

**TM\_Deutsch**

**COLLABORATORS**

	<i>TITLE :</i> TM_Deutsch		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		August 22, 2022	

**REVISION HISTORY**

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>TM_Deutsch</b>	<b>1</b>
1.1	TM_Deutsch.guide	1
1.2	TM_Deutsch.guide/Copyright	2
1.3	TM_Deutsch.guide/Wichtig	3
1.4	TM_Deutsch.guide/Adresse des Autors	4
1.5	TM_Deutsch.guide/Schnellinstallation	5
1.6	TM_Deutsch.guide/Einfuehrung	5
1.7	TM_Deutsch.guide/Konzepte	6
1.8	TM_Deutsch.guide/Beispiele	7
1.9	TM_Deutsch.guide/Distributionsdateien	8
1.10	TM_Deutsch.guide/Docs	9
1.11	TM_Deutsch.guide/Goodies	10
1.12	TM_Deutsch.guide/Graphics	10
1.13	TM_Deutsch.guide/L	11
1.14	TM_Deutsch.guide/Libs	12
1.15	TM_Deutsch.guide/Locale	12
1.16	TM_Deutsch.guide/Prefs	12
1.17	TM_Deutsch.guide/Programmers	13
1.18	TM_Deutsch.guide/Scripts	13
1.19	TM_Deutsch.guide/Source	13
1.20	TM_Deutsch.guide/WBStartup	13
1.21	TM_Deutsch.guide/Objekte	14
1.22	TM_Deutsch.guide/Programm	14
1.23	TM_Deutsch.guide/Bild	16
1.24	TM_Deutsch.guide/Ton	17
1.25	TM_Deutsch.guide/Menue	17
1.26	TM_Deutsch.guide/Piktogramm	18
1.27	TM_Deutsch.guide/Dock	18
1.28	TM_Deutsch.guide/Zugriff	20
1.29	TM_Deutsch.guide/Voreinstellungen	21

---

---

1.30	TM_Deutsch.guide/Gadgets im Hauptfenster . . . . .	22
1.31	TM_Deutsch.guide/Menues des Hauptfensters . . . . .	22
1.32	TM_Deutsch.guide/Erzeuge Objekte Fenster . . . . .	23
1.33	TM_Deutsch.guide/Objektfenster . . . . .	23
1.34	TM_Deutsch.guide/Tooltypes . . . . .	24
1.35	TM_Deutsch.guide/CLI Argumente . . . . .	25
1.36	TM_Deutsch.guide/Library . . . . .	25
1.37	TM_Deutsch.guide/Tastenbefehle . . . . .	26
1.38	TM_Deutsch.guide/InputEvent-Klassen . . . . .	28
1.39	TM_Deutsch.guide/Qualifier . . . . .	28
1.40	TM_Deutsch.guide/Tastenkodes . . . . .	29
1.41	TM_Deutsch.guide/rawkey Tastenkodes . . . . .	30
1.42	TM_Deutsch.guide/rawmouse Tastenkodes . . . . .	30
1.43	TM_Deutsch.guide/Beispiele für Tastenbefehle . . . . .	31
1.44	TM_Deutsch.guide/Fragen . . . . .	31
1.45	TM_Deutsch.guide/Geschichte . . . . .	33
1.46	TM_Deutsch.guide/Danksagungen . . . . .	34
1.47	TM_Deutsch.guide/Index . . . . .	35

---

# Chapter 1

## TM\_Deutsch

### 1.1 TM\_Deutsch.guide

ToolManager 2.1 Dokumentation

\*\*\*\*\*

Kapitel für alle Benutzer:

Copyright

Copyright und andere rechtliche Dinge

Wichtig

Wichtige Bemerkungen

Adresse des Autors

Wohin man Bug reports, Kommentare & Spenden schickt

Kapitel für die ungeduldigen Benutzer:

Schnellinstallation

Wie man ToolManager 2.1 schnell installiert

Kapitel für neue Benutzer:

Einführung

Was ist ToolManager?

Konzepte

Die Konzepte hinter ToolManager

Beispiele

Ein paar Beispiele

Distributionsdateien

Beschreibungen für alle Dateien in der Distribution

Kapitel zum Nachschlagen:

---

Objekte	Beschreibung der ToolManager-Objekte
Voreinstellungen	Der Voreinsteller für ToolManager
Library	Die ToolManager shared library Schnittstelle
Tastengebefehle	Wie man einen Tastengebefehl definiert

#### Anhänge:

Fragen	Die häufigsten Fragen zum ToolManager
Geschichte	Die Entwicklung von ToolManager
Danksagungen	Der Autor möchte danken...
Index	Index für dieses Dokument

## 1.2 TM\_Deutsch.guide/Copyright

Copyright und andere rechtliche Dinge

\*\*\*\*\*

Copyright (C) 1990-93 Stefan Becker

Diese Dokumentation darf kopiert und weitergegeben werden solange die Copyright-Notiz und diese Erlaubnis unverändert auf allen Kopien enthalten ist.

Es wird keine Garantie gegeben, daß die Programme, die in dieser Dokumentation beschrieben werden, 100%ig zuverlässig sind. Sie benutzen diese Programme auf eigene Gefahr. Der Autor kann auf keinen Fall für irgendwelche Schäden verantwortlich gemacht werden, die durch die Anwendung dieser Programme entstehen.

Das Paket ist "freely distributable", aber das Copyright liegt weiterhin bei Stefan Becker. Dies bedeutet, daß es von jedem kopiert werden darf solange er nicht mehr als eine angemessene Kopiergebühr dafür verlangt. Diese Gebühr darf nicht höher sein als US \$5 oder 5 DM.

Dieses Limit gilt auch für deutsche Public-Domain Händler!!

Dieses Paket darf in Public-Domain Sammlungen aufgenommen werden, insbesondere in Fred Fishs Amiga Disk Library (CD ROM Versionen dieser Sammlung eingeschlossen). Die Distributionsdatei darf in Mailboxsystemen oder auf FTP Servern abgelegt werden. Wenn Sie dieses Paket weitergeben wollen, dann müssen Sie die originale Distributionsdatei ToolManager2\_0bin.lha, ToolManager2\_0gfx.lha und ToolManager2\_0src.lha benutzen.

Weder die Programme noch der Quelltext (oder Teile davon) dürfen ohne eine schriftliche Genehmigung des Autors in kommerziellen Programmen benutzt werden.

Die Programme und der Quelltext (oder Teile davon) dürfen auf keinen Fall auf irgendeiner Maschine benutzt werden, die für die Forschung, Entwicklung, Konstruktion, Tests oder Produktion von Waffen oder anderen militärischen Gütern benutzt wird. Dies gilt natürlich auch für alle Maschinen, die für das Training von Personen in irgendeiner der obengenannten Tätigkeiten benutzt werden.

### 1.3 TM\_Deutsch.guide/Wichtig

#### Wichtige Bemerkungen

\*\*\*\*\*

Willkommen zur wundervollen Welt von ToolManager 2.1 :-)

- ToolManager und seine Konzepte wurden drastisch verändert (siehe

Geschichte

) seit der Version 1.5.

- Beginnend mit der Version 2.0 ist ToolManager GiftWare. Wenn Ihnen dieses Programm gefällt und Sie es sehr oft benutzen, dann sollten Sie darüber nachdenken, ob sie dem Autor eine kleine Spende schicken um die Arbeit zu honorieren, die er in dieses Programm gesteckt hat. Ich schlage eine Spende in der Größenordnung von US \$10-\$20 oder 10-20 DM vor. Bitte schicken Sie mir außerhalb Europas keine Schecks, da es meistens mehr Geld kostet, diesen Scheck einzulösen, als überhaupt Geld darauf ist.

Wenn Sie keine Spende schicken oder es sich nicht erlauben können, dann müssen sie sich nicht schuldig fühlen. Sie sollten mir aber trotzdem einen Brief schicken, wenn sie ToolManager benutzen (Ich liebe es Post zu bekommen :-). Siehe

Adresse des Autors

.

- Benutzer von ToolManager 1.X/2.0 können mit dem Kapitel über die Schnellinstallation beginnen (siehe Schnellinstallation). Einige Merkmale haben sich nicht geändert und der Rest ist relativ

einfach durch Ausprobieren herauszufinden. Für eine genaue Beschreibung des neuen Konzeptes und der neuen Merkmale schlagen sie bitte in dem Referenzteil dieser Dokumentation nach (siehe

Objekte  
).

Jede noch laufende ToolManager 1.X/2.0 Version muß entfernt werden oder die neue Version wird nicht funktionieren. Die neue Version kann leider nicht die Konfigurationsdateien der alten 1.X Versionen lesen.

- Erstbenutzer sollten die ganze Dokumentation lesen, um das Konzept und den Zweck des Programmes zu verstehen. Beginnen sie mit

Einführung

- ToolManager 2.1 benutzt einige Merkmale der AmigaOS Version V38 (und höher) und er unterstützt die neuen Netzwerkmöglichkeiten des AmigaOS, die (hoffentlich) bald für alle Benutzer zugänglich sein werden. Wenn sie noch die Version 2.0 benutzen (sie wird in dieser Dokumentation als V37 bezeichnet), dann brauchen sie keine Angst zu haben, denn ToolManager arbeitet auch ohne diese neue Merkmale. Alle erweiterten Merkmale sind in dieser Dokumentation markiert.

## 1.4 TM\_Deutsch.guide/Adresse des Autors

Wohin man Bug reports, Kommentare & Spenden schickt

\*\*\*\*\*

Der Autor kann unter folgenden Adressen erreicht werden:

Postadresse:

Stefan Becker  
Holsteinstrasse 9  
5100 Aachen  
GERMANY

Bitte benutzen Sie nach dem 1. Juli 1993 die folgende Adresse:

Stefan Becker  
Holsteinstrasse 9  
52068 Aachen  
GERMANY

InterNet Electronic Mail:

stefanb@pool.informatik.rwth-aachen.de

## 1.5 TM\_Deutsch.guide/Schnellinstallation

Wie man ToolManager 2.1 schnell installiert

\*\*\*\*\*

Die grundlegende ToolManager 2.1 Installation besteht aus den folgenden vier Teilen:

Libs/toolmanager.library => LIBS:

Dies ist das Hauptprogramm von ToolManager. Es verwaltet alle Programme, Menüs, Piktogramme und Docks (siehe Library ).

Prefs/ToolManager\* => SYS:Prefs

Dies ist der Voreinsteller für die Konfiguration (siehe Voreinstellungen ).

WBStartup/ToolManager\* => SYS:WBStartup

Mit diesem Hilfsprogramm kann ToolManager gestartet und gestoppt werden. Wenn es in der WBStartup Schublade liegt, dann wird ToolManager immer automatisch beim Starten der Maschine geladen.

L/WBStart-Handler => L:

Dieses Programm startet Programme mit der Workbench-Startmethode. Es ist ein eigener Prozess, so daß man ToolManager auch dann noch verlassen kann, wenn man Workbenchprogramme gestartet hat.

Nachdem Sie diese Dateien kopiert haben, sollten Sie eine noch laufende alte Version vom ToolManager stoppen und die neue Version durch Doppel-Klick des ToolManager Piktogramms im Verzeichnis WBStartup starten. Nun können Sie den Voreinsteller starten und mit den neuen Möglichkeiten herumspielen (Benutzen Sie das Gadget "Testen" statt "Benutzen"). Sie dürften die meisten Merkmale mit Trial & Error herausfinden. Für weitere Informationen schauen sie bitte in die Beschreibungen für die ToolManager-Objekten (siehe Objekte ).

Die Distribution enthält eine Beispielkonfiguration in der Datei TM\_Demo.prefs. Sie können diese in den Voreinsteller mit dem Menüpunkt Open laden.

## 1.6 TM\_Deutsch.guide/Einfuehrung

Was ist ToolManager?

\*\*\*\*\*

ToolManager ist ein flexibles Programm zur Verwaltung von Hilfsprogrammen in Ihrer Arbeitsumgebung. Es kann sowohl Workbench-

oder CLI-Programme starten, ARexx-Skripte ausführen und Tastenbefehle erzeugen. Des weiteren kann es Befehle an einen ToolManager auf einer anderen Maschine schicken. Die Benutzeroberfläche besteht aus Menüs, Piktogrammen und Dock-Fenstern. Wenn Sie einen lauten Computer mögen, dann können Sie zu jedem dieser Dinge einen Ton hinzufügen. Siehe

Tonobjekte

.

ToolManager kann Einträge zu dem Workbench Hilfsmittel Menü hinzufügen. Wenn Sie einen dieser Einträge auswählen, dann wird das dazugehörige Programm gestartet. Jedes Piktogramm, das zu diesem Zeitpunkt auf der Workbench ausgewählt ist, wird als Argument für dieses Programm benutzt. Menüeinträge sind nur dann möglich, wenn die Workbench läuft. Siehe

Menüobjekte

.

ToolManager kann Piktogramme zum Workbench-Fenster hinzufügen. Falls Sie ein solches Piktogramm doppelklicken, dann wird das dazugehörige Programm gestartet. Wenn Sie einige Piktogramme auf dieses Piktogramm schieben, dann wird das Programm mit diesen Piktogrammen als Argumenten gestartet. Piktogramme sind nur dann möglich, wenn die Workbench läuft. Siehe

Piktogrammobjekte

.

ToolManager kann ein Dock-Fenster aus einer Zusammenfassung von Programmen erstellen. Dieses Fenster kann auf jedem öffentlichen Schirm geöffnet werden. Jedes Programm wird durch ein Bild oder ein Textgadget repräsentiert. Um ein Programm zu starten brauchen Sie nur auf das Bild oder das Textgadget zu klicken. Falls das Fenster auf der Workbench geöffnet wurde und die Workbench läuft, dann können Sie auch einige Piktogramme auf das Bild oder das Textgadget schieben, um das Programm mit Argumenten zu starten. Siehe

Dock-Objekte

.

Weiterhin können Sie jedem Programm einen Tastenbefehl zuweisen. Wenn Sie diesen Tastenbefehl benutzen, dann wird das Programm gestartet. Beachten Sie dabei, daß in diesem Fall keine Argumente an das Programm übergeben werden können. Siehe

Programmobjekte

.

## 1.7 TM\_Deutsch.guide/Konzepte

Die Konzepte hinter ToolManager

\*\*\*\*\*

ToolManager 2.1 benutzt einen neuen objektorientierten Ansatz um ein flexibles und erweiterbares System zu ermöglichen. Dieser Ansatz machte viele Verbesserungen an den ToolManager 1.X Merkmalen möglich,

z.B. man kann jetzt mehrere Dock-Fenster haben.

Ein Objekt ist eine Zusammenfassung von Daten, die seine Merkmale beschreiben. Jedes Objekt hat einen Namen und einen Typ. Sie können von jedem Typ so viele Objekte erzeugen wie sie möchten, allerdings muß der Name jedes Objektes eindeutig sein, da er dazu benutzt wird um das Objekt anzusprechen.

Momentan gibt es sieben verschiedene Objekttypen: Programm, Bild, Ton, Menü, Piktogramm, Dock und Zugriff. Die ersten drei Typen sind einfache Objekte, d.h. sie verweisen auf keine anderen Objekte. Sie stellen den komplexen Objekten Daten oder Dienste zur Verfügung.

Die letzten vier Typen sind komplexe Objekte, d.h. sie verweisen auf einfache Objekte und benötigen deren Daten oder Dienste. Der Verweis geschieht aufgrund des Namens und falls kein einfaches Objekt mit diesem Namen existiert, dann ignoriert das komplexe Objekt es. Beachten Sie dabei, daß dies die Funktionalität des komplexen Objektes beeinträchtigen kann, z.B. benötigt ein Piktogrammobjekt die Daten eines Bildobjekts, anderenfalls wird es kein Piktogramm erzeugen.

Für eine detaillierte Beschreibung aller Objektparameter siehe

Objekte

.

## 1.8 TM\_Deutsch.guide/Beispiele

Ein paar Beispiele

\*\*\*\*\*

Haben Sie bis jetzt kein Wort verstanden? Sind Sie verwirrt durch Objekte, Programme und Referenzen? Verzweifeln Sie nicht, denn Hilfe ist unterwegs.

Ich werde Ihnen nun Schritt für Schritt ein paar Beispiele zeigen, wie Sie ToolManager konfigurieren. Sie müssen dazu nur ToolManager installieren und den Voreinsteller starten. Nach jedem Schritt benutzen Sie bitte das Gadget "Testen" im Hauptfenster, um die Konfiguration zu testen.

Wir benutzen das Textanzeigeprogramm More aus dem Verzeichnis SYS:Utilities für unser Beispiel. Als erstes müssen wir ToolManager sagen, welches Programm wir benutzen möchten. Informationen über Programme werden in Programmobjekten abgespeichert. Also wählen Sie jetzt "Programm" als Objekttyp im Hauptfenster des Voreinstellers und drücken das Gadget "Neu".

Nachdem Sie das Gadget gedrückt haben, sehen Sie das "Ändere Programmobjekt" Fenster. Nun öffnen Sie das Verzeichnis Utilities auf Ihrer Workbench-Partition und schieben das Piktogramm "More" auf das neue Fenster des Voreinstellers. Wie Sie jetzt sehen können, sind der Name des Objekts und der Befehl auf "More" gesetzt, während das

aktuelle Verzeichnis auf "SYS:Utilities" gesetzt wurde. Drücken Sie nun das "Benutzen" Gadget um diese Einstellungen zu übernehmen.

Da man mit einem Programmobjekt allein nicht viel anfangen kann, wollen wir nun dieses Programm zu dem Hilfsmittel Menü der Workbench hinzufügen. Dazu wählen Sie "Menü" als Objekttyp und drücken das Gadget "Neu". Nun sehen Sie das "Ändere Menüobjekt" Fenster. Setzen Sie den Namen des Objekts auf "Text anzeigen".

ToolManager muß wissen, welches Programm er starten soll, wenn der Menüeintrag ausgewählt wird. Daher erstellen wir eine Referenz von dem Menüobjekt zu einem Programmobjekt. Drücken Sie das Gadget "Programmobjekt" und wählen sie von der Liste das Objekt "More" aus. Nun drücken Sie das Gadget "Benutzen" und dann "Testen" im Hauptfenster. Jetzt können Sie eine Textdatei auf der Workbench auswählen und den neuen Menüeintrag anwählen. Das Programm "More" sollte starten und Ihren Text anzeigen. Das war doch einfach, oder?

Nun können wir einen Schritt weitergehen und ein Piktogramm auf der Workbench erzeugen. Für ein Piktogramm benötigen wir Grafikdaten, die in Bildobjekten abgelegt sind. Wählen Sie "Bild" als Objekttyp und drücken Sie das Gadget "Neu". Das "Ändere Bildobjekt" Fenster öffnet sich. Setzen Sie den Namen auf "Bild für More" und legen Sie das Piktogramm von More aus dem Verzeichnis Utilities auf das Fenster. Drücken Sie das Gadget "Benutzen" zur Übernahme der Einstellungen.

Im nächsten Schritt erstellen wir das Piktogramm. Wählen Sie "Piktogramm" als Objekttyp und drücken Sie das Gadget "Neu". Setzen Sie den Namen des Objekts auf "Text anzeigen". Drücken Sie das Gadget "Programmobjekt" und wählen Sie das Objekt "More" von der Liste aus. Drücken Sie das Gadget "Bildobjekt" und wählen Sie das Objekt "Bild für More" von der Liste aus. Setzen Sie die X Position auf 100 und die Y Position auf 50. Nun drücken Sie die Gadgets "Benutzen" und "Testen". Nach einer kurzen Pause sollte ein Piktogramm auf Ihrer Workbench erscheinen, auf das Sie die Piktogramme Ihrer Textdateien legen können, um diese anzuzeigen.

Nun sollten Sie in etwa verstanden haben, wie man ToolManager-Objekte benutzt und in welcher Weise man sie kombinieren muß, damit eine Konfiguration entsteht. Jetzt sollten Sie den Rest der Möglichkeiten durch Ausprobieren herausfinden. Sie können sich auch die Beispielkonfiguration in der Datei TM\_Demo.prefs anschauen.

## 1.9 TM\_Deutsch.guide/Distributionsdateien

Beschreibungen für alle Dateien in der Distribution

\*\*\*\*\*

Die ToolManager 2.1 Distribution besteht aus mehreren Verzeichnissen, die alle weiter unten erläutert werden. Beachten Sie bitte, daß die Distribution in drei Teile aufgespalten ist, d.h. es können einige Verzeichnisse nicht vorhanden sein.

Docs	Dokumentation
Goodies	Zusätzliche Programmpakete
Graphics	Animationen, Bilder & Piktogramme
L	Unterstützende Programme
Libs	ToolManager Library
Locale	Dateien zur Lokalisation
Prefs	ToolManager Voreinsteller
Programmers	Hilfsdateien für Programmierer
Scripts	ARexx- & Shell-Skripte
Source	Kompletter Quelltext zu ToolManager 2.1
WBStartup	Hilfsprogramme

## 1.10 TM\_Deutsch.guide/Docs

Das Docs Verzeichnis

=====

Dieses Verzeichnis enthält die Dokumentation für ToolManager. Die Dokumentation ist in vier verschiedenen Formaten und verschiedenen Sprachen vorhanden. Weiterhin existiert eine Datei in dem AutoDoc-Format, die die ToolManager shared library Schnittstelle beschreibt.

Prefix TM\_<Sprache>

Diese Datei enthält die Dokumentation in der angegebenen Sprache. Momentan sind folgende Sprachen vorhanden: Deutsch, Français, English, Svenska.

Postfix .doc

Diese Datei enthält die Dokumentation als einfacher ASCII-Text.

**Postfix .dvi**

Diese Datei enthält die Dokumentation in TeXs DVI-Format. Um daraus eine gedruckte Dokumentation zu erstellen, müssen sie diese Datei durch einen TeX Druckertreiber schicken.

**Postfix .guide**

Diese Datei enthält die Dokumentation im AmigaGuide-Format. Obwohl dieses Format nur aus einfachem ASCII-Text mit ein paar Befehlen besteht, benötigen Sie AmigaGuide um die Hypertext Verkettungen nutzen zu können.

**Postfix .tex**

Diese Datei enthält die Dokumentation im Texinfo-Format. Dieses Format wurde von der Free Software Foundation (FSF) erfunden. Zusammen mit dem texinfo.tex Makropaket können sie TeX und texindex benutzen, um eine Datei im DVI-Format zu erstellen (siehe oben).

**toolmanager.doc**

Diese Datei beschreibt die ToolManager shared library Schnittstelle im AutoDoc-Format.

## 1.11 TM\_Deutsch.guide/Goodies

### Das Goodies Verzeichnis

=====

Dieses Verzeichnis enthält zusätzliche Programmpakete, die nützlich für ToolManager sind.

**GetPubName.lha**

Dieses kleine Programm gibt den Namen des obersten öffentlichen Schirms auf die Standardausgabe oder in eine Umgebungsvariable aus. Es wurde geschrieben von Michael "Mick" Hohmann.

**updl\_20.lha**

Das Programm upd wurde geschrieben von Jonas Petersson. Es ist ein kleines Program, daß einen ARexx-Port öffnet und auf Befehle wartet. Mit Hilfe von ARexx können Sie upd veranlassen digitalisierte Töne und Geräusche abzuspielen. ToolManager benutzt diese Eigenschaft um seine Tonobjekte zu realisieren. Siehe

Tonobjekte

.

## 1.12 TM\_Deutsch.guide/Graphics

### Das Graphics Verzeichnis

=====

---

Dieses Verzeichnis enthält eine reichhaltige Sammlung von Bildern aus denen Sie wählen können. Laden Sie sie einfach als Bildobjekte in ToolManager (siehe Bildobjekte).

Die Dateien wurden von verschiedenen Personen zur Verfügung gestellt (siehe

Danksagungen). Jeder von ihnen hat ein eigenes Unterverzeichnis erhalten. Da diese Dateien von verschiedenen Personen erstellt wurden, stammen sie aus verschiedenen Umgebungen (Farbpalette, Tiefe, Auflösung, Größe) und haben unterschiedliche Desginstile. Es kann also durchaus vorkommen, daß sie nicht gut auf Ihrem Rechner aussehen.

Damit man die unterschiedlichen Formate, die von ToolManager unterstützt werden, unterscheiden kann, hat jeder Dateiname einen speziellen Postfix:

.anmb

Dies ist eine IFF ANIM Datei, die von einem Mal- oder Animationsprogramm erstellt wurde. Sie kann mehrere Bilder enthalten. Obwohl ToolManager komplette ANIM Dateien laden kann, sollten sie eine ähnliche Funktion wie DPaints "AnimBrush" benutzen, um den interessanten Teil aus der Animation herauszuschneiden.

.brush

Dies ist eine IFF ILBM Datei, die von einem Malprogramm erzeugt wurde. Sie enthält nur ein Bild.

.info

Dies ist eine normale Amiga Piktogrammdatei, die durch IconEdit (oder etwas ähnliches) erzeugt wurde. Sie kann zwei Bilder enthalten.

## 1.13 TM\_Deutsch.guide/L

Das L Verzeichnis

=====

Dieses Verzeichnis enthält nur eine Datei, nämlich WBStart-Handler. Sie müssen diese Datei in das L: Verzeichnis kopieren, sonst kann ToolManager keine Programmobjekte mit der Workbench-Startmethode starten (siehe Programmobjekte).

Das komplette WBStart 1.2 Paket befindet sich auf Fish Disk 757.

## 1.14 TM\_Deutsch.guide/Libs

Das Libs Verzeichnis

=====

Dieses Verzeichnis enthält nur eine Datei, `toolmanager.library`. Dies ist das Hauptprogramm von ToolManager und muß in das LIBS: Verzeichnis kopiert werden.

## 1.15 TM\_Deutsch.guide/Locale

Das Locale Verzeichnis

=====

Dieses Verzeichnis enthält alle Dateien für die Lokalisation von ToolManager. Da die `locale.library` erst in V38 enthalten ist, brauchen Sie keine dieser Dateien zu kopieren, falls sie noch V37 benutzen. Wenn Sie schon V38 benutzen, dann wählen Sie die Ihrer Sprache entsprechende Datei.

Catalogs/<Sprache>/toolmanager.catalog

Dies ist eine Übersetzungsdatei für die angegebene Sprache. Kopieren Sie die Datei für Ihre Sprache in das Verzeichnis LOCALE:Catalogs/<Sprache>.

Languages/<Sprache>.language

Einige Sprachen werden noch nicht unterstützt von der Standard V38 Locale Distribution. Daher haben einige Übersetzer eine `.language` Datei mitgeliefert, damit ToolManager ihre Übersetzungsdateien benutzen kann. Kopieren Sie die Datei für Ihre Sprache in das Verzeichnis LOCALE:Languages. Es sind folgende zusätzlichen Sprachen verfügbar: Finnish (suomi), Eefeler Platt (eifel).

## 1.16 TM\_Deutsch.guide/Prefs

Das Prefs Verzeichnis

=====

Der ToolManager Voreinsteller und sein Icon befinden sich in diesem Verzeichnis. Kopieren Sie beide Dateien in das Verzeichnis SYS:Prefs. Für weitere Informationen siehe

Voreinstellungen

.

## 1.17 TM\_Deutsch.guide/Programmers

Das Programmers Verzeichnis

=====

Dieses Verzeichnis enthält alle Dateien, die von den verschiedenen Programmiersprachen und deren Compilern benötigt werden, um die ToolManager shared library Schnittstelle zu benutzen. Das Unterverzeichnis examples enthält mehrere Beispiele dazu. Für eine komplette Beschreibung der Schnittstelle lesen Sie bitte die Datei Docs/toolmanager.doc.

Momentan werden die folgenden Sprachen bzw. Compiler unterstützt: AmigaOberon, DICE C, M2Amiga Modula-2, MANX Aztec C and SAS C.

## 1.18 TM\_Deutsch.guide/Scripts

Das Scripts Verzeichnis

=====

Dieses Verzeichnis enthält eine Sammlung von ARexx- und Shell-Skripten, die in ToolManager Programmobjekten benutzt werden können. Beachten Sie, daß diese Skriptdateien an Ihre persönliche Arbeitsumgebung oder Shell angepasst werden müssen.

## 1.19 TM\_Deutsch.guide/Source

Das Source Verzeichnis

=====

Dieses Verzeichnis enthält den kompletten Quelltext zu ToolManager 2.1 und seinen Hilfsprogrammen. Jedes Programm hat sein eigenes Unterverzeichnis. Der Autor stellt den Quelltext als Beispiel für die Programmierung unter OS 2.x/3.0 zur Verfügung.

Das Unterverzeichnis locale ist für Übersetzer interessant. Falls Ihre Sprache in dieser Version nicht unterstützt wird und Sie eine Übersetzung machen möchten, dann sollten Sie sich die Datei empty.ct anschauen. Sie müssen nur die Leerzeilen füllen und die Datei zu mir senden. Die Übersetzung wird dann eventuell in der nächsten Version enthalten sein.

## 1.20 TM\_Deutsch.guide/WBStartup

Das WBStartup Verzeichnis

=====

---

Dieses Verzeichnis enthält nur ein Programm: ToolManager. Dieses Hilfsprogramm startet und stoppt ToolManager 2.1. Meistens wird dieses Programm in das Verzeichnis SYS:WBStartup kopiert, aber man kann es auch von der Shell aus benutzen.

## 1.21 TM\_Deutsch.guide/Objekte

Beschreibung der ToolManager-Objekte

\*\*\*\*\*

Dieses Kapitel beschreibt die ToolManager-Objekte im einzelnen. Jedes Objekt hat einen Typ und einen Namen. Der Name wird benutzt, um das Objekt anzusprechen. Es gibt sechs verschiedene Typen von Objekten:

Programm	Programmobjekte
Bild	Bildobjekte
Ton	Tonobjekte
Menü	Menüobjekte
Piktogramm	Piktogrammobjekte
Dock	Dock-Objekte
Zugriff	Zugriffsobjekte

## 1.22 TM\_Deutsch.guide/Programm

Programmobjekte

=====

Programmobjekte beschreiben Programme oder Aktionen, die von ToolManager gestartet werden sollen. Es werden drei verschiedene Typen von Programmen unterstützt: CLI, Workbench und ARexx. Drei verschiedene Typen von Aktionen werden unterstützt: Dock, Hot Key und Network. Jedes Programmobjekt hat die folgenden Parameter. Die Vorgabewerte sind in Klammern gesetzt:



#### Tastenbefehl

Sie können für jedes Programmobjekt einen Tastenbefehl angeben. Wenn Sie diesen Tastenbefehl benutzen wird das Programm gestartet. Beachten Sie: Das Programm wird ohne Argumente gestartet.

#### Verzögerung (0)

Nach der Aktivierung eines Programmobjekts wartet ToolManager Verzögerung Sekunden bevor er das Programm startet. Wenn dieser Wert negativ ist, dann wird das Programm alle Verzögerung Sekunden gestartet. Falls Sie ein Programmobjekt stoppen wollen, das auf die Ausführung wartet, aktivieren Sie es einfach noch einmal. Beachten Sie: Wenn Verzögerung gesetzt ist, dann wird das Programm ohne Argumente gestartet.

## 1.23 TM\_Deutsch.guide/Bild

#### Bildobjekte

=====

Bildobjekte geben die Bilddaten an, die von ToolManager für Piktogramme und Dock-Fenster benutzt werden. Dieser Objekttyp hat nur einen Parameter:

#### Dateiname

Dieser Parameter gibt den Namen der Datei an, aus der ToolManager die Bilddaten lesen soll. ToolManager versucht den Typ der Bilddaten automatisch zu erkennen:

1. Er versucht die Daten als IFF Daten zu laden. Momentan kann ToolManager ILBM (ein Bild) oder ANIM (zwei oder mehr Bilder) Dateien lesen.
2. Er versucht die Datei als Piktogrammdatei zu lesen. Ein Piktogramm kann ein oder zwei Bilder enthalten.

Animation werden momentan nur von Dock-Objekten unterstützt. Piktogrammobjekte entnehmen momentan nur das erste und das zweite Bild einer Animation um ein Piktogramm aus 2 Bildern zu erstellen. Wenn Sie eine Animation für ToolManager erstellen wollen, dann sollten Sie sich an folgende Regeln halten:

#### Bild 1

Dieses Bild sollte den inaktiven Zustand repräsentieren.

#### Bild 2

Dieses Bild sollte den selektierten Zustand repräsentieren. Normalerweise ist es eine invertierte Kopie des ersten Bildes.

#### Bild 3 bis N-1

Dies sind die Bilder der eigentlichen Animation. Jedes Bild wird für eine 1/3 Sekunde angezeigt.

Bild N

Das letzte Bild der Animation wird für eine Sekunde angezeigt.  
Danach wird wieder das erste Bild angezeigt.

## 1.24 TM\_Deutsch.guide/Ton

Tonobjekte

=====

Ein Tonobjekt wird benutzt, wenn ToolManager laut werden soll. ToolManager hat selber keinen eingebauten Spieler für digitalisierte Geräusche. Er benutzt ARexx, um einen externen Geräuscheabspieler anzusteuern. Dieser Objekttyp hat zwei Parameter:

Befehl

Dieser Parameter gibt den ARexx-Befehl an, den ToolManager an den externen Geräuscheabspieler senden soll. Für upd könnte dies etwas ähnliches wie `file samples:boing` sein, was upd anweist die IFF Datei `samples:boing` abzuspielen. Siehe  
Goodies  
.

ARexx-Port

Dieser Parameter gibt den Namen des ARexx-Ports an, an den ToolManager Befehl senden soll. Der voreingestellte Name ist `PLAY`, der Name für das Programm `upd`.

## 1.25 TM\_Deutsch.guide/Menue

Menüobjekte

=====

Menüobjekte erzeugen Menüeinträge im Workbench Hilfsmittel Menü. Der Objektname wird benutzt als Menütext. Um ein solches Objekt zu aktivieren müssen Sie einfach den entsprechenden Menüeintrag selektieren. Menüobjekte sind nur verfügbar wenn die Workbench läuft.

Dieser Objekttyp hat zwei Parameter:

Programmobjekt

Dies ist der Name eines Programmobjektes, das aktiviert werden soll, wenn der Menüeintrag selektiert wurde. Jedes Piktogramm, das zu diesem Zeitpunkt ausgewählt ist, wird als Argument für das Programm benutzt.

Tonobjekt

Dies ist der Name eines Tonobjekts, das aktiviert werden soll, wenn der Menüeintrag selektiert wurde.

---

ToolManager 1.X Benutzer: Wenn Sie den alten Tooltyp "Dummy" simulieren wollen, dann erzeugen Sie einfach ein Menüobjekt und geben kein Programm- oder Tonobjekt an.

## 1.26 TM\_Deutsch.guide/Piktogramm

Piktogrammobjekte

=====

Piktogrammobjekte beschreiben Applikationspiktogramme im Workbench-Fenster. Solch ein Objekt kann aktiviert werden durch Doppelklicken des Piktogramms oder indem einige Piktogramme auf das Applikationspiktogramm geschoben werden. Piktogrammobjekte sind nur verfügbar wenn die Workbench läuft.

Die Parameter für diesen Objekttyp sind wie folgt:

Bildobjekt

Dies ist der Name eines Bildobjekts. Die Bilddaten dieses Objektes werden dazu benutzt das Applikationspiktogramm zu erstellen.

Linke Kante (Vorgabe: 0)

Dies setzt die linke Kante für das Applikationspiktogramm.

Obere Kante (Vorgabe: 0)

Dies setzt die obere Kante für das Applikationspiktogramm.

Programmobjekt

Dies ist der Name eines Programmobjektes, das aktiviert werden soll wenn das Piktogramm selektiert wurde. Jedes Piktogramm, das auf das Applikationspiktogramm geschoben wurde, wird als Argument für das Programm benutzt.

Tonobjekt

Dies ist der Name eines Tonobjekts, das aktiviert werden soll wenn das Piktogramm selektiert wurde.

Zeige Namen (Vorgabe: Ja)

Wenn dieser Parameter gesetzt ist, dann wird der Objektname als Namen für das Applikationspiktogramm benutzt.

Die Workbench ist sehr eigensinnig mit Piktogrammpositionen. Wenn Sie eine Position gewählt haben, die der Workbench nicht zusagen, dann ignoriert sie sie einfach und legt das Piktogramm irgendwo anders hin.

## 1.27 TM\_Deutsch.guide/Dock

Dock-Objekte

=====

---

Dock-Objekte beschreiben Fenster. Diese Fenster fassen mehrere Programme zusammen, welche durch Bilder oder Textgadgets repräsentiert werden. Um ein Programm zu starten, braucht man nur auf das Bild oder das Textgadget zu klicken. Natürlich können auch Piktogramme auf das Bild oder das Textgadget geschoben werden, um das Programm mit Argumenten zu versorgen.

Jedes Dock-Objekt hat die folgenden Parameter. Die Vorgaben sind in Klammern gesetzt:

Aktiviert (Ja)

Ein Dock-Fenster kann aktiv (offen) oder nicht aktiv (geschlossen) sein.

Backdrop (Nein)

Wenn dieser Parameter gesetzt ist, dann setzt sich das Dock-Fenster nach dem Öffnen sofort in den Hintergrund.

Fensternamen

Dieser Parameter setzt den Namen des Dock-Fensters. Wenn Sie einen Fensternamen angeben, dann öffnet sich ein normales OS 2.0 Fenster mit Dragbar, Close-Gadget, Depth-Gadget und einem Rand. Wenn Sie keinen Namen angeben, dann öffnet sich ein Fenster, das keinen Rahmen und nur einen Dragbar hat.

Linke Kante (0)

Linke Kante des Dock-Fensters. Dieser Parameter wird ignoriert, wenn der Parameter Zentriert gesetzt ist.

Menü (Nein)

An jedes Dock-Fenster kann ein Menü angehängt werden. Dieses Menü hat zwei Einträge:

- \* Dock schliessen Schliessen des Dock-Fensters.
- \* TM beenden ToolManager beenden.

Muster (Nein)

Das Dock-Fenster richtet sich automatisch nach der Grösse des größten Bildes. Da jeder Eintrag die gleiche Grösse hat und kleinere Bilder zentriert werden, haben kleinere Bilder einen leeren Rand. Falls Sie diesen Rand nicht mögen, dann können Sie mit diesem Parameter ein Muster einschalten.

Obere Kante (0)

Obere Kante des Dock-Fensters. Dieser Parameter wird ignoriert, wenn der Parameter Zentriert gesetzt ist.

Öffentlicher Schirm (Default)

Dies gibt den öffentlichen Schirm an, auf dem das Dock-Fenster geöffnet werden soll. Wenn das Fenster mit Hilfe des Tastenbefehls geöffnet wurde, dann wird der öffentliche Schirm nach vorne geholt. Dieser Parameter wird ignoriert, wenn der Parameter Vorderster gesetzt ist.

PopUp (Nein)

Wenn dieser Parameter gesetzt ist, dann schliesst sich das

Dock-Fenster automatisch, nachdem ein Eintrag ausgewählt wurde. Dieser Parameter ist besonders nützlich in Verbindung mit den Parametern Vorderster, Zentriert und einem Tastenbefehl der Klasse rawmouse (siehe Tastenbefehle).

#### Spalte (1)

Dieser Parameter setzt die Anzahl der Spalten in dem Dock-Fenster. Die Einträge werden zeilenweise einsortiert, beginnend mit der linken Spalte und fortlaufend bis zur rechten Spalte.

#### Stationär (Nein)

Normalerweise merkt sich ein Dock-Fenster beim Schliessen seine letzte Position und öffnet sich an der gleichen Stelle wieder. Wenn das Dock-Fenster immer an der gleichen Position aufgehen soll, dann setzen Sie diesen Parameter.

#### Tastenbefehl

Sie können für jedes Dock-Objekt einen Tastenbefehl angeben. Wenn Sie diesen Tastenbefehl benutzen, dann wird der Status des Dock-Fensters umgeschaltet, d.h. es wird geöffnet oder geschlossen.

#### Text (Nein)

Mit diesem Parameter können Sie zwischen Bildern und Textgadgets in Dock-Fenstern wählen. Dock-Fenster mit Textgadgets sind besonders nützlich in Verbindung mit dem Parameter PopUp.

#### Vertikal (Nein)

Dieser Parameter setzt die Richtung der Dragbar. Dieser Parameter wird ignoriert, wenn Sie einen Fensternamen mit dem Parameter Fensternamen angegeben haben.

#### Vorderster (Nein)

Wenn dieser Parameter gesetzt ist, dann öffnet sich das Dock-Fenster immer auf dem vordersten öffentlichen Schirm.

#### Zeichensatz (Zeichensatz des Schirms)

Wenn in einem Dock-Objekt der Parameter Text gesetzt ist, dann kann mit diesem Parameter der Zeichensatz für die Textgadgets gewählt werden.

#### Zentriert (Nein)

Wenn dieser Parameter gesetzt ist, dann wird das Dock-Fenster immer zentriert zur aktuellen Mausposition geöffnet.

## 1.28 TM\_Deutsch.guide/Zugriff

### Zugriffsobjekte

=====

Zugriffsobjekte kontrollieren die Zugriffsrechte für Anfragen über

---

Netzwerke. Grundsätzlich wird jede Anfrage abgelehnt, so daß ein anderer ToolManager nicht die Arbeit Ihrer Maschine durch Aktivierung von Programmobjekten stören kann. Mit Zugriffsobjekten jedoch können sie es einzelnen Maschinen erlauben, einige Programmobjekte von Ihrem ToolManager auszuführen.

Der Name eines Zugriffsobjekt hat dabei eine spezielle Bedeutung. Er wird mit dem Namen der Maschine verglichen, von dem die Anfrage stammt. Dabei benutzt ToolManager das folgende dreistufige Vergleichsschema:

1. Vergleiche mit dem kompletten Maschinennamen
2. Vergleiche mit dem Bereichsnamen
3. Suche nach einem Zugriffsobjekt mit dem Namen anyone

Wenn das entsprechende Objekt gefunden wurde, dann gibt dieses Objekt die Zugriffsrechte der anderen Maschine an. Das Objekt mit dem Namen anyone wird für jede Netzwerkanfrage benutzt, für die kein Zugriffsobjekt gefunden werden kann.

Zugriffsobjekte haben nur einen Parameter:

Programmobjekt

Dieser Parameter kann mehrfach benutzt werden und gibt die Programmobjekte an, die von der anderen Maschine aktiviert werden können. Wenn Sie keinen Objektnamen angeben, dann kann die andere Maschine alle Programmobjekte auf Ihrer Maschine aktivieren.

## 1.29 TM\_Deutsch.guide/Voreinstellungen

Der Voreinsteller für ToolManager

\*\*\*\*\*

Mit dem Voreinsteller können Sie die globale Konfiguration von ToolManager manipulieren. Diese Konfiguration wird automatisch beim Start von ToolManager geladen. Um den Voreinsteller zu starten, müssen Sie sein Piktogramm doppelklicken. Danach öffnet sich das Hauptfenster.

Die meisten Gadgets in den Voreinstellerfenstern haben einen Tastenkürzel. Sie sind mit einem Unterstrich () markiert. Beachten Sie allerdings, daß Sie bei einem aktiven Zeichenketten-Gadget erst die Return-Taste betätigen müssen, bevor Sie das Tastenkürzel benutzen können.

Gadgets im Hauptfenster

Menüs des Hauptfensters

Erzeuge Objekte Fenster

Objektfenster

Tooltypes

CLI Argumente

## 1.30 TM\_Deutsch.guide/Gadgets im Hauptfenster

Gadgets im Hauptfenster

=====

Das Hauptfenster hat mehrere Gruppen von Gadgets:

Objekttyp

Mit diesem zyklischem Auswahlgadget kann der Typ der Objekte gewählt werden, die Sie erzeugen oder ändern wollen.

Objektliste

Dieses Gadget zeigt die Liste aller Objekte des aktuellen Typs an. Sie können ein Objekt auswählen indem sie auf seinen Namen klicken. Wenn sie einen Eintrag doppelklicken, dann öffnet sich das Objektfenster.

Objekt verschieben

Wenn ein Objekt ausgewählt worden ist, dann kann es mit diesen Gadgets in der Liste verschoben werden. Das Sortieren Gadget dient dazu, die Liste alphabetisch zu sortieren.

Objekt manipulieren

Diese Gadgets manipulieren Objekte. Das Neu Gadget erzeugt ein neues Gadget vom aktuellen Typ. Dieses neue Objekt wird automatisch selektiert. Das Ändern Gadget öffnet das Objektfenster. Mit dem Kopieren Gadget kann man eine Kopie des selektierten Objektes erstellen. Das Entfernen Gadget löscht das selektierte Objekt.

Konfiguration

Sie haben mehrere Möglichkeiten die Konfiguration zu speichern. Mit dem Speichern Gadget wird die Konfiguration permanent in der Datei ENVARC:ToolManager.prefs gespeichert. Für eine temporäre Änderung kann die Konfiguration mit dem Benutzen Gadget in der Datei ENV:ToolManager.prefs abgespeichert werden. Diese Datei wird bei einem Reset gelöscht. Wenn Sie eine Konfiguration testen wollen, ohne den Voreinsteller zu verlassen, dann benutzen Sie das Testen Gadget. Mit dem Abbrechen Gadget können Sie den Voreinsteller ohne Abspeichern verlassen.

## 1.31 TM\_Deutsch.guide/Menues des Hauptfensters

Menüs des Hauptfensters

=====

---

Das Hauptfenster hat mehrere Menüs:

#### Project

Mit den Menüeinträgen öffnen und speichern als können Sie die Konfiguration laden und speichern. Der Information Eintrag öffnet einen Informationsrequester. Mit dem Eintrag beenden kann man den Voreinsteller ohne Abspeichern verlassen.

#### Vorgaben

Mit den zwei Menüeinträgen können ältere Konfigurationen wiederhergestellt werden. Der Eintrag auf zuletzt gespeichertes lädt die Konfiguration aus der Datei ENVARC:ToolManager.prefs. Mit dem Eintrag auf vorherigen Stand wird diejenige Konfiguration, die vor dem Start des Voreinstellers gültig war, aus der Datei ENV:ToolManager.prefs geladen.

#### Optionen

Mit dem Eintrag Piktogramme erzeugen? können Sie vorgeben, ob der Eintrag speichern als ein Piktogramm erzeugen soll oder nicht.

## 1.32 TM\_Deutsch.guide/Erzeuge Objekte Fenster

### Erzeuge Objekte Fenster

=====

Wenn Sie ein Piktogramm auf das Hauptfenster legen, dann erscheint das "Erzeuge Objekte" Fenster. In diesem Fenster können sie auswählen, wie das Piktogramm verarbeitet werden soll. Dies können Sie benutzen, um ein Program einfach und schnell zu Ihre Konfiguration hinzuzufügen.

Sie können entweder nur ein Programm- oder Bildobjekt aus dem Piktogramm erzeugen, in dem Sie einen der ersten beiden Punkte auswählen. Sie können aber auch ein komplettes Menü- und/oder Piktogrammobjekt erzeugen, in dem Sie einen der drei letzten Punkte auswählen.

## 1.33 TM\_Deutsch.guide/Objektfenster

### Objektfenster

=====

Jeder Objekttyp hat ein eigenes Objektfenster, in dem man die Objektparameter setzen kann. Für eine detaillierte Liste aller Parameter siehe

Objekte

.

Jedes Objektfenster hat ein Gadget für den Objektnamen. Dieser Name

---

ist wichtig, da er dazu benutzt wird das Objekt anzusprechen. Beachten Sie, daß es keine Querverweise gibt, d.h. wenn Sie den Namen eines Objektes ändern, das schon von einem anderen Objekt verwendet wird, so wird der Verweis in diesem Objekt nicht aktualisiert. Sie müssen diesen Verweis per Hand aktualisieren.

Die Tastengadgets in den Objektfenstern öffnen verschiedene Requester. Sie können einen Eintrag auswählen, indem sie den Namen und das Benutzen Gadget anklicken oder indem sie den Namen doppelklicken. Um einen Requester ohne Änderungen zu verlassen, benutzen Sie das Abbrechen Gadget. Wenn sie ein Feld löschen wollen, das nur mit einem Requester ausgewählt werden kann, dann öffnen Sie den Requester und klicken das Benutzen Gadget ohne vorher einen Eintrag auszuwählen.

Die Objektfenster für Programm- und Bildobjekte haben ein zusätzliches Merkmal. Sie können ein Piktogramm auf diese Fenster schieben und die Parameter werden aus diesem Piktogramm entnommen.

## 1.34 TM\_Deutsch.guide/Tooltypes

Tooltypes  
=====

Wenn Sie den Voreinsteller von der Workbench starten, dann können Sie in den Piktogrammen für das Programm oder den Konfigurationsdateien verschiedene Tooltypes setzen, um den Voreinsteller zu kontrollieren.

USE

Wenn Sie diesen Tooltype in einem Piktogramm für eine Konfigurationsdatei setzen, dann installiert der Voreinsteller diese Datei als aktuelle Konfiguration.

SAVE

Wenn Sie diesen Tooltype in einem Piktogramm für eine Konfigurationsdatei setzen, dann installiert der Voreinsteller diese Datei als aktuelle und permanente Konfiguration.

PUBSCREEN

Dieser Tooltype gibt den Namen des öffentlichen Schirms an, auf dem der Voreinsteller seine Fenster öffnen soll.

CREATEICONS

Wenn dieser Tooltype auf YES gesetzt wird, dann erzeugt der Voreinsteller für jede Konfigurationsdatei, die mit dem Menüeintrag speichern als erzeugt wird, ein Piktogramm.

DEFAULTFONT

Normalerweise benutzt der Voreinsteller den Zeichensatz des öffentlichen Schirms um seine Gadgets zu erzeugen. Wenn Sie diesen Tooltype auf YES setzen, dann benutzt der Voreinsteller den Standard Systemzeichensatz.

---

XPOS

Dies gibt die X Position des Hauptfensters an.

YPOS

Dies gibt die Y Position des Hauptfensters an.

MINLISTCOLUMNS

Dies gibt die minimale Anzahl der Spalten in den Listen-Gadgets an.

MINLISTROWS

Dies gibt die minimale Anzahl der Reihen in den Listen-Gadgets an.

### 1.35 TM\_Deutsch.guide/CLI Argumente

CLI Argumente

=====

Wenn der Voreinsteller von der Shell gestartet wird, dann benutzt er folgende Schablone für die Kommandozeile:

```
FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K,DEFAULTFONT/S
```

FROM

Dieser Parameter gibt an, welche Konfigurationsdatei der Voreinsteller laden soll.

USE

Wenn Sie diesen Parameter benutzen, dann installiert der Voreinsteller die Datei, die mit dem FROM Parameter angegeben wurde, als aktuelle Konfiguration.

SAVE

Wenn Sie diesen Parameter benutzen, dann installiert der Voreinsteller die Datei, die mit dem FROM Parameter angegeben wurde, als aktuelle und permante Konfiguration.

PUBSCREEN

Dieser Parameter gibt den Namen des öffentlichen Schirms an, auf dem der Voreinsteller seine Fenster öffnen soll.

DEFAULTFONT

Normalerweise benutzt der Voreinsteller den Zeichensatz des öffentlichen Schirms um seine Gadgets zu erzeugen. Wenn Sie diesen Parameter angeben, dann benutzt der Voreinsteller den Standard Systemzeichensatz.

### 1.36 TM\_Deutsch.guide/Library

---

## Die ToolManager shared library Schnittstelle

\*\*\*\*\*

Der ToolManager-Handler ist in eine Amiga shared library eingebettet. Diese Library bietet mehrere Funktionen an, um ToolManager-Objekte zu erzeugen und zu manipulieren, so daß Sie sie von Ihren Programmen aus benutzen können.

Momentan sind 6 Funktionen vorhanden:

**AllocTMHandle()**

Damit Sie ToolManager-Objekte erzeugen können, müssen Sie zuerst einen TMHandle allozieren. Diese Datenstruktur speichert alle Informationen über Ihre Objekte und sie wird dazu benutzt Ihre Objekte anzusprechen. Beachten Sie, daß die Information, die in dieser Datenstruktur gespeichert ist, nur dem Programm zugänglich ist, das den TMHandle erzeugt hat.

**FreeTMHandle()**

Diese Funktion gibt einen TMHandle und alle mit ihm verbundenen ToolManager-Objekte frei. Jeder AllocTMHandle() Aufruf muß mit einem FreeTMHandle() Aufruf gepaart sein!

**CreateTMOBJECTTags()****CreateTMOBJECTTagList()**

Diese Funktion erzeugt ein ToolManager-Objekt. Sie müssen einen Namen, den Objekttyp und verschiedene Tags für die Objektparameter angeben. Der Name des Objekts ist wichtig, da er dazu benutzt wird, das Objekt anzusprechen.

**ChangeTMOBJECTTags()****ChangeTMOBJECTTagList()**

Sie können die Parameter eines ToolManager-Objekts mit dieser Funktion verändern. Beachten Sie, daß Bildobjekte momentan nicht geändert werden können.

**DeleteTMOBJECT()**

Diese Funktion löscht ein ToolManager-Objekt. Falls andere Objekte auf dieses Objekt verweisen, dann werden diese benachrichtigt, damit sie ihren Status erneuern können.

**QuitToolManager()**

Diese Funktion teilt dem ToolManager-Handler mit, daß er so bald wie möglich stoppen sollte.

Die komplette Library Schnittstellenbeschreibung ist im AutoDoc-Format vorhanden (siehe Dokumentation).

## 1.37 TM\_Deutsch.guide/Tastenbefehle

## Wie man einen Tastenbefehl definiert

\*\*\*\*\*

Diese Kapitel beschreibt wie man einen Tastenbefehl als einen Input Description String definiert, der dann von Commodities ausgewertet werden kann. Jedes Mal, wenn ein Tastenbefehl ausgeführt wird, erzeugt Commodities eine Ereignis, das dann von ToolManager dazu benutzt wird Programmobjekte zu aktivieren oder Dock-Objekte umzuschalten. Ein Description String hat die folgende Syntax:

```
[<Klasse>] {[-][<Qualifier>]} [-][upstroke] [<Tastencode>]
```

Alle Befehlswoorte können groß oder klein geschrieben werden.

Klasse beschreibt die InputEvent-Klasse. Dieser Parameter ist optional und falls er weggelassen wird, dann wird die Vorgabe rawkey benutzt. Siehe

InputEvent-Klassen

.

Qualifier sind "Signale", die gesetzt oder nicht gesetzt sein müssen zu dem Zeitpunkt, an dem der Tastenbefehl ausgeführt wird, sonst wird kein Ereignis erzeugt. Für jeden Qualifier, der gesetzt sein soll, müssen Sie das Befehlswort angeben. Alle anderen Qualifier müssen dann nicht gesetzt sein. Falls Sie einen Qualifier ignorieren wollen, dann setzen sie ein - vor sein Befehlswort. Siehe

Qualifier

.

Normalerweise wird ein Ereignis erzeugt, wenn eine Taste gedrückt wird. Falls das Ereignis generiert werden soll wenn die Taste losgelassen wird, dann müssen Sie das Befehlswort upstroke angeben. Wenn sowohl beim Drücken als auch beim Loslassen der Taste ein Ereignis erzeugt werden soll, dann müssen sie das Befehlswort -upstroke angeben.

Der Tastencode ist abhängig von der InputEvent-Klasse. Siehe

Tastenkodes

.

InputEvent-Klassen

Qualifier

Tastenkodes

Beispiele für Tastenbefehle

Achtung: Wählen Sie ihre Tastenbefehle sorgfältig, denn Commodities hat eine hohe Priorität in der InputEvent-Handlerkette, d.h. vorgegebene Definitionen werden übergangen.

## 1.38 TM\_Deutsch.guide/InputEvent-Klassen

### InputEvent-Klassen

=====

Commodities unterstützt die meisten der InputEvent-Klassen, die von dem `input.device` erzeugt werden. Diese Sektion beschreibt die Klassen, die nützlich für ToolManager sind.

#### rawkey

Dies ist die vorgegebene Klasse. Sie beschreibt alle Ereignisse, die durch die Tastatur erzeugt werden können. Zum Beispiel erzeugt `rawkey` `a` oder `a` jedesmal ein Ereignis, wenn die Taste "a" gedrückt wird. Sie müssen einen Tastenkodex für diese Klasse angeben. Siehe

`rawkey`

.

#### rawmouse

Diese Klasse beschreibt alle Ereignisse, die durch die Maus erzeugt werden können. Sie müssen einen Tastenkodex für diese Klasse angeben. Siehe

`rawmouse`

.

#### diskinserted

Ereignisse dieser Klasse werden generiert, wenn eine Diskette in ein Laufwerk gelegt wird. Diese Klasse besitzt keine Tastenkodes.

#### diskremoved

Ereignisse dieser Klasse werden generiert, wenn eine Diskette aus einem Laufwerk genommen wird. Diese Klasse besitzt keine Tastenkodes.

## 1.39 TM\_Deutsch.guide/Qualifier

### Qualifier

=====

Einige Befehlswozte wurden erst bei Commodities V38 eingeführt. Diese sind mit einem \* markiert.

`lshift`, `left_shift` \*

Linke Shift-Taste.

`rshift`, `right_shift` \*

Rechte Shift-Taste.

`shift`

Irgendeine Shift-Taste.

`capslock`, `caps_lock` \*

Caps-Lock-Taste.

caps

Irgendeine Shift-Taste oder die Caps-Lock-Taste.

control, ctrl \*

Control-Taste.

lalt, left\_alt \*

Linke Alt-Taste.

ralt, right\_alt \*

Rechte Alt-Taste.

alt

Irgendeine Alt-Taste.

lcommand, lamiga \*, left\_amiga \*, left\_command \*

Linke Amiga-/Kommando-Taste.

rcommand, ramiga \*, right\_amiga \*, right\_command \*

Rechte Amiga-/Kommando-Taste.

numericpad, numpad \*, num\_pad \*, numeric\_pad \*

Dieses Befehlswort muß angegeben werden, wenn eine Taste von der Zehnertastatur benutzt wird.

leftbutton, lbutton \*, left\_button \*

Linke Maustaste. Siehe unten.

midbutton, mbutton \*, middlebutton \*, middle\_button \*

Mittlere Maustaste. Siehe unten.

rbutton, rightbutton \*, right\_button \*

Rechte Maustaste. Siehe unten.

repeat

Dieser Qualifier ist gesetzt, wenn die Tastenwiederholung aktiv ist. Dies ist nur sinnvoll für die InputEvent-Klasse rawkey.

Achtung: Commodities V37 hat einen Fehler, der die Benutzung von leftbutton, midbutton und rbutton als Qualifier verhindert. Dieser Fehler wurde in V38 behoben.

## 1.40 TM\_Deutsch.guide/Tastenkodes

Tastenkodes

=====

Jede InputEvent-Klasse besitzt ihre eigenen Tastenkodes:

rawkey

---

rawmouse

## 1.41 TM\_Deutsch.guide/rawkey Tastenkodes

Tastenkodes für die InputEvent-Klasse 'rawkey'

-----

Einige Befehlswoorte wurden erst bei Commodities V38 eingeführt. Diese sind mit einem \* markiert.

a-z, 0-9, ...  
ASCII-Zeichen.

f1, f2, ..., f10, f11 \*, f12 \*  
Funktionstasten.

up, cursor\_up \*, down, cursor\_down \*  
left, cursor\_left \*, right, cursor\_right \*  
Cursor-Tasten.

esc, escape \*, backspace, del, help  
tab, comma, return, space, spacebar \*  
Spezial-Tasten.

enter, insert \*, delete \*  
page\_up \*, page\_down \*, home \*, end \*  
Tasten der Zehnertastatur. Diese Tastenkodes müssen mit dem  
Qualifier numericpad benutzt werden!

## 1.42 TM\_Deutsch.guide/rawmouse Tastenkodes

Tastenkodes für die InputEvent-Klasse 'rawmouse'

-----

Diese Befehlswoorte wurden erst bei Commodities V38 eingeführt. Sie sind nicht verfügbar in V37.

mouse\_leftpress  
Drücke die linke Maustaste.

mouse\_middlepress  
Drücke die mittlere Maustaste.

mouse\_rightpress  
Drücke die rechte Maustaste.

Achtung: Um einen dieser Tastenkodes zu benutzen, müssen sie auch das entsprechende Qualifier-Befehlswoort angeben, z.B.

---

```
rawmouse leftbutton mouse_leftpress
```

## 1.43 TM\_Deutsch.guide/Beispiele für Tastenbefehle

Beispiele für Tastenbefehle

=====

```
ralt t
```

Rechte Alt-Taste festhalten und "t" drücken.

```
ralt lalt t
```

Rechte und linke Alt-Taste festhalten und "t" drücken.

```
alt t
```

Irgendeine Alt-Taste festhalten und "t" drücken.

```
rcommand f2
```

Rechte Amiga-Taste festhalten und die zweite Funktionstaste drücken.

```
numericpad enter
```

Enter-Taste auf der Zehnertastatur drücken.

```
rawmouse midbutton leftbutton mouse_leftpress
```

Mittlere Maustaste festhalten und die linke Maustaste drücken.

```
diskinserted
```

Eine Diskette in ein Laufwerk einlegen.

## 1.44 TM\_Deutsch.guide/Fragen

Die häufigsten Fragen zum ToolManager

\*\*\*\*\*

Hier sind die Antworten für die am häufigsten gestellten Fragen zum ToolManager:

- Warum kann ToolManager keine mehrfachen Menüs oder Untermenüs erstellen?

Mehrfache Menüs oder Untermenüs werden zur Zeit noch nicht vom Betriebssystem unterstützt. Um sie zu erzeugen, muß man sie in das System patchen, was zu einem instabilen System führen kann. Da ich keine un stabile Software schreiben möchte, werde ich so etwas nicht in ToolManager einbauen.

- WB-Programme starten nicht, aber alles andere funktioniert.

ToolManager benötigt das Programm L:WBStart-Handler, um WB-Programme starten zu können. Es gibt zwei Möglichkeiten,

warum ToolManager diese Programm nicht ausführen kann:

- \* Die Datei L:WBStart-Handler existiert nicht. Bitte kopieren Sie diese Datei aus der Distribution nach L:.
- \* Die Execute-Flagge (e) ist bei dieser Datei nicht gesetzt. Benutzen Sie das folgende Kommando, um diese Flagge zu setzen: protect L:WBStart-Handler +e

- Wie kann ich ein horizontales Dock-Fenster erzeugen?

Setzen Sie die Anzahl der Spalten gleich der Anzahl der Einträge in dem Dock-Objekt.

- Wie kann ich ein Ausgabefenster für CLI-Programme erzeugen?

Ausgabefenster können mit Hilfe des CON: Geräts erzeugt werden. Benutzen Sie den folgenden Dateinamen, um ein sich automatisch öffnendes Fenster mit einem Close-Gadget zu erzeugen, das nach dem Ende des Programmes sich nicht automatisch schliesst:

```
CON:10/10/640/100/Ausgabefenster/AUTO/CLOSE/WAIT
```

Das Gerät CON: hat viele Optionen, die Sie dem AmigaDOS Handbuch entnehmen können.

- Wie kann ich die Argumente in die Mitte einer CLI/ARexx Kommandozeile einfügen?

Normalerweise werden alle Argumente an die Kommandozeile angehängt. Um sie jedoch irgendwo in der Kommandozeile einzufügen, benutzt ToolManager die gleiche [] Syntax, wie der AmigaShell-Befehl alias. Dies sieht dann z.B. so aus:

```
Dir [] all
```

Alle Argumente werden vor dem Schlüsselwort all eingefügt.

- Wie kann ich eine Referenz von einem komplexen Objekt zu einem einfachen Objekt löschen?

Nachdem Sie das Gadget "xxxobjekt" gedrückt haben, müssen Sie das Gadget "Benutzen" drücken ohne vorher ein Objekt auszuwählen. Dies bedeutet, daß Sie kein Objekt ausgewählt haben und daher wird die Referenz gelöscht.

- Wie kann ich Unter-Docks erzeugen?

Sie müssen dafür Programmobjekte des Typs Dock verwenden. Setzen Sie solche Objekte in Ihr Haupt-Dock und sie werden dann die anderen Dock-Fenster öffnen bzw. schliessen.

- ToolManager rührt sich nicht mehr, nachdem er eine Netzwerkanfrage erzeugt hat.

Momentan gibt es ein Problem in der Software für Netzwerke: Lokale Anfragen werden bei Zeitüberschreitung nicht abgebrochen.

Wenn Ihre Maschine nun Maschinel heißt und Sie ein Programmobjekt vom Typ Network mit dem Befehl Objekt@Maschinel haben, dann gerät ToolManager in eine Ausschluß-Situation, wenn Sie dieses Objekt aktivieren. Bitte benutzen Sie nur Namen von anderen Rechnern!

## 1.45 TM\_Deutsch.guide/Geschichte

Die Entwicklung von ToolManager

\*\*\*\*\*

2.1, Datum 16.05.1993

- Neue Programmobjekttypen: Dock, Hot Key, Network
- Neue Dock-Objekt Parameter: Backdrop, Stationär
- Neuer Objekttyp: Zugriff
- Netzwerkunterstützung
- Das Voreinsteller Hauptfenster ist jetzt ein AppWindow
- Tastenkürzel für die Gadgets im Voreinsteller
- Neue Tooltypes für den Voreinsteller
- Mehrere Fehler beseitigt
- Verbesserte Dokumentation

2.0, Datum 26.09.1992, Fish Disk #752

- Komplett neues Konzept (objektorientiert)
  - (Fast) komplette Neuprogrammierung
  - ToolManager ist nun in zwei Teile aufgespalten
  - Der Handler ist in eine shared Library eingebettet
  - Die Konfiguration wird nun mit einem Voreinsteller bearbeitet
  - Das Konfigurationsdateiformat wurde wieder geändert :-) Es ist nun eine IFF Datei und liegt in ENV:
  - Mehrere Docks und Docks mit mehrere Spalten sind nun möglich
  - Es gibt Docks mit neuem Aussehen
  - Docks richten sich nun automatisch nach der Größe des größten Bildes
  - Sound Unterstützung
  - ARexx werden direkt in Programmobjekten unterstützt
-

- ToolManager kann nun ohne Workbench benutzt werden. Wenn die Workbench nicht läuft, dann benutzt er keine App\* Merkmale mehr.
- Locale Unterstützung
- Der Befehlspfad der Workbench wird für CLI Programme benutzt
- Ein eigener Prozess startet die WB Programme

1.0 bis 1.5

Die Bemerkungen zu diesen Versionen entnehmen Sie bitte der englischen Dokumentation.

## 1.46 TM\_Deutsch.guide/Danksagungen

Der Autor möchte danken...

\*\*\*\*\*

ToolManager hat mehrere große Entwicklungsphasen seit seiner ersten Programmierung Mitte 1990 durchlebt. Diese Entwicklung wäre unmöglich gewesen, wenn ich nicht den enormen Feedback von einigen ToolManager Benutzern gehabt hätte. Viele Ideen & Merkmale stammen aus dieser Quelle...

Daher möchte ich den folgenden Personen danken:

Für die Alpha-/Beta-Tests, Ideen & Bug Reports:

Die Amigagruppe unseres Computerclubs (Computerclub an der RWTH Aachen), Olaf 'Olsen' Barthel, Georg Hessmann (Gucky), Markus Illenseer (ill), Klaus Melchior, Rickard Olsson (Richie), Matthias Scheler (Tron), Ralph Schmidt (laire), Roger Westerlund (Budda), Juergen Weinelt, Brian Wright (SteveVai), Petra Zeidler (stargazer) und viele andere...

Matthew Dillon

Ohne dein exzellentes C Entwicklungssystem DICE und verschiedener anderer Hilfsprogramme würde es keinen ToolManager geben!

Für ihre hervorragenden Grafiken:

Andreas Harrenberg, Georg Hessmann, Michael "Mick" Hohmann, Markus Illenseer, Oliver Koenen, Klaus Melchior, Rickard Olsson, Jan Peter, Matthias Scheler, Brian Wright

Für die Übersetzungen:

Tomi Blinnikka (Finnisch), Dr. Peter Kittel (Deutsch), Klaus Melchior (Eifel Platt), Rickard Olsson (Schwedisch), Marc Schaefer (Französisch & Italienisch)

Alle Leute in West Chester:

Für die Entwicklung des Amigas und seines hervorragenden Betriebssystems.

Alle Benutzer, die mir Geld geschickt haben:

Ich habe in den 1.X Versionen nicht danach gefragt, aber es ist erfreulich, wenn jemand meine Arbeit so sehr schätzt.

Alle Benutzer, die mir einen Brief geschickt haben:  
Ich habe es genossen, Eure Briefe zu lesen!

und natürlich alle, die ich vergessen habe zu erwähnen...

## 1.47 TM\_Deutsch.guide/Index

```

Index
*****

Übersetzer
                                Source

Übersetzungen
                                Locale

TeX
                                Docs

Adresse
                                Adresse des Autors

AmigaGuide
                                Docs

Antworten
                                Fragen

ARexx-Skripte
                                Scripts

ASCII Dokumentation
                                Docs

Beispielbilder
                                Graphics

Beispiele
                                Beispiele

Beispiele für Tastenbefehle
    Beispiele für Tastenbefehle

Bildobjekte
                                Bild

Bug reports
                                Adresse des Autors

Catalog-Dateien
                                Locale

```

---

Compilerunterstützung Programmers	
Copyright	Copyright
Danksagungen	Danksagungen
DeleteTool	Goodies
Diskinserted	InputEvent-Klassen
Diskremoved	InputEvent-Klassen
Distribution	Copyright
Distributionsdateien Distributionsdateien	
Dock-Objekte	Dock
Docs Verzeichnis	Docs
Dokumentation	Docs
E-Mail	Adresse des Autors
Einführung zu den ToolManager-Objekten Konzepte	
Einführung zum ToolManager Einfuehrung	
Fragen	Fragen
Gedruckte Dokumentation Docs	
Genehmigungen	Copyright
Geschichte	Geschichte
GetPubName	Goodies

---

GiftWare	Wichtig
Goodies Verzeichnis	Goodies
Graphics Verzeichnis	Graphics
InputEvent-Klassen	InputEvent-Klassen
Installation (schnell)	Schnellinstallation
InterNet Adresse	Adresse des Autors
Kommentare	Adresse des Autors
Konfiguration	Voreinstellungen
Konzepte	Konzepte
L Verzeichnis	L
Library Dokumentation	Docs
Library Schnittstelle	Library
Libs Verzeichnis	Libs
List: Qualifier	Qualifier
Liste: rawkey Tastenkodes	rawkey Tastenkodes
Liste: rawmouse Tastenkodes	rawmouse Tastenkodes
Locale Verzeichnis	Locale
Lokalisation	Locale
Menüobjekte	Menue

---

---

Merkmale von V38 (und höher)  
Wichtig

Objekte  
Objekte

Piktogrammobjekte  
Piktogramm

Postadresse  
Adresse des Autors

Prefs Verzeichnis  
Prefs

Programmers Verzeichnis  
Programmers

Programmkonzepte  
Konzepte

Programmobjekte  
Programm

Programmversionen  
Geschichte

Qualifier  
Qualifier

Quelltext  
Source

Rawkey  
InputEvent-Klassen

Rawmouse  
InputEvent-Klassen

Rechtliche Dinge  
Copyright

Referenz: Distributionsdateien  
Distributionsdateien

Referenz: Library Schnittstelle  
Library

Referenz: Tastenbefehle  
Tastenbefehle

Referenz: ToolManager-Objekte  
Objekte

Referenz: Voreinsteller  
Voreinstellungen

---

---

Schnellinstallation  
Schnellinstallation

Scripts Verzeichnis  
Scripts

Shell-Skripte  
Scripts

Source Verzeichnis  
Source

Spenden  
Adresse des Autors

Sprachdateien  
Locale

Sprachen  
Locale

Tastebefehle  
Tastebefehle

Tastenkodes für rawkey  
rawkey Tastenkodes

Tastenkodes für rawmouse  
rawmouse Tastenkodes

Texinfo  
Docs

Tonobjekte  
Ton

ToolManager-Objekte  
Objekte

UPD  
Goodies

Verbote  
Copyright

Versionen  
Geschichte

Voreinsteller  
Voreinstellungen

WBStart 1.2  
L

WBStart-Handler  
L

---

WBStartup Verzeichnis  
WBStartup

Wichtige Bemerkungen  
Wichtig

Zugriffsobjekte  
Zugriff