backslash endet, wird jetzt ein Backslash angehängt.
17. Durch ein Build-Tag kann ShortCut nun auch über die Update-Funktion aktualisiert werden, wenn sich die Versionsnummer nicht geändert hat.
18. Die Funktion `windowlist()` liefert nun auch Fenster mit leerem Titel, wenn ihr als Argument der reguläre Ausdruck `*` übergeben wird. Wird ihr kein Argument übergeben, liefert sie die Liste aller Fenster, deren Titel nicht leer ist.
19. Die Funktion `dateval()` kann nun auch ohne Argument aufgerufen werden und liefert in diesem Fall die Fließkommazahl, die Datum und Uhrzeit des Rechners repräsentiert.
20. Auf den **Kommandozeilenparameter** `-i` kann nun optional der Pfad der Konfigurationsdatei `ShortCut.ini` als weiterer Parameter folgen. Dieser Pfad kann auch als Wert der **Konfigurationsoption** `BaseIni` in der Datei `ShortCut.ini` im Installationsverzeichnis angegeben werden.

3.5.9.8

1. Nach der Ausführung von **Makrofunktionen**, mit denen Makros geändert werden, wird nun die Makroliste aktualisiert.
2. Die Funktionen `macro`, `addmacro`, `macrotext` und `copymacro` verarbeiten die gegebenenfalls übergebenen Flags für die Auslösetasten nun richtig.
3. Wenn per Kontextmenü der Makroliste ein neues Makro aus einem bestehenden angelegt wird, erhält es nun auch die Programmdefinition des Quellmakros.
4. Makrokürzel mit manueller Auflösung können als letzte Zeichen nun Leerzeichen enthalten. Leerzeichen bei der Eingabe des Kürzels werden im Kürzel selbst durch das Zeichen `-` (Alt+0172) repräsentiert (dies betrifft nur die Darstellung; Leerzeichen müssen als Leerzeichen eingegeben werden). Das Zeichen für die Darstellung von Leerzeichen in Kürzeln kann mit der **Konfigurationsoption** `ShortcutSpace` umdefiniert werden. Durch dieses Feature ist es möglich, ein Kürzel durch mehrfaches Drücken der Leertaste aufzulösen, jedoch nur, wenn das Kürzel nicht automatisch aufgelöst wird. Beispiel: Kürzel ist `"abc-"`; Option `"Leertaste"` ist ausgewählt; Option `"Automatisch auflösen"` ist nicht ausgewählt: Sie können nun `"abc"` eingeben und zweimal die Leertaste drücken, um das Kürzel aufzulösen.

5. Die Erstellung von Backups ist nun fester Bestandteil von ShortCut und muss nicht mehr per Makro realisiert werden. ShortCut legt beim Start ein Backup der Konfigurationsdatei ShortCut.ini in dem Verzeichnis an, das auf der Registerkarte **Allgemein** festgelegt wurde. Voreingestellt werden in diesem Verzeichnis 100 Backups gehalten. Das Backup-Verzeichnis sowie die Nummer des nächsten Backups können mittels **Konstanten** abgefragt werden. Mit der **Konfigurationsoption** BackupMode kann eingestellt werden, ob auch vor jedem Speichern ein Backup erstellt werden soll.
6. Die Systemmakros \$init und \$exit enthalten nun keine Zusatzfunktionalität mehr und werden ab der nächsten Version aus der Importdatei AutoImport.ini entfernt. Wenn Sie diese Datei importieren und die Makros \$init und \$exit geändert haben, gehen Ihre Änderungen verloren.
7. Die automatische Verifikation (Freischaltung) ist jetzt voreingestellt abgeschaltet. Ferner können Administratoren ShortCut nun auf anderen Rechnern freischalten. Details hierzu sind auf Anfrage verfügbar.
8. Die Lite-Version darf im Netzwerk auf nicht mehr als zwei Rechnern gleichzeitig betrieben werden.

3.5.9.7

1. Bei der Übertragung von Makros und Applikationen über das Netzwerk (*Senden an, Anfordern von* im Kontextmenü) werden die Computer in der Arbeitsgruppe nun schneller ermittelt.
2. Der Body der Anweisungen list und lists kann nun in geschweifte Klammern eingeschlossen werden. Bisher konnten diese Anweisungen nur mit einem Semikolon abgeschlossen werden.
3. In Kürzeln können nun die Platzhalter ? und @ mit dem Platzhalter * kombiniert werden. Beispiel: Das Kürzel .em?* verlangt, dass auf .em genau ein weiteres beliebiges Zeichen sowie bis zu 8 beliebige Zusatzzeichen folgen.
4. Neben der **vordefinierten Variablen** SUFFIX kann in Makros nun auch die Variable SHORTCUT genutzt werden. Sie beinhaltet die komplette als Kürzel eingegebene Zeichenkette. Das Kürzel des laufenden Makros erhält man mit macroshortcut (+SUFFIX).
5. Die Schleifenanweisung loop akzeptiert nun eine until-Klausel.
6. Innerhalb von Schleifen kann mit der neuen Anweisung continue der nächste Schleifendurchlauf begonnen werden.
7. Einträge in der Wörterliste, die mit der Funktion popup() geöffnet werden kann, können nun mit Spalten versehen werden.
8. Mit dem neuen **Konfigurationsschlüssel** HookClipboard kann ShortCuts Funktion als Clipboard-Viewer deaktiviert werden.
9. ShortCuts Textpuffer wird nun immer gespeichert, so dass beim nächsten Start der gleiche Stand vorliegt wie beim Beenden. Dies ist für die Vervollständigung relevant.
10. Der Textpuffer kann nun mit Hilfe der **eingebauten Variablen** TEXTBUFFER so ausgelesen werden, wie er ist (mit allen Trennzeichen etc.). Die Funktion textbuffer() liefert den Textpuffer wie bisher als Wörterliste.
11. Bei der **Synchronisation** der Konfiguration wird nun auch der Textpuffer berücksichtigt.
12. Die Handhabung der Datei AutoImport.ini wurde verbessert. Voreingestellt wird dem Benutzer bei einem Versionswechsel angeboten, die darin enthaltenen Elemente zu importieren. Dieses Verhalten kann mit dem neuen **Konfigurationsschlüssel** AutoImportMode geändert (auch abgeschaltet) werden. In dieser Version enthält AutoImport.ini neue Fassungen der Makros \$init und \$exit, da diese nicht mehr das Einlesen und Speichern des Textpuffers übernehmen müssen. Ferner enthält \$init Code, mit dem beim Start von ShortCut Backups der Konfiguration (ShortCut.ini) im Benutzerverzeichnis angelegt werden. \$init und \$exit rufen die Makros \$init+ und \$exit+ auf, in denen der Benutzer eigenen Code unterbringen kann. Diese beiden Makros gehören zur Standardausstattung, werden in der Lite-Version also nicht zu den Benutzermakros gerechnet.

3.5.9.6

1. Die Kürzelerkennung wurde nochmals verbessert.
2. In den **Makrotext-Optionen** gibt es den neuen Punkt **Bei manueller Auflösung Leerraum nach Kürzel ignorieren**.
3. Im **Optionen**-Menü eines Makros kann man nun wählen, ob das Kürzel mit der Leertaste, der Tabulatortaste oder der Eingabetaste aufgelöst können werden soll - zusätzlich zum Hotkey, der auf der Registerkarte **Steuerung** angegeben ist.
4. Die neue **Konstante** LASTKEYDOWN liefert den Tastencode der zuletzt gedrückten Taste. Damit lässt sich feststellen, mit welcher Taste ein Kürzel manuell aufgelöst wurde.
5. Im Textmodus müssen Makro-Anweisungen nun immer mit < oder { begonnen werden. \< und \{ sind nicht mehr zulässig.

3.5.9.5

1. Sich überschneidende Kürzel (beispielsweise `aa`, `aaa`, und `aa*`) werden nun besser erkannt.
2. In den **Makrotext-Optionen** ist der Punkt **Platzhalter in Kürzeln zulassen** entfallen (Platzhalter sind immer erlaubt und werden immer ausgewertet).
3. Ebenfalls in den **Makrotext-Optionen** ist die Option **Platzhalter (*) ersetzt möglichst viele Zeichen** hinzugekommen. Sie bewirkt, dass ShortCut zuerst versucht, möglichst viele der eingegebenen Kürzelzeichen dem Platzhalter zuzuordnen anstatt möglichst wenige. Letzteres ist das bisherige Verhalten und die Voreinstellung.
4. Die Option **Mehrfache Auflösung von Kürzeln ermöglichen** (auf der Registerkarte **Steuerung** ist entfallen. Diese Option stammte aus der Zeit, als ShortCut noch keine Hotkeys kannte.
5. Auf der Registerkarte **Allgemein** kann nun eingestellt werden, ob ShortCut beim Start von Windows ausgeführt werden soll.
6. Die Optionen **Tooltips** und **Tray-Icon** auf der Registerkarte **Allgemein** funktionieren nun wieder.
7. Im Tray-Icon-Menü gibt es den neuen Menüpunkt **Makro anhalten**. Dieser Menüpunkt hat die gleiche Funktion wie die Pause-Paste während der Ausführung eines Makros. Da in Endlosschleifen unter Umständen jedoch die Tastatur blockiert ist, kann man mit diesem Menüpunkt das betreffende Makro doch noch beenden.
8. Die neue Funktion `intersect()` liefert die sich überschneidenden Elemente zweier Listen.
9. Der **Ganzzahl-Operator** `#` kann nun auch in Ausdrücken verwendet werden.
10. Die Funktion `wait()` gibt es nun auch als Anweisung `wait`.

3.5.9.4

1. Die Makroliste wird wieder aktualisiert, wenn man ein Makro manuell löscht.
2. Mit der neuen Funktion `copymacro()` können neue Makros als Kopie eines bestehenden an beliebiger Stelle in die Makroliste eingefügt werden.
3. Die neue Funktion `categorylist()` liefert die Liste der Makrokategorien.
4. Die neue Funktion `macrocategory()` liefert den Titel und den Index der Kategorie, zu der ein Makro gehört.
5. Mit der neuen Funktion `categorymacros()` können die zu einer Kategorie gehörenden Makros ermittelt werden.
6. Die Funktionen `macro()` und `addmacro()` behandeln Kategorie-Makros nun richtig.
7. Die Funktion `wait()` liefert nun die vergangene Zeit.
8. Die in Version 3.5.9.2 eingeführte Verlegung der Statistikdatei in das Benutzerverzeichnis funktioniert nun zuverlässiger.
9. Es ist nicht mehr möglich, eine Liste durch Redefinition in eine einfache Variable umzuwandeln. `A[] := (); A:=5;` führt zu einem Fehler.

3.5.9.3

1. Mit den **speziellen Zeichenfolgen** `\0..` sowie `\(0..)` können Sonderzeichen als Kombination aus gedrückter Alt-Taste und einer darauf folgenden Dezimalzahl ausgegeben werden. In den meisten Anwendungen lassen sich damit Unicode-Zeichen ausgeben.
2. Die Funktion `httpget()` liefert nun das korrekte Ganzzahl-Ergebnis.
3. Im Kontextmenü der Makroliste befindet sich nun der Menüpunkt **Löschen mit Auswahl**, der einen Dialog zum Löschen mehrerer Makros öffnet.

3.5.9.2

1. Der Tooltip in der Makroliste wird nun auch angezeigt, wenn die Wartezeit auf der Registerkarte **Allgemein** auf mehr als 1000 ms eingestellt ist.
2. Drückt man im eingebauten **Texteditor** die Escape-Taste, so wird der Benutzer gefragt, ob er seine Änderungen speichern möchte, falls er Änderungen am Makrotext vorgenommen hat. *Abbrechen* dagegen verwirft wie bisher alle Änderungen sofort, auch wenn der Text zwischendurch gespeichert wurde.
3. Löscht man in einem Makro mit `deletemacro()` ein Makro, so werden der Titel und das Kürzel des aktuellen Makros nicht mehr geändert.
4. Die Funktion `deletemacro()` hat nun ein weiteres Argument, mit dem festgelegt werden kann, ob das zu löschende Makro wiederhergestellt können werden soll.
5. Die Option **Konfigurationsfenster ab dem zweiten Start öffnen** ist entfallen. Wird ShortCut ausgeführt, während das Programm bereits läuft, wird das Konfigurationsfenster jetzt immer geöffnet.
6. Statistische Daten über den Gebrauch von Makros und Applikationen werden jetzt nur noch dann in ShortCuts Installationsverzeichnis abgelegt, wenn der Benutzer Schreibzugriff auf dieses Verzeichnis hat (bisher wurde eine Fehlermeldung angezeigt, wenn dies nicht der Fall war). Andernfalls werden diese Daten im Benutzerverzeichnis gespeichert. Durch diese Änderung benötigen Benutzer keinen Schreibzugriff mehr auf das Installationsverzeichnis, wodurch bei Computern mit mehreren Benutzern eine zentrale ShortCut-Installation ausreicht.

3.5.9.1

1. In Kürzeln kann als weiterer Platzhalter neben ? und * nun auch das @-Zeichen verwendet werden. Es steht für eine einzelne Ziffer. Das Kürzel `#ab@` beispielsweise deckt also die Eingaben `#ab0` bis `#ab9` ab.
2. Nach *Speichern* im eingebauten **Texteditor** wird der Makrotext nun richtig umgewandelt und vergrößert dadurch die Konfigurationsdatei *ShortCut.ini* nicht unnötig.
3. In die Konfigurationsdatei *ShortCut.ini* können mit der Anweisung `#include` nun kundenspezifische Parameter aufgenommen werden. Diese Parameter werden nicht gespeichert, wenn die Konfigurationsdatei gesichert wird. Voreingestellt wird die Datei *UserSettings.ini* inkludiert.
4. Mit Hilfe einer Proxy-Liste können weitere Proxy-Server für ShortCuts HTTP-Zugriffe (Freischaltung, Update-Prüfung, Downloads) definiert werden. Diese Liste wird voreingestellt aus der Datei *UserSettings.ini* gelesen und kann auf der Registerkarte **Allgemein** deaktiviert werden.

3.5.9.0

1. Auf der Registerkarte **Info** kann jetzt ein Intervall für die Suche nach Updates angegeben werden. Zu dieser Registerkarte gibt es jetzt eine Hilfeseite.
2. Im eingebauten **Texteditor** wird der angezeigte Ausschnitt beim Drücken der Eingabe- oder Tabulatortaste nur noch dann verschoben, wenn sich die Eingabemarke am unteren Rand befindet.
3. Die neue Funktion `filedate()` ermöglicht es, das Änderungsdatum einer Datei zu ermitteln und zu setzen.