

FinalWrapper

NDY's

COLLABORATORS

	TITLE : FinalWrapper		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY	NDY's	July 19, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	FinalWrapper	1
1.1	Hitchhikers guide to FinalWrapper	1
1.2	Was zum Teufel ist FinalWrapper?	1
1.3	Installation	2
1.4	Bemerkung zur Installation	2
1.5	Bedienungsanleitung	2
1.6	Bemerkung zur Benutzung	6
1.7	Bemerkung zur Benutzung	7
1.8	Bemerkung zur Benutzung	7
1.9	Bemerkung zur Benutzung	7
1.10	Programmiererecke	7
1.11	Bekannte Fehler und Einschränkungen	9
1.12	Die Geschichte	10
1.13	Bemerkung zur Geschichte	11
1.14	Die Zukunft	11
1.15	Gesetzliches	12
1.16	Adresse des Autors	13
1.17	Danksagungen	14

Chapter 1

FinalWrapper

1.1 Hitchhikers guide to FinalWrapper

```
*****
*                               *
*   FinalWrapper 1.3 (27.04.94) by NDY's   *
*                               *
*****
```

```
Was zum Teufel ist FinalWrapper?
Installation
Bedienungsanleitung
Programmiererecke
Bekannte Fehler und Einschränkungen
Die Geschichte
Die Zukunft
Gesetzliches
Adresse des Autors
Danksagungen
```

1.2 Was zum Teufel ist FinalWrapper?

Kurze Übersicht

FinalWrapper ist ein FinalWriter(TM) Makro, das einen Textblock oder einen Textausschnitt um einen Kreis herumlaufen lässt (engl. to wrap = [ein]wickeln). Es können auch kinderleicht Spiralen erzeugt werden!

Merkmale:

- Unterstützt alle Arten von Textattributen und Schriftarten
- Unterstützt alle möglichen Textumlaufsarten
- Unbeschränkte Länge des Textausschnitts
- Unterstützt gedrehte Ovale
- Zusätzliche Effekte mit Rotation der Buchstaben

- Oval zum Textobjekt auf verschiedene Arten hinzugruppieren
- Eingebaute Hilfefunktion
- Benützt die eingestellte Sprache (im Moment Englisch/Deutsch verfügbar)
- Kann leicht in andere Sprachen übersetzt werden (auch unter OS2.0 verfügbar ←
!)
- Verschiedene Spiralentypen
- Automatische/einstellbare Verkleinerung der Buchstaben gegen's ←
Spiraleninnere

1.3 Installation

Installationsanleitung

Kopieren Sie einfach FinalWrapper in Ihr FWMacros Verzeichnis. (Ich schlage ←
vor,
Sie installieren es im Benutzermenu bzw. der Iconleiste!)

Die "rexxmathlib.library" muss sich im LIBS: Verzeichnis befinden. Daneben ←
muss
natürlich Rexxmast laufen und es wäre durchaus nicht schlecht, wenn sie ←
Final
Writer besitzen würden B-)!

Wenn Sie nicht wissen, wie Sie das alles bewerkstelligen sollen, oder einfach ←
zu
faul dazu sind, es selbst zu tun (wie ich es wäre 8-]), können Sie ←
das
beiliegende Installer-Skript benützen, das Ihnen alle Arbeit abnimmt. Um es ←
zu
benützen, brauchen Sie Commodore's Installer. Er befindet sich auf der OS3 ←
.0+
Install Diskette und ist auch via PD erhältlich (Fish Disk 870).\$^1\$

Installation starten

1.4 Bemerkung zur Installation

Anmerkung:

Das Skript verwendet intern den Befehl Version, der wiederum die version. ←
library
braucht. Beides befindet sich auf Ihrer Workbench, falls Sie es nicht ←
gelöscht
haben.

1.5 Bedienungsanleitung

Bedienung

(Es ist nicht notwendig, die Fussnoten zu lesen, um das Programm zu benutzen;
sie geben lediglich ein paar zusätzliche Informationen.)

* Bevor das Makro aufgerufen wird

Zuerst sind ein Oval und ein Text auszuwählen. Dabei ist es gleichgültig, ob
dies ein Textblock oder ein sonstiger Textabschnitt ist. Der ganze Text wird um
das Oval herumgeführt.

Normaler Text hat den Vorteil, dass eine unbeschränkte Anzahl von Zeichen in
verschiedenen Schriftarten, Farben etc. verwendet werden kann. Bedingt durch den
verwendeten Algorithmus ist es aber einiges langsamer als ein Textblock, der
jedoch höchstens 33 Zeichen fasst und nur einen Stil haben kann¹.

War das Oval gedreht, wird der Kreistext ebenfalls gedreht.

Also: Text auswählen, den "Mauspfeil"-Schalter anklicken und das Oval auswählen,
resp. Textblock auswählen, Shift Taste drücken und auf das Oval klicken².
Jetzt
kann FinalWrapper gestartet werden.

* Zur Benützung des Requesters

Nun erscheint ein Requester, in den man alle benötigten Optionen eingeben kann
(Ok und Abbruch funktionieren wie gewohnt). Es ist auch möglich, nichts
einzugeben, um die voreingestellten Werte zu verwenden. Alle Optionen müssen
durch eine Leerzeichen (Space) voneinander getrennt werden³.
Unbekannte
Optionen werden ignoriert.

* Sektorgrösse: $A_n = n / A_{-n} = -n$ (Voreinstellung: 360)

Die erste Option gibt die zu verwendende Sektorgrösse in Grad an. Dies erlaubt
es, den Text nur um einen Teil des Ovals herumlaufen zu lassen. Sie müssen eine
Zahl zwischen -360 und 360 eingeben⁴. Negative Zahlen drehen die
Schreibrichtung um, die Unterseite der Buchstaben schaut dann nach aussen, statt
nach innen wie bei positiven Zahlen (Probieren geht über Erklären!). Wenn kein
"a" vorangestellt wird, muss diese Option als erste angegeben werden. Falls die

Option fehlt oder ein ungültiger Wert steht, wird die Voreinstellung von 360 verwendet. Zu grosse oder zu kleine Werte werden in 360 bzw. -360 umgewandelt. ("a" steht für engl. "angle" = Winkel oder "arc" = Bogen.)

Bsp.: 360 Den ganzen Kreis verwenden (a360/a+360/A+360/A360 ist dasselbe)
)
 180 Nur ein Halbkreis wird gebraucht
 -40 Nur ein kleiner Bogen ist zu verwenden

* Löschen: D+ / D- / D= (Voreinstellung: Oval löschen)

Das Oval wird normalerweise gelöscht, dies kann aber durch die "d" Option geändert werden: "d+" löscht das Oval und den Textblock, wogegen "d-" keines von beiden löscht. Dies verlangsamt die Berechnung; besser ist es daher, Sie verwenden die "d=" Option, die das Oval kopiert, sodass sie es nachher wieder einfügen können, sobald es gebraucht wird. ("d" steht für "delete" = löschen.)

* Rotation: R+n / R-n / R+ / R- / R= (Voreinstellung: R+0)

"r\ensuremath{\pm}n" dreht alle Buchstaben um weitere n Grad nach rechts (+) bzw. links (-).
 "r\ensuremath{\pm}" ist eine Abkürzung für "r\ensuremath{\pm}90". n muss zwischen -180 und +180 liegen.

"rn" setzt die Drehung aller Zeichen gleich n (d.h. alle schauen in dieselbe Richtung). n wird relativ zur Drehung des Ovals gemessen, d.h. mit "r0" werden alle Zeichen parallel zur Ovalachse gezeichnet. "r=" funktioniert wie "rn", jedoch wird die Drehung der Zeichen gleich der des Textblocks (resp. 0 für Textausschnitte). n darf hier zwischen 0 und 360 liegen.

Bsp.: r+90 Die Buchstaben bilden eine "Kolonne"
 r+180 Wie "a-360", aber die Schreibrichtung ist umgekehrt
 r180 Alle Zeichen erscheinen verkehrt herum

* Gruppierung: G- / G+ (Voreinstellung: Oval nicht gruppieren)

Die "g" Option erlaubt es, das Oval zum erzeugten Objekt hinzuzugruppieren. "l+" fügt das Oval wie es ist hinzu, wogegen "l-" es zuvor unsichtbar macht. Dies ist v.a. nützlich, wenn ein Sektor von weniger als 360 verwendet wird und der Haupttext um das ganze Oval herumlaufen soll.

* Startpunkt: $S_n = S+n / S-n / S+ / S-$ (Voreinstellung: S_0)

Mit dieser Option kann der Start des Textes festgelegt werden. Mit " s_0 " wird der Text relativ zur oberen Mitte des Ovals zentriert. Positive Werte (das + kann auch weggelassen werden) verschieben den Startpunkt um n Grad im Gegenuhrzeigersinn, negative dagegen im Uhrzeigersinn. " $s\text{\textbackslash ensuremath\{\pm\}}90$ " entspricht " $s\text{\textbackslash ensuremath\{\pm\}}90$ ". Dies erlaubt es, eine beliebige Anfangsposition für den Text zu verwenden, d. h. einen bestimmten Teil des Ovals zu benutzen. n muss zwischen -180 und 180 liegen.

Bsp.: s_0 Der Text beginnt auf der rechten Seite des Ovals
 $a_90 s_90 90\text{\textbackslash textdegree\{}}^\circ$ Winkel, zentriert zur linken Seite des Ovals

* Spiralen: $@_n = @+n / @-n$ (Voreinstellung: $@_0$)

Spiralen werden mit der " $@_n$ " Option erzeugt: n muss zwischen -100 und 100 liegen und gibt das Verhältnis vom inneren zum äusseren Radius an (in %). Mit einer negativen Zahl beginnt der Text im Spiraleninnern, ansonsten aussen. Mit " $@_n$ " beginnt die Spirale aussen, der Radius wird gegen innen immer kleiner, bis er am Ende des Textes $n\%$ des Anfangswertes erreicht. Die Zeichengrösse wird ebenfalls gleichmässig verkleinert (sofern die " h "-Option nicht angegeben wurde, siehe unten). Die Sektorgrösse ist für Spiralen unbegrenzt. " $@_0$ " entspricht " $@_{100}$ " bzw. " $@_{-100}$ " (keine Spirale).

Bsp.: $@_{30} a_{1080}$ Beginnt aussen, drei volle Kreise beschreibend, bis 30% der Originalgrösse erreicht wird
 $@_{-1} a_{720}$ Zwei volle Kreise, in der Mitte beginnend

* Höhe: $H_n = H+n / H-n$ (Voreinstellung: Wert von $@$, wenn angegeben, sonst H_0)

Die " h_n "-Option ist hauptsächlich in Verbindung mit Spiralen nützlich (obwohl sie natürlich auch sonst verwendet werden kann); sie bestimmt das Verhältnis zwischen der Höhe des ersten Zeichens und derjenigen des letzten. Sie funktioniert exakt gleich wie die " $@$ " Option. " h_0 " entspricht " h_{100} " und " h_{-100} ".

Der Zweck der Option ist es hauptsächlich, die Zeichengrösse in Spiralen konstant zu halten ("h0"), aber es kann natürlich auch eine stärkere oder weniger starke Reduktion der Zeichengrösse als des Radius' eingestellt werden.

Bsp.: "h-20" Der erste Buchstabe hat 20% der Originalgrösse und alle folgenden ein wenig mehr, bis der letzte schliesslich in der ursprünglichen Grösse gezeichnet wird.

* Optionen speichern: O+ / O- (Voreinstellung: Nicht speichern)

Wenn Sie gedenken, dieselben Optionen mehrmals zu verwenden (um damit herumzuprobieren), können Sie die "o" Option verwenden, um die aktuelle Eingabe zu sichern. Jedesmal, wenn Sie danach FinalWrapper starten, erscheinen wieder dieselben Optionen. Der Unterschied zwischen den beiden Optionen ist, dass "o" vor dem Speichern aus der Optionsliste entfernt wird und das Programm dann abbricht. Dies ist nützlich, wenn die Optionen nur einmal gespeichert werden sollen. "o+" speichert alle Optionen und fährt mit der Berechnung fort.

Bsp.:	Erster Aufruf	Bei allen weiteren Aufrufen erscheint:
	r+ s-90 d= o- a	R+ S-90 D= A (einfach die fehlende Zahl eingeben)
	120 d= s- o+ @120	120 D= S- O+ @120

* Hilfe: ? (Voreinstellung: Keine Hilfe)

Um eine kurze Erläuterung der Optionen zu erhalten, können Sie irgendwo "?" eingeben. In den erscheinenden Requestern kann man sich mit "Letzte Seite" und "Nächste Seite" durch die verschiedenen Textseiten durchklicken. "Zurück" bringt einen wieder zur Eingabe.

Bsp.: a120 g+?

1.6 Bemerkung zur Benutzung

Anmerkung:

Momentan werden Gross-/Kleinschreibung, Position und Schriftschnitt von Final Writer für Textblöcke nicht unterstützt.

1.7 Bemerkung zur Benutzung

Anmerkung:

Sie sollten nur ein Oval und einen Textblock auswählen. FinalWrapper ist es unmöglich, herauszufinden, welches Objekt Sie zuerst gewählt haben, es nimmt daher einfach das Oval und den Textblock mit der kleinsten internen Nummer. Wenn ein Textblock ausgewählt ist, wird evtl. selektierter Text ignoriert. Es kann (bedingt durch Final Writer) nicht nur ein einzelnes Zeichen ausgewählt werden (ausser es wird ein Textblock verwendet), aber das ist keine wirkliche Einschränkung, oder?

1.8 Bemerkung zur Benutzung

Anmerkung:

Wird eine Option mehrmals angegeben, wird nur die erste beachtet. Ist die Sektorgrösse einmal mit und einmal ohne "a" vorhanden, wird diejenige mit dem "a" benutzt.

1.9 Bemerkung zur Benutzung

Anmerkung:

Alle Zahlen sind Ganzzahlen, d.h. es kann kein Komma verwendet werden.

1.10 Programmiererecke

Informationen für Gurus und solche, die es gerne wären

Wenn Sie nicht mit Arexx vertraut sind, überspringen Sie diesen Abschnitt am besten.

* Der Source wurde in Unterroutinen aufgeteilt, um Ihnen (und in erster Linie mir) das Leben beim Erweitern von FinalWrapper zu erleichtern. Da der Code immer grösser wird, wurden die Kommentare auf ein Minimum reduziert.

* Das "PROC" vor allen Subs erlaubt es, diese mit GoldEd zu falten (in Konfig/Diverses Falt-Start/-Ende auf PROC/RETURN setzen). Dies wird von Arexx

ignoriert, da es vor dem Label steht (schlau, nicht? ß->).

* In der ersten Zeile der Unterprogramme sind allfällige Parameter aufgelistet, damit sie nicht ständig aufgeklappt werden müssen. Die Zahl in [Klammern] gibt die Version von FinalWrapper an, in der sie zuletzt geändert wurden. Dadurch wissen Sie, was sich geändert hat, wenn Sie eine neue Version erhalten. Ich schlage vor, Sie fügen einen Stern oder sowas dazu, wenn Sie eine Routine selber ändern.

Wie man das Programm in andere Sprachen übersetzt

Kopieren Sie die englischen Strings im "locale" Sub und übersetzen Sie diese. Entfernen sie keine Anführungszeichen ("), die sind wichtig! Einem Hochkomma (') muss ein weiteres Hochkomma folgen (d.h. verwenden Sie '' anstelle von ').

Die folgenden Platzhalter können in den Strings verwendet werden:

```
"errtext": %n Fehlernummer (=RC)
            %l Fehlerzeile (=SIGL)
            %t Fehlertext (Arexx liefert die Fehlertexte immer in Englisch,
            zumindest bis und mit OS3.0) (=ErrorText(RC))
"nolib":    %y Library-Name (=library)
"input":    %i Programminfo (=info)
            %d Defaultwerte (=default)
```

Bsp.:

```
info="FinalWrapper 1.3 by NDY's"
help.7="'Dies ist %i" "Es wurde verbessert von:" "Pit «Kojack» Langhaar"
wird zu: "'Dies ist FinalWrapper 1.3 by NDY's" "Es wurde verbessert..."
```

"helppages" kann nach Belieben für jede Sprache erhöht werden, falls nötig.

Entfernen Sie doch bitte meinen Namen nicht aus dem Infostring und fügen Sie diesen irgendwo ein...

Um eine neue Sprache auszutesten setzten Sie die Variable "test" gleich 1 und starten das Makro, dann setzen Sie sie gleich 2 usw. Die folgenden Dinge werden getestet:

```
test=1          Fehlerrequester
test=2          Library-nicht-gefunden-Requester
test=3          Falsche Objekte ausgewählt
test=4          Requester / Hilfsfunktion ("Ok" anwählen und ALLE
                Seiten
                ausprobieren)
test=10,20,100,200 FinalWriter Fehlercodes
```

Wenn alles klappt, können Sie "test" wieder auf 0 zurücksetzen.

FW liest seine Sprache aus der Variablen ENV:Locale, die von der locale. library
V40+ installiert wird. Wenn sie diese Version noch nicht besitzen (ich hab' sie
auch nicht!), keine Angst: das Programm ENV_Locale installiert sie und
ebenso
ihr Pendant im ENVARC:. Es wird vom Installationsprogramm automatisch
aufgerufen
und setzt die Variable entsprechend der eingestellten Systemsprache
(Prefs/Locale). Wenn sie FW übersetzen, sollten sie den Installationsschalter
in
dieser (oder besser der englischen) Dokumentation kopieren und " LANGUAGE
deutsch" entsprechend ändern (resp. in der englischen Anleitung hinzufügen
).
(Wenn Sie natürlich die ganze Anleitung und das Skript übersetzen (:-),
können
Sie den Schalter gleich dort einbauen.) Dies ermöglicht es auch OS2.0
Anwenden,
Ihre Sprache zu nutzen.

Wenn Sie mir Ihre Übersetzung oder eine verbesserte Version schenken, erhalten
Sie
meine neueste Version inklusive Ihrer Erweiterungen, sofern möglich.
übrigens:
Wer wäre bereit, mir jeweils die englische Dokumentation ins Deutsche
zu
übersetzen??? Ich brauche jedesmal viel Zeit dazu, die ich besser
zum
Programmieren verwenden könnte...

1.11 Bekannte Fehler und Einschränkungen

Bekannte Fehler und Einschränkungen

- * Die Ergebnisse mit flachen Ovalen sind nicht sehr gut, da der Abstand
zwischen
den Buchstaben dann in den engen Bögen zu klein wird. Mir scheint aber
dies
sowieso nicht nützlich und darum hab' ich mich nicht weiter darum bemüht, es
zu
verbessern. Benützen Sie einfach eine Sektorgrösse von 120\textdegree{} oder
so, dann sollte
es gut aussehen. Derselbe Effekt tritt auch bei Spiralen auf.
 - * Die Spiraleffekte sind noch nicht allzu perfekt.
 - * Sie sollten das Dokument nicht verändern, solange FinalWrapper arbeitet,
da
dies zu Fehlfunktionen führen könnte.
-

* Bei normalem Text wird die größe eines Leerzeichens vom letzten ↵
 benutzen
 Zeichensatz berechnet (ist schneller so!). Dadurch können möglicherweise ↵
 die
 Abstände zwischen den Worten etwas ändern. Ggf. müssen Sie Leerzeichen ↵
 einfügen.
 Tabulatoren und Absatzenden werden ebenfalls durch ein Leerzeichen ersetzt, ↵
 was
 den gleichen Effekt mit sich bringt.

* Ist das Oval rotiert und die Sektorgröße nicht um $\pm 360^\circ$ ↵
 textdegree{}, kann das erzeugte
 Objekt evtl. nicht exakt auf dem Oval liegen (mit g- funktioniert's perfekt).

FinalWrapper wurde mit Final Writer Release 1 (21.10.93, deutsche Version) ↵
 auf
 einem A1200 (OS3.0, 2MB Chip, 4MB Fast, HD) erstellt und getestet, es ↵
 sollte
 aber auch problemlos unter OS2.0/2.1 und anderen Hardwarekonfigurationen ↵
 laufen.

1.12 Die Geschichte

Geschichtliches

(Die Alpha- und Betaversionen etc. wurden nicht veröffentlicht)

- 1.0 (24.02.94) Erste Veröffentlichung
- 1.1a (06.03.94) Name von Wrap in FinalWrapper geändert
 Versionsstring korrigiert
 Benutzung von normalem Text und verschiedenen Schriften ↵
 möglich
 Textumlauf wird nun vom Oval übernommen
 D Option hinzugefügt
 Englisches Dokfile nun separat
- 1.1b (10.03.94) Fehlerbehandlungsroutine hinzugefügt
 Daraufhin RemLib Syntaxfehler korrigiert
 Test, ob kein Objekt ausgewählt ist
 Ungültige Zeichen (TABs, CRs etc.) werden durch Spaces ↵
 ersetzt
 Sektorgröße 6 nun auch möglich
 Sektorgröße limitiert auf -360 bis 360, Defaultwert 360
- 1.1c (11.03.94) RemLib wird jetzt auch im Fehlerfall aufgerufen
 SELECTED zu "SELECTED" korrigiert etc.
 Unterstützung von gedrehten Ovalen
 Einrückung reduziert -> Source um ~2.5k kleiner!
 "Falsche Eingabe"-Requester entfernt
- 1.1 (17.04.94) Fehler bei normalem Text behoben
 Bessere deutsche Dokumentation (jetzt gleich wie englische)
- 1.2a (18.03.94) R Option hinzugefügt
 L Option hinzugefügt
 Hilfsrequester hinzugefügt
- 1.2b (26.03.94) Unterstützt jetzt die bevorzugte Sprache der Locale-Prefs\$^1\$
 Zusammengesetzte Strings durch Punktvariablen (x.y) ersetzt

- Mehrere Textblöcke nicht möglich (Auswahlreihenfolge unbekannt) ←
- Benutzt jetzt Datatype() statt komplizierten Verify()s
S Option hinzugefügt
- 1.2 (27.03.94) Dokumentation jetzt im AmigaGuide-Format
Buchstabenobjekte werden im Fehlerfalle gelöscht
Installer-Skript hinzugefügt
- 1.3a (02.04.94) @/h Optionen hinzugefügt (inspiriert bei CBM's Bullet-Lib Demo) ←
- Dummen Fehler entfernt wenn "Abbruch" gewählt wurde
Prozedurköpfe geändert -> können mit Golded gefaltet werden
Installer-Skript kann nun aus der Anleitung gestartet werden
- 1.3b (03.04.94) Ganzer Source neustrukturiert (wieder ~2k gespart!)
Komma wird auch im Fehlerfall wieder eingestellt
Alle druckbaren Zeichen funktionieren jetzt (;=" gingen nicht ←
-)
- Hilfsseiten werden in der richtigen Reihenfolge angezeigt
Das Installer-Skript fragt jetzt vor dem Überschreiben
- 1.3c (13.04.94) d= Option hinzugefügt
Fehler in der Hilfe entfernt: Zurück funktionierte nicht ←
- immer
- 1.3d (14.04.94) O Option hinzugefügt
L Option heisst jetzt G (einleuchtender)
Teile des Handbuchs überarbeitet
Hilfe löscht den Requesterinhalt nicht mehr
- 1.3 (27.04.94) Benutzt jetzt die in ENV:Locale eingestellte Sprache
ENV_Locale hinzugefügt
Installer-Skript prüft jetzt, ob ältere Version vorhanden
- Wurde ausgiebig getestet, sollte zuverlässig laufen
(alle Optionskombinationen kann ich natürlich nicht testen ←
- ...)

1.13 Bemerkung zur Geschichte

Anmerkung:

Deshalb gibt es von nun an keine spezielle deutsche Version mehr, d.h. ←
alle
Kommentare sind englisch, aber ich denke, das sollten eh alle ←
Programmierer
verstehen...

1.14 Die Zukunft

Zukünftiges

Die nächste Version wird vermutlich folgendes enthalten:

- Requester mit Schalter für alle Optionen
- Schnellere Bearbeitung von normalem Text
- Bessere Ergebnisse mit flachen Ovalen und Spiralen (ich hab' da 'ne Idee, ' ←
mal
seh'n wie's läuft!)

- Andere nette kleine Verbesserungen

Vielleicht haben Sie ja eine gute Idee? Dann zögern Sie nicht, mir zu schreiben! ↵

Wenn Sie einen Fehler in FinalWrapper gefunden haben oder eine gute Idee für eine Verbesserung haben, schreiben Sie mir; Sie erhalten dann eine neue Version, in der der Fehler behoben oder die Verbesserung - falls möglich - eingebaut ist. ↵

Bitte legen Sie in jedem Falle eine Disk (mit cooler Soft drauf?) und etwas Geld (sfr., DM, US\$) fürs Rückporto (resp. Geld für beides) bei, sonst kann ich nicht garantieren, dass Sie Antwort erhalten! ↵

Andere Makros

Ich habe noch folgende Final Writer Makros im Kopf:

- Funktionsplotter mit zig Optionen
- Sinusschrift (yeah!)
- Makro, dass den ausgewählten Text zu einem Textblockobjekt macht (↵ mit verschiedenen Schriften und Stilen, nützlich für Bildunterschriften usw.)
- Effekte mit Neigung (/Text\ \Text/ etc.)
- Effekte mit Textgrösse
- Was immer Sie vorschlagen

1.15 Gesetzliches

Gesetzliches

* Haftung:

Sie benutzen das Programm auf Ihr eigenes Risiko. Weder ich noch irgendein anderer Autor von irgendeiner beigefügten Datei kann für etwaige durch dieses Programm verursachte Schäden zur Verantwortung gezogen werden. ↵

Im Speziellen nicht, wenn...

- Ihr Computer wegen FinalWrapper abstürzt und 10 Stunden Arbeit an ihrem atemberaubenden Ballerspiel zunichte gemacht werden. ↵
- Ihre Katze die Disk mit FinalWrapper gegessen hat und krank geworden ist.
- Das lokale Kraftwerk beim Benutzen von FinalWrapper zusammengebrochen ist (↵ ein bekannter Fehler, der nur sehr sehr selten auftritt)!
- Sie vor lauter Erstaunen über die mit FinalWrapper erzielten Ergebnisse den Mund nicht mehr zubringen (betrachten sie einen PC, das sollte sie wieder ↵

ernüchtern!)

3-) ↔

* Vertrieb:

Dieses Programm ist Public Domain, d.h. es darf nach Belieben benutzt
und
verändert werden. Alle Dateien sollten nur zusammen weitergegeben
werden.
Zumindest sollten aber die rexxmathlib.library und eine der ShortInfo
Dateien
mitgegeben werden.

Folgende Dateien sind enthalten:

FinalWrapper.rexx	+ .info	Das Makro
FinalWrapper.Guide	+ .info	Englisches AmigaGuide Dokument
FinalWrapperD.Guide	+ .info	Deutsches AmigaGuide Dokument
ShortInfo.doc		Englischer Kurzbeschreib
ShortInfo.dok		Dasselbe in Deutsch
FinalWrapper.install		Installer-Skript
DemoDoc	+ .info	Final Writer Beispieldokument
ENV_Locale		Vom Install-Skript gebraucht
E/ENV_Locale.e		E Source des Programms \ Nützlich nur für
E/locale.m		Zum Compilieren benötigt / E Programmierer
!		
libs/rexxmathlib.library		Für die trigonometrischen Funktionen

* Copyrights:

Final Writer ist (c) by Soft Wood Inc.

Der Installer und AmigaGuide sind (c) by Commodore. Beide sind auf der Fish
Disk
870 zu finden.

Workbench und Amiga sind eingetragene Warenzeichen von Commodore Amiga Inc.

Arexx ist (c) by William S. Hawes und Commodore

GoldEd ist (c) by Dietmar Eilert

Die AmigaGuide Icons sind (c) by Martin Huttenloher

Falls ich unwissentlich irgendwelche Copyrights verletzt haben sollte, bitte
ich
um Benachrichtigung!

1.16 Adresse des Autors

Für Verbesserungsvorschläge, Fehlermeldungen, Geschenke :-} etc. schreiben Sie
an:

Andreas Weiss
Dorfstrasse 24
CH-8212 Nohl
(Schweiz)

P.S: Dies ist mein allererstes Arexx-Programm. Gefällt es Ihnen?

Viel Spass!
ND

1.17 Danksagungen

Dank...

...an Dietmar Eilert für seinen grossartigen GoldEd, SoftWood für
ihren
FinalWriter (wann kommt Release 2?) und an den Programmierer
der
rexxmathlib.library (dessen Namen ich leider nicht kenne, da mir
die
Dokumentation fehlt)!

Danken möchte ich auch William S. Hawes für seine Portierung von ARexx!!!

Und nicht zu vergessen: Matrin Huttenloher für seine wunderbaren MagicWB
Icons
und Wouter van Oortmerssen für seinen E-Compiler!