RexxM2Error

COLLABORATORS

	TITLE : RexxM2Error	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 22, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

RexxM2Error

1.1 RexxM2Error Version 0.97

"RexxM2Error" ist ein nützliches Programm im Zusammenhang mit dem Modula-2 Compiler M2Amiga: Es stellt einen ARexx-Port zur Verfügung und liefert die Fehlermeldungen zu einem erfolglos kompilierten Programm. Dadurch können Fehlermeldungen in jedem beliebigen Editor (der ebenfalls eine ARexx-Schnittstelle hat) angezeigt werden.

Ich benutze z.B. den GoldED von Dietmar Eilert.

1.2 Installation und Aurfuf

Installation:

Die Installation ist sehr einfach: Kopiere das Programm "RexxM2Error" in ein beliebiges Verzeichnis, am besten eins, das auch im Suchpfad liegt (z.B. C:).

Zu diesem Paket gehören auch ein paar ARexx-Makros für den GoldED-Editor. Sie sollten am besten in das

ARexx-Verzeichnis von GoldED kopiert werden.
Eine Beschreibung der Makros ist im Abschnitt
RexxM2Error und GoldED zu finden.

Aufruf:

"RexxM2Error" tut nichts anderes, als auf eingehende ARexx-Kommandos zu warten. Das Programm sollte daher am besten mit RUN von der Shell, oder per Doppelklick auf das Icon auf der Workbench gestartet werden.

Die Umgebung des Programms kann mit den Tooltypes beeinflußt werden.

Beim Start von der Shell aus, stehen folgende Optionen zur Verfügung:

```
ERRORMSGFILE/K,DEBUG/S
```

ERRORMSGFILE bestimmt die Datei, in der die Fehlertexte stehen (siehe auch Beschreibung zum gleichnamigen Tooltype).
Die Option DEBUG sollte nicht verwendet werden. Sie ist von mir zu Testzwecken eingefügt worden.

Beenden:

"RexxM2Error" kann mit CTRL-C oder dem ARexx-Kommando QUIT beendet werden.
Das ARexx-Skript QuitRexxM2Error.rexx erledigt dies z.B.

1.3 Die Tooltypes von RexxM2Error

"RexxM2Error" kennt zur Zeit nur ein einziges Tooltype:

ERRORMSGFILE

Für die möglichen Optionen beim Aufruf aus der Shell siehe Installation und Aufruf.

1.4 Das Tool-Type ERRORMSGFILE

Format : ERROMSGFILE=<Datei mit Fehlertexten>

Funktion : Datei mit den Fehlertexten bestimmen.

Beschreibung:

Mit diesem Tool-Type kann die Datei angegeben werden, in der sich die Fehlertexte befinden. In der deutschen

Version von M2Amiga ist dies z.B. die Datei M2:Fehler-Meldungen.
Dies ist zugleich auch der voreingestellte Wert, wenn ERRORMSGFILE nicht angegeben ist.

1.5 Die ARexx-Kommandos von RexxM2Error

"RexxM2Error" wird über den ARexx-Port "REXXM2ERROR" angesprochen und versteht derzeit folgende Befehle:

```
ERROR  liefert eine Fehlermeldung.
ERRORS liefert die Anzahl der Fehler.
LOAD   lädt eine Fehlerdatei.
QUERY  zur Abfrage interner Variablen.
QUIT   beendet "RexxM2Error".
RESET  setzt den Fehlerzähler neu.
```

Wird ein unbekanntes Kommando oder ein falsches Kommando an "RexxM2Error" geschickt, öffnet sich ein Requester mit einer Fehlermeldung.

1.6 AREXX: ERROR

```
Format           : ERROR [ NEXT | PREV | FIRST |
                   BYTE=Offset | NUMBER=Nummer]

Befehlsschablone : NEXT/S,PREV/S,FIRST/S,BYTE/K/N,
                   NUMBER/K/N

Funktion         : Fehlermeldung und Position liefern.

Beschreibung     :
```

In Abhängigkeit der verwendeten Option liefert ERROR den nächsten (NEXT), vorhergehenden (PREV) oder ersten Fehler der geladenen Fehlerdatei (siehe LOAD).
Mir der Option BYTE kann ein Offset (Position relativ zum Dateianfang in Bytes) angegeben werden. ERROR liefert dann den im Quelltext hierauf folgenden Fehler.
Ein Fehler kann auch direkt mit der Option NUMBER angesprungen werden. Der Wert muß zwischen 0 und MAXERRORS-1 liegen (wird von ERRORS) geliefert. Liegt der Wert außerhalb dieses Bereichs, wird eine Warnung (RC = 5) zurückgegeben.
Eine Warnung bedeutet, daß kein weiterer Fehler gefunden wurde.
Ist RC = 0, so wurde noch ein weiterer Fehler gefunden.
Das Ergebnis steht in RESULT und hat folgendes Format:

```
   OFFSET/A,ERRNUM/A,ERRMSG/A
```

OFFSET ist die Position des Fehlers in Bytes, relativ zum

Dateianfang des Quelltextes. ERRNUM die vom Compiler erzeugte Fehlernummer und ERRMSG der zugehörige Fehlertext.

Rückgabewerte von 10 und mehr weisen auf schwere Probleme bei der Analyse der Fehlercodierung hin.

1.7 AREXX: ERRORS

Format : ERRORS

Befehlsschablone: -

Funktion : Anzahl der Fehler ermitteln.

Beschreibung:

ERRORS gibt in RESULT die Anzahl der Fehler zurück, oder eine Warnung, wenn keine Fehlerdatei geladen wurde.

Weitere interne Variablen können mit QUERY abgefragt werden.

1.8 AREXX: LOAD

Format : LOAD Datei [ERRORMSG]

Befehlsschablone: FILE/A,ERRORMSG/S

Funktion : Fehlerdatei oder Datei mit Fehlertexten laden.

Beschreibung:

Die gebräuchlichste Form in der Verwendung dieses Befehls liegt vor, wenn die Option ERRORMSG nicht angegeben ist. Dann ist Datei der Dateiname eines Moduls. LOAD hängt selber eine "E" an, um die zum Modul zugehörige Fehlerdatei zu laden. Wurde sie nicht gefunden, wird eine Warnung gemeldet.
Rückgabewerte größer 5 machen auf schwere Fehler aufmerksam (z.B. Lesefehler oder fehlendem Speicherplatz).

Ist die Option ERRORMSG angegeben, so bezeichnet Datei die Datei mit den Fehlertexten (z.B. M2:Fehler-Meldungen). Zum Start des Programms wird diese Datei auch durch das Tooltype ERRORMSGFILE angegeben.

1.9 AREXX: QUERY

```
Format          : QUERY FILE | CURRERR | MAXERRORS |
                  ERRORMSGFILE

Befehlsschablone: FILE/S,CURRERR/S,MAXERRORS/S,
                  ERRORMSGFILE/S

Funktion        : Abfrage interner Variablen.

Beschreibung:
```

Mit QUERY können diverse interne Variablen von "RexxM2Error" abgefragt werden.

FILE liefert den Namen der aktuellen Fehlerdatei oder eine Warnung, wenn derzeit keine Fehlerdatei geladen ist.

Mit CURRERR kann der Stand des Fehlerzählers abgefragt werden. Dieser wird durch ERROR beeinflußt.

MAXERRORS liefert wie ERRORS die Anzahl der Fehler insgesamt.

Und ERRORMSGFILE ergibt den Namen der aktiven Fehlertextedatei oder eine Warnung, wenn keine geladen wurde.

1.10 AREXX: QUIT

```
Format          : QUIT

Befehlsschablone: -

Funktion        : RexxM2Error beenden.

Beschreibung:
```

Beendet "RexxM2Error" und gibt die belegten Ressourcen wieder frei. Danach gibt es keinen ARexx-Port mehr, an den irgendwelche Befehle gesendet werden könnten.

1.11 AREXX: RESET

```
Format          : RESET [ NUMBER=Nummer ]

Befehlsschablone: NUMBER/K/N

Funktion        : Internen Fehlerzähler zurücksetzen.

Beschreibung:
```

Ohne Option setzt RESET den internen Fehlerzähler auf den ersten Fehler zurück.

Mit NUMBER kann dieser auf den Fehler Nummer gesetzt werden. Nummer muß zwischen 0 und MAXERRORS-1 liegen (siehe ERRORS).

1.12 Mitgelieferte ARexx-Skripte

In diesem Paket befinden sich einige ARexx-Skripte. Dies sind:

QuitRexxM2Error.rexx beendet "RexxM2Error"
GoldED-Makros

Die GoldED-Makros werden im Abschnitt RexxM2Error und GoldED erklärt.

1.13 ARexx-Skript: QuitRexxM2Error

Dieses kurze Skript beendet "RexxM2Error". Es kann entweder von der Shell aus mit

```
rx QuitRexxM2Error.rexx
```

oder per Doppelklick auf sein Piktogramm gestartet werden. Ein laufendes "RexxM2Error" Programm wird dann aus dem Speicher entfernt.

1.14 RexxM2Error und GoldED

Zu diesem Paket wurden folgende Skripte für den Editor GoldED von Dietmar Eilert mitgeliefert:

M2Error.ged
ResetM2Error.ged
QuitRexxM2Error.ged

Das erste Skript zeigt alle Fehlermeldungen im Quelltext an, das zweite lädt die zugehörige Fehlerdatei und das dritte beendet "RexxM2Error".

1.15 GoldED-Skript: M2Error.ged

Wenn sich im aktuellen Fenster ein Modula-2 Quelltext befindet (zu erkennen an der Dateiendung auf ".mod" oder ".def"), wird der nächste Fehler im Quelltext angezeigt und die Fehlermeldung in einem Requester angezeigt.
Gibt es keinen weiteren Fehler, so wird der Benutzer darüber informiert. Danach positioniert das Makro den

internen Fehlerzähler wieder auf den ersten Fehler. Bei einem erneuten Aufruf würde also wieder der erste Fehler im Quelltext angezeigt.

Ich habe dieses Makro auf F2 gelegt (wie von M2emacs gewohnt).

Werden Änderungen im Quelltext vorgenommen, so verschiebt sich logischerweise der Byteoffset für die Fehler. Das Makro kann hierauf leider nicht so reagieren wie M2emacs.

Fehlermeldungen:

"Dies ist kein Modula-2 Quelltext!"
"This is no Modula-2 source"

Im aktiven Fenster befindet sich vermutlich kein Modul-2 Code, da der Dateiname weder auf ".mod" noch auf ".def" endet.

"RexxM2Error läuft nicht! Bitte starten."
"RexxM2Error is not running! Please start that program."

Dieses Makro kann nur arbeiten, wenn das Programm "RexxM2Error" im Hintergrundläuft. Es sollte vorher gestartet werden (siehe Installation und Aufruf).

"Fehlerdatei paßt nicht!"
"Errorfile does not match!"

Der Quelltext im aktiven Fenster hat eine andere Fehlerdatei als die, die gerade in "RexxM2Error" aktiv ist.
Abhilfe: Mit dem Makro ResetM2Error.ged die zugehörige Fehlerdatei laden.

1.16 GoldED-Skript: ResetM2Error.ged

Befindet sich im aktuellen Fenster ein Modula-2 Quelltext, wird die zugehörige Fehlerdatei geladen --- falls vorhanden --- und der interne Fehlerzähler auf den ersten Fehler gesetzt.

Dieses Makro liegt bei mir auf SHIFT-F2.

Fehlermeldungen:

"Dies ist kein Modula-2 Quelltext!"
"This is no Modula-2 source"

Im aktiven Fenster befindet sich vermutlich kein Modul-2 Code, da der Dateiname weder auf ".mod" noch

auf ".def" endet.

"RexxM2Error läuft nicht! Bitte starten."
"RexxM2Error is not running! Please start that program."

Dieses Makro kann nur arbeiten, wenn das Programm "RexxM2Error" im Hintergrundläuft. Es sollte vorher gestartet werden (siehe Installation und Aufruf).

"Fehlerliste nicht gefunden!"
"Errorlist not found!"

Die zugehörige Fehlerdatei (gleicher Dateiname plus Endung "E") wurde nicht gefunden.

1.17 GoldED-Skript: QuitRexxM2Error.ged

Entfernt das Programm "RexxM2Error" aus dem Speicher.

Ich habe dieses Makro auf ALT-F2 gelegt.

1.18 Aufbau einer Fehlerdatei

Trifft der Compiler bei der Übersetzung auf Fehler, so erzeugt er eine Fehlerdatei (Dateiname mit dem Zusatz "E"). In dieser Datei stehen die Fehlerinformationen in codierter Form. In BNF (Backus-Naur-Form) sieht das so aus:

```
Fehlerdatei = ERRFILE {Error} End.
Error       = ErrorPos {ErrorPart}.
ErrorPos    = ERR Position.
ErrorPart   = ErrorNum | String.
ErrorNum    = INTEGER.
String      = STR {CHAR}.
Position    = LONGINT.
ERRFILE     = LONGINT(3).
ERR         = "ÁERR".
STR         = CHAR(0C2H).
End         = INTEGER(-1).
```

Position gibt die Position des Fehlers im Quelltext relativ zum Dateianfang (gemessen in Bytes) an.
String muß mit 0C abgeschlossen sein. Ggf. ist noch ein weiteres 0C einzufügen, damit der folgende Teil auf einer geraden Adresse beginnt.
ErrorNum ist eine Codierung des Fehlers. Der zugehörige Fehlertext ist in der Datei M2:Fehler-Meldungen zu finden.

siehe auch:
Handbuch zu M2Amiga, Abschnitt 7.1 (Seite 7-13ff)

1.19 Aufbau der Datei mit den Fehlertexten

In der Datei "M2:Fehler-Meldungen" stehen die Fehlertexte zu den codierten Fehlern (codiert durch eine positive INTEGER-Zahl, siehe Fehlerdatei). Diese Datei hat folgendes Format:

```
MessageFile = {Message} End.
Message     = Next Number String.
Next        = LONGINT.
Number      = INTEGER.
String      = {CHAR}.
End         = LONGINT(0) LONGINT(0).
```

Next gibt die nächste Message in der Datei an. Number ist die Fehlercodierung und String der eigentliche Fehlertext, der mit einem 0C abgeschlossen ist. Damit die nächste Message auf einer geraden Adresse beginnt, ist ggf. noch ein zweites 0C vorhanden.

siehe auch:
Handbuch zu M2Amiga, Abschnitt 7.1 (Seite 7-14f)

1.20 Copyright und Adresse

Für Fehlerhinweise, Anregungen, Wünsche und Kuscheltiere, CDs oder Postkarten bin ich jederzeit dankbar:

Fin Schuppenhauer
Braußpark 10
20537 Hamburg
(Germany)

Oder per E-Mail:

schuppen@rzdspc2.informatik.uni-hamburg.de

1.21 geschichte

Die Geschichte von "RexxM2Error" begann mit dem Skript M2Error.ged für den Editor GoldED:

So fasziniert ich von jenem "Urknall"-Skript bin (es war einer meiner ersten Versuche in Sachen ARexx-Programmierung), so schnell war ich auch von dessen Langsamkeit genervt. Außerdem mußte ich schon wenige Stunden, nachdem ich es ins Aminet downloadete, feststellen, daß es fehlerhaft war.

So entschloß ich mich, dieses Programm ("RexxM2Error") zu schreiben, hatte aber nicht die geringste Ahnung, wie man Programme mit einem ARexx-Port versieht.

Ein Artikel in der April-Ausgabe der c't und das Studium diverser Quelltexte lehrten mich dann. Ich hoffe, daß es halbwegs richtig und konform geworden ist. Für Hinweise, Tips und Anregungen bin ich dankbar.

17.3.94 : Erste Veröffentlichung von M2Error.ged
(Version 0.9) im Aminet.
20.3.94 : Neues Programm "RexxM2Error" geschrieben und
die Skripte für den GoldED angepaßt. Jetzt geit
das richtig schnell! Version 0.95.
21.3.94 : Kommandooptionen und Tooltypes hinzugefügt;
ausgiebig getestet und gefundene Fehler behoben
-> 0.96

1.22 danksagung

Mein besonderer Dank geht an Dietmar Eilert für seinen wirklich hervorragenden Editor GoldED.

Eine weiteres Dankeschön an Uwe Röhm für seinen Artikel in der April-Ausgabe der c't.