

# MUI - MagicUserInterface

---

Ein System zum Erstellen und Verwalten von grafischen Benutzeroberflächen

- Benutzerhandbuch -

Version 2.0  
03. Februar 1994

Stefan Stuntz

---



# 1 Einführung

## 1.1 Das Konzept von MUI

MagicUserInterface, abgekürzt MUI, ist ein komplettes System zum Erstellen und Verwalten von grafischen Benutzeroberflächen. Das Generieren solcher Oberflächen war auf dem Amiga schon immer ein großes Problem, weil dem Programmierer vom Betriebssystem dabei relativ wenig Unterstützung geboten wird. Durch die Einführung der `'gadtools.library'` unter Kickstart 2.0 war zwar ein erster Schritt in die richtige Richtung getan, aber auch damit ist das Erzeugen einer anspruchsvollen Oberfläche schwierig und langwierig. Es gibt zwar mittlerweile zahlreiche Hilfsprogramme, welche die Arbeit mit der `'gadtools.library'` vereinfachen sollen, doch auch die damit erzielten Ergebnisse sind meist alles andere als befriedigend.

Das größte Problem der herkömmlichen Tools liegt in deren relativ unflexiblen Ergebnis. Die meisten Programme arbeiten heute immer noch mit fest eingestellten Zeichensätzen und Fenstergrößen, obwohl das im Zeitalter von hochauflösenden Grafikkarten eigentlich eine Zumutung ist. Aber die Amiga-Benutzer sind ja in dieser Hinsicht leider sowieso nicht allzu verwöhnt, selbst die Preferences-Programme auf der Workbench verwenden alle noch den Standard-Zeichensatz `'topaz/8'`.

Hier will MUI Abhilfe schaffen. Der zentrale Gedanke besteht dabei darin, daß nur der Benutzer (und nicht der Programmierer) einer Applikation weiß, wie diese auf seinem persönlichen Gerät am Besten aussieht. Deswegen enthalten MUI Programme auch so gut wie keine absoluten Größen- und Positionsangaben, der Programmierer spezifiziert lediglich gewisse Gruppenzugehörigkeiten von Bedienungselementen. Das eigentliche Plazieren der Objekte wird dann während der Laufzeit des Programms nach den Vorgaben des Benutzers vom MUI System übernommen.

Eine MUI Applikation bietet ihrem Benutzer demnach viele wesentliche Vorteile gegenüber normalen Programmen:

- Zeichensatz-Sensitivität

Bei jedem MUI Programm können die verwendeten Zeichensätze frei eingestellt werden. Vorbei sind die Zeiten in denen A2024-Besitzer mit winzig kleinen topaz/8 Programmen kämpfen müssen. MUI schränkt die Auswahl der Zeichensätze in keinsten Weise ein, das gilt insbesondere auch für Proportionalschriften. Diese verleihen einem Programm zum einen ein wesentlich hübscheres und professionelleres Aussehen und verringern zum anderen den Platzbedarf erheblich.

- Veränderbare Fenstergrößen

Alle MUI-Fenster besitzen ein Sizing-Gadget mit dem sich die Größe beliebig bestimmen läßt. Je kleiner das Fenster, desto näher rücken die Bedienungselemente zusammen, je größer, desto mehr Platz ist für die Darstellung von Informationen vorhanden. Größe und Position eines Fensters lassen sich zudem dauerhaft abspeichern, damit man gleich nach dem Start einer Applikation seine Lieblingseinstellungen vor sich hat.

- Flexibilität

Nahezu alle Elemente einer Oberfläche können in ihrem Aussehen beeinflußt werden. Als Benutzer bestimmt man, wie dick die Rahmen sind, wie die Scrollbalken auszusehen haben, welche Images verwendet werden oder wieviel Platz zwischen den Zeilen eines Listviews eingefügt werden soll. MUI bietet hier eine Fülle von Möglichkeiten, auf die später noch genauer eingegangen wird.

- Tastatur-Bedienbarkeit

Normalerweise werden grafische Benutzeroberflächen, natürlich auch die von MUI, mit einer Maus bedient. Viele Benutzer würden allerdings auch gerne die Tastatur verwenden, weil das in manchen Fällen einfach schneller und bequemer ist. Deshalb lassen sich alle Objekte einer MUI Oberfläche, seien es nun String-Gadgets, Radio-Buttons oder Listen, ohne weiteres auch mit Tastatur steuern. Die Maus kann man dabei getrost bei Seite schieben, sie wird nicht mehr benötigt.

- System-Integration

MUI Applikationen arbeiten auf vielfältige Weise mit dem Betriebssystem zusammen. Jedes Programm kann etwa auf Knopfdruck (oder mit dem Commodities Exchange Programm) ikonifiziert und später wieder zum Leben erweckt werden. Außerdem besitzen alle Applikationen einen ARexx-Port, mit dem man unter anderem die komplette Benutzeroberfläche “fernsteuern” kann.

- Umgebungsanpassung

Einem MUI Programm ist es gleichgültig, ob es auf der Workbench oder einem anderem Public Screen läuft, ob die Auflösung 640x200 oder 1280x1024 Pixel beträgt oder ob 4 oder 256 Farben zur Verfügung stehen. Jede Applikation kann vom Benutzer auf beliebige Bildschirme geschaltet werden und paßt sich dort automatisch an ihre Umgebung an.

Alle eben genannten (und noch viele andere) Einstellungen können vom Benutzer einer Applikation mit Hilfe des MUI Preferences Programms vorgenommen werden und zwar je nach Notwendigkeit global oder für jede Applikation einzeln.

## 1.2 Systemanforderungen

MUI läuft mit allen Betriebssystemversionen ab Kickstart 2.0 und ist so programmiert daß es auch unter allen folgenden Versionen arbeitet. Einige besondere Features bleiben jedoch den Besitzern von aktuellen Betriebssystemversionen (Kickstart 3.0 oder aufwärts) vorbehalten. Kickstart 1.3 wird nicht mehr unterstützt, die Zeiten sind endgültig vorbei.

Dringend anzuraten ist außerdem eine Festplatte, obwohl MUI sich auch mit Diskettenlaufwerken zufrieden gibt. Bedingt durch das modulare Konzept kann es dabei jedoch beim ersten Starten einer Applikation zu etwas längeren Ladezeiten kommen.

MUI stellt keine besonderen Anforderungen an den Prozessortyp, aber auch hier gilt natürlich: je mehr desto besser. Für die Verwaltung und das Layout von Fenstern sind teilweise komplizierte Berechnungen nötig, die auf einem 68000er eventuell etwas langsam sind.

Speicher kann man sowieso nie genug haben, ein Megabyte ist zum Betrieb von MUI allerdings mehr als ausreichend. Bei (nur) 512 kByte könnte es je nach Applikation etwas knapp werden.

## 1.3 Installation

MUI wird mit dem Installer von Commodore ausgeliefert, deshalb ist die Installation denkbar einfach. Ein simpler Doppelklick auf **'MUI-Install'** startet den Vorgang, danach wird man vom Script weiter geführt.

Eine funktionsfähige MUI Installation besteht aus der **'muimaster.library'** im **'libs:'** Verzeichnis sowie den Unterlibraries für einige Bedienelemente im Verzeichnis **'libs:mui/'**. Außerdem sollte sich das Preferences Programm zusammen mit den Voreinstellern der Workbench im **'sys:Prefs'** Ordner befinden. Bei Verwendung des Installers braucht man sich über die Anordnung jedoch keine Gedanken zu machen, die Dateien werden automatisch an die richtigen Positionen kopiert.

## 2 Bedienen von MUI-Applikationen

### 2.1 Fenster

MUI Applikationen verhalten sich zunächst wie ganz normale Amiga-Programme und können auch so bedient werden. Allerdings bieten sie zusätzlich noch einige Features, die im folgenden etwas näher erläutert werden sollen.

Wie bereits in der Einleitung erwähnt, sind fast alle MUI Fenster in ihrer Größe beliebig veränderbar. Ein Benutzer kann selbst bestimmen, ob die Bedienungselemente eher klein und platzsparend oder größer und übersichtlicher erscheinen. Damit man nicht bei jedem Start einer Applikation seine bevorzugten Fenstergrößen (und Positionen) neu einstellen muß, merken sich MUI Fenster diese Werte automatisch und stellen sie bei einem erneuten Start des jeweiligen Programms wieder zur Verfügung. Das gilt sowohl für die normale Fenster-Position und Größe als auch für die entsprechenden Werte im "Zoomed"-Zustand (nach Betätigen des Zoom-Gadgets).

Nach einem Reset sind die gespeicherten Informationen über die Fenster gewöhnlich verloren, es sei denn man hat nachträglich im Abschnitt 3.1 [PRF'INTRO], Seite 8 die Speicher-Funktion aktiviert. Dadurch werden unter anderem sämtliche - die Fenster betreffenden Daten - gespeichert und stehen fortan dauerhaft zur Verfügung.

Alle Fenster einer Applikation enthalten außerdem im oberen Rahmen neben dem Depth- und dem Zoom-Gadget noch einen dritten Knopf. Dieser Knopf kann dazu benutzt werden, die gesamte Applikation zu ikonifizieren. MUI schließt daraufhin alle Fenster (und eventuell Screens) die zu der Applikation gehören und erzeugt ein kleines AppIcon auf der Workbench. Ein Doppelklick auf dieses AppIcon bringt die Applikation dann wieder zum Vorschein.

### 2.2 Tastatur

Alle Bedienungselemente in einem MUI-Fenster sind vollständig mit der Tastatur steuerbar. Die von herkömmlichen Programmen verwendeten Tastatur-Shortcuts (gekennzeichnet durch einen unterstrichenen Buchstaben in dem jeweiligen Gadget) werden natürlich unterstützt, diese Methode stößt jedoch bei Objekten wie Listviews oder Cycle-Gadgets schnell an ihre Grenzen.

Für MUI wurde deshalb das von String-Gadgets bekannte Prinzip des **Tab-Cyclings** aufgegriffen und erweitert. In MUI-Fenstern können mit **Tab** neben String-Gadgets auch alle anderen Objekte aktiviert werden. Ein aktives Objekt kann dann jeweils mit der Tastatur bedient werden:

- Button-Gadgets

**Return** entspricht einem Anklicken des Knopfs mit der Maus, dabei wird zwischen Drücken und Loslassen unterschieden. Ein bereits gedrückter Knopf kann (vor dem Loslassen von **Return**) durch **Shift** abgebrochen werden, ohne daß die dazugehörige Aktion ausgelöst wird.

- Checkmark-Gadgets

Ein aktives Checkmark-Gadget kann mit **Space** oder **Return** beeinflusst werden, der Status kehrt sich dadurch jeweils um.

- Slider

Die vier Cursor-Tasten, zusammen mit den entsprechenden Qualifiern beeinflussen den Wert von horizontalen oder vertikalen Slidern.

- Cycle-Gadgets

Bei aktivierten Abschnitt 2.3 [USE CYCLE], Seite 5 können die Einträge mit den Cursor-Tasten durchgeschaltet werden. Die **Return**-Taste läßt das zugehörige Popup-Menü aufklappen (falls dieses Feature nicht abgeschaltet wurde).

- Radio-Buttons

Auch hier dienen die Cursor-Tasten zur Steuerung.

- Listviews

In einem aktivierten Listview läßt sich der Cursor mit den Cursor-Tasten und den entsprechenden Qualifiern zeilenweise, seitenweise oder ganz nach oben bzw. unten bewegen. Die **Return**-Taste simuliert einen Doppelklick. Falls in dem Listview mehrere Einträge selektiert werden können, kann dazu **Space** verwendet werden.

- Fenster

Sollte eine Applikation mehrere, gleichzeitig bedienbare Fenster geöffnet haben, dann können mit **Alt-Tab** bzw. **Shift-Alt-Tab** diese Fenster der Reihe nach aktiviert werden. Ein Drücken von **Escape** wirkt wie ein Mausklick auf das Close-Gadget (falls vorhanden).

Sämtliche Erklärungen beziehen sich auf die Standard-Einstellungen, mit dem Preferences-Programm lassen sich alle verwendeten Tastenkombinationen beliebig einstellen.

## 2.3 Cycle Gadgets

In MUI Applikationen verwendete Cycle-Gadgets bieten neben ihrer normalen Funktion (nächster Eintrag durch anklicken, vorheriger Eintrag mit **Shift**) auch optional ein Popup-Menü.

Dieses Menü klappt auf, sobald der Text-Bereich im Cycle-Gadget angeklickt wird und erlaubt eine schnelle und übersichtliche Selektion eines der vorhandenen Einträge.

Das genaue Verhalten dieser Popup-Menüs kann im Preferences-Programm festgelegt werden (siehe Abschnitt 3.5 [PRF`LISTVIEWS], Seite 13).

## 2.4 Commodities Interface

Jedes MUI Programm bindet sich als Commodity ins System ein. Dadurch hat man als Benutzer die Möglichkeit, Applikationen mit dem ‘**Exchange**’-Programm der Workbench zu kontrollieren. Die Kontrolle besteht dabei im wesentlichen darin, Programme zu ikonifizieren oder ganz zu verlassen.

## 2.5 ARexx Port

Alle MUI-Applikationen können über einen integrierten ARexx-Port Befehle empfangen. Dabei stehen immer einige Standard-Befehle zur Verfügung, die jedes Programm versteht:

- QUIT  
Die Applikation wird beendet.
- HIDE  
Die Applikation wird ikonifiziert.
- SHOW  
Eine ikonifizierte Applikation wird wieder zum Leben erweckt.
- INFO ITEM/A

Je nach angegebenem Parameter wird der Ergebnis-String entsprechend aufgefüllt:

- “title” Titel der Applikation
- “author” Autor der Applikation
- “copyright” Copyright-Notiz
- “description” Kurzbeschreibung
- “version” Versionsstring
- “base” Name des ARexx Ports
- “screen” Name des Public-Screens



- `HELP FILE/A`

Eine Liste aller ARexx-Befehle wird in die angegebene Datei geschrieben. Zusätzlich zu den Standard-Befehlen kann (und sollte) eine Applikation natürlich noch eine Menge eigener Befehle definieren. Die Help-Liste wird dann auch diese Befehle enthalten.

Falls beim Bearbeiten der Befehle irgendwelche Fehler auftreten, werden die folgenden Werte als result code zurückgeliefert:

- -1

Fehlerhafte Definition des ARexx-Befehls im Programm. Sollte eigentlich nie passieren.

- -2

Kein Speicher für die Bearbeitung des Befehls.

- -3

Unbekannter ARexx-Befehl.

- -4

Syntax-Error in den Parametern eines Befehls.

Einige kleine Beispiel-Scripts sind der Distribution beigelegt und finden sich im ‘Rexx’-Ordner im Hauptverzeichnis.

## 3 Preferences-Programm

### 3.1 Allgemeines

Bei herkömmlichen Applikationen hat man als Benutzer normalerweise keine, oder nur sehr eingeschränkte Möglichkeiten, das Aussehen und die Funktionalität der Benutzeroberfläche zu beeinflussen. Man ist dem Programmierer gewissermaßen hilflos ausgeliefert und muß sich seinem Geschmack unterordnen bzw. anpassen.

Im Gegensatz dazu ist die Oberflächengestaltung bei MUI Applikationen wesentlich flexibler. Der Programmierer macht hier so gut wie gar keine Angaben über das eigentliche Aussehen von Bedienelementen, er legt lediglich deren Typ und gewisse Parameter fest. Wie diese Elemente dann letztendlich auf dem Bildschirm erscheinen, liegt fast ausschließlich in den Händen des Benutzers.

So würde zum Beispiel ein Programmierer im Falle eines File-Requesters lediglich spezifizieren, daß sein Fenster oben eine Dateiliste, darunter ein Pfad- und ein Datei-Eingabefeld und darunter wiederum einen OK- und einen Abbruch-Knopf besitzen soll. Er macht jedoch keinerlei Vorgaben bezüglich Größen, Farben oder Zeichensätzen. Alle diese, und noch viele andere Gesichtspunkte der Bedienoberfläche können vom Benutzer einer Applikation jederzeit nachträglich verändert und an den persönlichen Geschmack angepaßt werden.

Zum Vornehmen aller dieser Einstellungen dient nun das MUI Preferences Programm. Es befindet sich nach erfolgter Installation mit dem Installer im Prefs Ordner auf der System-Partition und wird einfach durch Doppelklick gestartet.

### 3.2 Hauptfenster

Das Hauptfenster des Preferences-Programms läßt sich in drei Bereiche untergliedern. Ganz oben befindet sich ein Textfeld mit Popup-Button und zwei weiteren Knöpfen rechts daneben, darunter dann die verschiedenen Einstellseiten für alle Konfigurationsparameter und schließlich am unteren Fensterrand noch eine Knopfleiste.

Mit dem Preferences-Programm ist es möglich, sämtliche Einstellungen nach Applikationen getrennt vorzunehmen. Im Textfeld links oben wird dazu der Name des betroffenen Programms angezeigt, mit dem Popup-Button daneben kann jederzeit die Applikation gewechselt werden. MUI

führt automatisch eine Liste aller bereits einmal gestarteten Applikationen, aus diesen kann in der Popup-Liste eine beliebige gewählt werden.

Meist wird man jedoch Parameter global, d.h. für alle Applikationen auf einmal, konfigurieren wollen. Dazu gibt es in der Liste der Programme einen speziell gekennzeichneten Eintrag namens **‘Global’**, der auch als erstes beim Start des Preferences-Programms aktiviert ist. Unter diese Überschrift getroffene Einstellungen wirken sich auf alle Applikationen aus, d.h. immer dann, wenn eine globale Einstellung verändert wird, überträgt diese sich automatisch auf die entsprechenden lokalen Werte, sofern diese vor der Änderung mit dem globalen Wert übereingestimmt haben. Wenn also zum Beispiel global der Zeichensatz **‘helvetica/13’** und speziell für das MUI-Demo Programm der Zeichensatz **‘Times/18’** konfiguriert ist, dann wirkt sich die globale Änderung von **‘helvetica/13’** auf alle anderen Applikationen, nicht aber auf das Demo-Programm aus.

Die allermeisten Parameter werden immer in den globalen Einstellungen konfiguriert werden, lokale Abänderungen sind nur bei relativ wenigen Feldern, etwa bei **‘Iconify-Hotkey’** und **‘Public-Screen’** sinnvoll. Theoretisch ist es aber ohne weiteres möglich, verschiedenen Applikationen völlig verschiedene Oberflächen-Outfits zu verpassen. In wie weit das auch sinnvoll ist, sei dahingestellt.

Der **‘Info’**-Knopf neben dem Applikationsnamen ist nur erreichbar wenn gerade lokale Einstellungen editiert werden. Damit werden zu dem entsprechenden Programm einige nützliche Informationen angezeigt, allerdings nur dann wenn das Programm gerade läuft.

Durch MUI’s automatische Applikationslistenverwaltung sammeln sich mit der Zeit einige Programme an, die man vielleicht inzwischen gar nicht mehr verwenden möchte. Mit dem **‘Delete’**-Knopf können solche unerwünschte Applikationen aus der Liste entfernt werden. Damit gehen natürlich alle Einstellungen die lokal für diese Programme getroffen wurden verloren.

Die eigentlichen Einstellungen, entweder global oder pro Applikation, werden dann im Hauptteil in der Mitte des Fensters vorgenommen. Hier befindet sich der - in Seiten aufgeteilte - Konfigurationsbereich. Die Seiten können mit dem Cycle-Gadget am oberen Rand, oder auch mit den Einträgen im Page-Menü, durchgeschaltet werden. Die einzelnen Elemente der verschiedenen Seiten werden in den folgenden Kapiteln ausführlich beschrieben.

Am unteren Fensterrand befinden sich die, von den System Preferences Programmen bekannten, **‘Save’**-, **‘Use’**- und **‘Cancel’**-Knöpfe sowie zusätzlich ein **‘Test’**-Gadget. Dieses **‘Test’**-Gadget, ist wohl das am meisten benutzte Gadget im Preferences Programm. Gerade zu Beginn wird man ein wenig mit den vielen Parametern experimentieren, bis man seine ideale Einstellung gefunden hat. Ein Druck auf **‘Test’** veranlaßt die aktuelle Applikation (oder alle, wenn gerade global konfiguriert

wird) dazu, sich automatisch an die neu spezifizierten Werte anzupassen. Man kann also, nachdem man eine Applikation gestartet hat, deren Einstellungen einem noch nicht zusagen, parallel dazu das Preferences Programm aufrufen und die neue Applikation sozusagen im direkten Dialog dem eigenen Geschmack anpassen.

Mit dem **‘Use’**-Gadget werden alle getroffenen Einstellungen temporär (im **‘env:’** Verzeichnis) abgelegt und das Preferences Programm beendet. Diese Einstellungen gehen natürlich mit dem nächsten Reset verloren.

Zur dauerhaften Speicherung dient das **‘Save’**-Gadget, alle Parameter werden auf Festplatte gesichert und stehen fortan immer zur Verfügung. Neben den konfigurierten Werten speichert MUI außerdem die Fensterpositionen einer Applikation.

Durch **‘Cancel’** werden alle Änderungen verworfen, Applikationen die sich eventuell durch die **‘Test’**-Funktion angepaßt haben, nehmen ihr altes Aussehen wieder an. Es erfolgt hier - im Einklang mit den System Preferences Programmen - keine Sicherheitsabfrage.

### 3.3 Fonts-Seite

Alle mit MUI erzeugten Benutzeroberflächen sind font-sensitiv, d.h. sie passen sich automatisch an beliebige Zeichensätze an. Normalerweise werden die Standard-Zeichensätze des Systems verwendet, natürlich lassen sie die Fonts aber auch direkt einstellen.

In MUI gibt es fünf Standard-Zeichensätze die von Applikationen genutzt werden können:

- **‘Normal’**

Dieser Zeichensatz wird für alle Texte benutzt für die nicht explizit ein anderer Zeichensatz konfiguriert ist.

- **‘List’**

Das ist der Default-Zeichensatz für Listviews.

- **‘Tiny’**

Der Tiny-Zeichensatz wird verwendet, um kleine, relativ unwichtige Beschriftungen anzubringen. Die Skala eines Scale-Objects (siehe **‘MUI-Demo’**) benutzt zum Beispiel diesen Font.

- **‘Fixed’**

Wenn ein Programm einen nicht-proportionalen Zeichensatz benötigt, tritt dieser hier in Aktion.

- ‘Title’

In diesem Zeichensatz werden Gruppentitel angezeigt.

Bei einem leeren Feld verwendet MUI entsprechende Standard-Zeichensätze, und zwar für den Fixed-Font den System-Default-Font und ansonsten den Default-Font des Screens. Empfehlenswert ist es, für den ‘Title’ Zeichensatz einen etwas kleineren Font zu wählen. Dadurch passen sich die Titel besser dem umgebenden Rahmen an und werden nicht vielleicht an unerwünschten Stellen abgeschnitten.

### 3.4 Frames-Seite

Rahmen sind ein wichtiges Element um eine Benutzeroberfläche übersichtlich zu gestalten und einzelne Gruppen von Objekten voneinander abzugrenzen. Rahmen sind aber auch Geschmackssache, deswegen kann man sie bei MUI alle in ihrem Aussehen beeinflussen.

Bei den in herkömmlichen Amiga-Programmen verwendeten Rahmen sind die vertikalen Linien immer doppelt so dick wie die horizontalen Linien. Dies rührt wohl noch aus den Zeiten her, in denen 640 x 256 die übliche Auflösung war, die Pixel waren dabei ungefähr doppelt so hoch wie breit. Im Zeitalter von Flickerfixern und hochauflösenden Grafikkarten etablieren sich jedoch immer mehr Auflösungen, bei denen das Verhältnis von Pixel-Breite zu Pixel-Höhe ungefähr 1:1 beträgt, es besteht also kein Grund mehr, Rahmen unnötig zu verbreitern. MUI bietet deshalb bei ‘Width’ die Möglichkeit, von den üblichen dicken Rahmen auf dünne Rahmen umzustellen.

Eingerahmte Gruppen können in ihrem Rahmen einen Titel beinhalten. Die Farbe dieses Gruppen-Titels kann mit der ‘Title’ Einstellung konfiguriert werden. Drei Möglichkeiten stehen hier zur Auswahl, entweder schwarz, weiß oder mit Schattenwurf.

Der Gruppentitel wird im oberen Teil eines Gruppenrahmens immer horizontal zentriert angezeigt. Mit ‘Pos.’ kann die vertikale Position dieses Titeltexes beeinflusst werden. Bei ‘centered’ erscheint der Titeltext auch vertikal zentriert in der Mitte des Rahmens, bei ‘above’ befindet sich die Grundlinie des Textes genau auf Rahmenhöhe. In Zusammenhang mit dieser Einstellung steht auch der innere Abstand eines Gruppenrahmens, der im nächsten Abschnitt besprochen wird.

Beim Erstellen von Rahmen in einer MUI-Applikation spezifiziert der Programmierer nicht das Aussehen sondern lediglich den Typ eines Rahmens. So bekommt zum Beispiel ein Button einen Button-Rahmen oder ein String-Gadget einen String-Rahmen. Wie diese Rahmen dann im Endeffekt aussehen, das bestimmt der Benutzer. Dazu sind alle möglichen Typen in einer Liste aufgeführt,

jeweils mit einem kleinen Bild welches das gerade eingestellte Aussehen anzeigt. Die einzelnen Rahmen werden wie folgt verwendet:

- Button  
für normale Button-Gadgets, wie zum Beispiel der ‘Edit’-Button unter dieser Liste.
- Image  
für kleine Buttons die nur ein Image enthalten, zum Beispiel die Pfeil-Gadgets in einem Scrollbalken.
- Text  
für Text-Felder die weder angeklickt noch editiert werden können und nur zur Information dienen, zum Beispiel Statuszeilen.
- String  
für String Gadgets.
- Read-List  
für Listviews die lediglich zur Anzeige einer Liste dienen und in denen nichts angeklickt werden kann.
- Input-List  
für Listviews in denen der Benutzer Einträge selektieren kann, zum Beispiel alle Listviews im Preferences-Programm.
- Prop  
für alle Proportional-Gadgets, unter anderem verwendet in Scrollbalken und Schieberegler.
- Gauge  
für Füllstandsanzeigen.
- Group  
zum Gruppieren von Objekten.
- Popup  
für die Rahmen der Cycle-Gadget Popup-Menüs.
- Virtual  
für die Einrahmung der (bisher selten benutzten) virtuellen Gruppen.
- Virtual  
für Slider-Gadgets.

Mit den rechts neben der Rahmen-Liste befindlichen Gadgets läßt sich das Aussehen des aktivierten Rahmens beeinflussen. Dabei kann mit dem ‘Type’-Cycle-Gadget aus einer Liste von vorgefertigten Designs eines ausgewählt werden. Alle Designs existieren in zwei Formen, entweder ‘Raised’ (herausragend) oder ‘Recessed’ (eingedrückt).

Zu einem Rahmen gehört außerdem noch der Abstand, der ihn von seinem inneren Objekt trennt. Dieser Abstand kann mit den vier Schiebereglern bestimmt werden. Normalerweise wird man wohl für ‘Left’ und ‘Right’ sowie ‘Top’ und ‘Bottom’ identische Werte einstellen, deswegen sind diese Gadgets einseitig miteinander verbunden. Sobald also der ‘Left’- bzw. ‘Top’-Slider bewegt wird, bewegen sich ‘Right’- bzw. ‘Bottom’-Slider automatisch mit. Um dennoch getrennte Einstellmöglichkeiten zu gewährleisten, gilt diese Verbindung nicht in der anderen Richtung, ein Verschieben von ‘Right/Bottom’ beeinflußt die Positionen von ‘Left/Top’ nicht. Je nach Zeichensatz kann es durchaus sinnvoll sein, zum Beispiel beim Button-Frame den unteren Abstand etwas kleiner als den oberen zu wählen, um damit durch Unterlängen von Buchstaben eventuell verursachte Unsymmetrien auszugleichen.

Rechts oben auf der Rahmen-Seite können noch einige Werte eingestellt werden, die nicht direkt mit einem speziellen Rahmen verbunden sind, aber dennoch zum Komplex Rahmen und Abstand gehören.

Die vier Schieberegler unter ‘Window’ bestimmen dabei pixelgenau den Abstand, der zwischen dem Rahmen des Fensters und dessen Inhalt eingefügt wird. Auch hier wird man üblicherweise für ‘Left’ und ‘Right’ sowie ‘Top’ und ‘Bottom’ identische Werte einstellen, deswegen sind auch diese Gadgets einseitig miteinander verbunden.

Die beiden ‘Group’-Werte beeinflussen den Abstand der zwischen Elementen von horizontalen bzw. vertikalen Gruppen eingefügt wird. Gerade damit, und mit der Einstellung für den ‘Group-Frame’ kann man Oberflächen entweder eng und platzsparend oder aber weit und aufgelockert gestalten. Die ‘Radio’ Schieberegler schließlich sind für das Aussehen der (selten benutzten) Radio-Button Gadgets verantwortlich.

### 3.5 Lists-Seite

Hier finden sich gesammelt einige Konfigurationsmöglichkeiten für Listen, Popup-Menüs und in verschiedene Seiten aufgeteilte Gruppen.

Der ‘Leading’-Wert bestimmt die Anzahl der zusätzlichen Pixel, die zwischen den Zeilen eines Listviews eingefügt werden sollen um die Lesbarkeit zu erhöhen. Je nach verwendetem Zeichensatz und persönlichem Geschmack können hier durchaus größere Werte sinnvoll sein, insbesondere bei kleinen Fonts wie ‘topaz/8’ wird die Übersichtlichkeit dadurch erheblich erhöht.

Mit ‘Smoothing’ kann der bei vielen Leuten beliebte “Nachzieh-Effekt” eines Listviews eingestellt werden. Dieser Effekt bewirkt, daß die Position einer Liste beim Scrollen nicht unmittelbar an den

Scrollbar angepaßt wird, sondern diesem in, von der Geschwindigkeit der Bewegung abhängigen Schritten, folgt. Eine '0' verhindert das Nachziehen vollständig.

Das Smoothing ist nur dann aktiv wenn die Listviews auch wirklich pixelorientiert arbeiten. Normalerweise wird immer nur um ganze Zeilen gescrolled, die Pixel-Orientierung muß erst mit dem 'Pixel'-Gadget eingeschaltet werden.

Bei Listviews die das gleichzeitige Anwählen von mehreren Einträge gestatten, den sogenannten Multi-Select-Listviews, kann der Benutzer zwischen zwei verschiedenen Selektions-Mechanismen wählen. Bei 'Shifted' muß, ähnlich der Workbench, nach der ersten Selektion die Shift-Taste gehalten werden, sonst werden die bereits selektierten Einträge wieder deselektiert. Mit 'Always' kann das Drücken der Shift-Taste unterbleiben.

'Refresh' bestimmt die Art in der eine Liste neu aufgebaut wird: bei 'linear' werden die Zeilen wie üblich von oben nach unten gedruckt, 'intermixed' verursacht einen ineinander verschränkten Aufbau.

Die Position der Pfeil-Gadgets an den Scrollbalken eines Listviews kann mit dem Arrows-Gadget beeinflusst werden. Hier stehen drei Möglichkeiten zur Auswahl.

MUIs Cycle-Gadgets bieten als Bedienungserleichterung ein Popup-Menü, das bei Betätigen das Gadgets aufklappt und eine einfache und schnelle Selektion des gewünschten Eintrags zuläßt. Mit 'Level' kann man bestimmen, ab wieviel Einträgen ein Cycle-Gadget solch ein Popup-Menü zur Verfügung stellen soll. Falls man die Popup-Menüs gar nicht mag, stellt man diesen Wert einfach genügend groß ein und wird sie nie zu Gesicht bekommen.

Normalerweise erscheinen die Popup-Menüs immer direkt unter dem Gadget. Zwecks schnellerer Bedienung und Minimierung der Mausbewegung kann man sie aber auch so konfigurieren, daß der gerade aktive Eintrag immer direkt unter dem Mauszeiger erscheint. Dazu dient das 'Position' Gadget mit seinen beiden Einstellmöglichkeiten.

Die Benutzung eines bisher im Amiga-Betriebssystem nicht bekannten Bedienelements, das sogenannte 'Register', kann mit den beiden folgenden Slidern bestimmt werden. Dieser 'Register' wird immer dann verwendet, wenn in einem Fenster mehrere Seiten vorhanden sind unter denen der Benutzer beliebig umschalten kann. Normalerweise werden solche Seiten von MUI, wie beim Amiga üblich, mit einem Cycle-Gadget am oberen Gruppenrand implementiert. Allerdings kann diese Darstellung auch durch eine Art Karteikasten ersetzt werden, wie er inzwischen bei vielen anderen Betriebssystemen üblich ist. Mit 'Register Level' bestimmt man, bis zu welchen Verschachtelungstiefen Karteikästen anstelle von Cycle-Gadget verwendet werden sollen. Stell man



hier zum Beispiel den Wert 1 ein, dann werden äußere Seitengruppen immer als Karteikasten dargestellt, falls darin jedoch weitere Gruppen vorhanden sind, erscheinen diese dann wie gewohnt mit einem Cycle-Gadget. Solche Verschachtelungen tauchen unter anderem im Preferences-Programm auf der Images-Seite auf.

Karteikästen sind übersichtlicher und erlauben schnellere Bedienung als Cycle-Gadgets. Allerdings verlieren sie, abhängig von Fenster- und Zeichensatzgrößen, ab einer gewissen Anzahl von Seiten ihren Sinn. Deswegen kann man unter **‘Max Pages’** einstellen, ab welcher Seitenzahl die Karteikästen unabgänglich vom **‘Register-Level’** immer durch Cycle-Gadgets dargestellt werden sollen.

### 3.6 Images-Seite

Grafische Benutzeroberflächen arbeiten oft mit kleinen Bildern, im folgenden auch Images genannt. Die am häufigsten verwendeten Images sind wohl die Pfeile, sichtbar etwa im Scrollbalken eines Listviews. Für viele derartige Positionen stellt MUI Standard-Images zur Verfügung. Das hat zum einen den Vorteil, daß nicht jeder Programmierer selbst solche Images erzeugen muß, zum anderen kann der Benutzer das Aussehen frei einstellen.

Außerdem kann man unter MUI für viele Objekte verschiedene Hintergrund-Farben/Pattern einstellen. Wenn man das Glück hat, mit einer leistungsfähigen Grafikkarte arbeiten zu können, sollte man hier unbedingt ein wenig experimentieren. Leicht unterschiedliche Grautöne für Fenster-, Button- und ListView-Hintergründe geben einer Oberfläche ein wesentlich professionelleres und vor allem übersichtlicheres Aussehen als man es von normalen Programmen gewohnt ist.

Hier nun eine Aufstellung der vorhandenen Images und Hintergründe, wie sie sich auch in der Liste im Preferences-Programm befinden.

- **‘ArrowUp’**, **‘ArrowDown’**, **‘ArrowLeft’**, **‘ArrowRight’**

Vier Pfeile in die verschiedenen Richtungen.

- **‘CheckMark’**, **‘Radio-Button’**, **‘Cycle’**

Die bekannten Images aus den jeweiligen Bedienungselementen. Das Besondere am Checkmark-Image ist, daß es nur im selektierten Zustand sichtbar ist.

- **‘PopUp’**, **‘PopFile’**, **‘PopDrawer’**

Images für Popup-Buttons neben String-Gadgets. Wenn Dateien oder Verzeichnisse abgefragt werden sollen, wird **‘PopFile’** bzw. **‘PopDrawer’** verwendet, ansonsten **‘PopUp’**.

- ‘Drawer’, ‘HardDisk’, ‘Disk’, ‘Chip’, ‘Volume’, ‘Network’, ‘Assign’  
Images für Einträge in einem File-Requester.
- ‘TapePlay’, ‘TapePlayback’, ‘TapePause’, ‘TapeStop’, ‘TapeRecord’, ‘TapeUp’, ‘TapeDown’  
Images für Tapedeck-Anwendungen.
- ‘Prop-Knob’, ‘Slider-Knob’  
Schieberegler im Proportionalgadget und Slidern.
- ‘BG Window’  
wird überall da verwendet wo sonst kein anderer Hintergrund zutrifft, also insbesondere da wo sich keine Objekte befinden.
- ‘BG Groups’  
befindet sich hinter dem Inhalt von Karteikästen oder virtuellen Gruppen.
- ‘BG Requester’  
der Hintergrund für MUI-Requester, findet sich zum Beispiel hier im ‘About’-Requester.
- ‘BG Textfield’  
eingerahmte Text-Felder, zum Beispiel Statuszeilen, werden mit diesem Hintergrund hinterlegt.
- ‘BG Button’  
wird für große Buttons die üblicherweise Text enthalten oder auch für Cycle-Gadgets verwendet.
- ‘BG Selected Gadget’  
ein mit der Maus angeklicktes Gadget wird (neben der Invertierung des Rahmens) durch diesen Hintergrund kenntlich gemacht.
- ‘BG Listview’  
erscheint hinter den Zeilen eines Listviews.
- ‘BG Listview Cursor’  
der Cursor in einem Listview.
- ‘BG Listview Selected’  
selektierte Einträge in einem Listview.
- ‘BG Listview Selected+Cursor’  
der Cursor auf einem selektierten Eintrag im Listview.
- ‘BG Prop-Gadget Container’  
entspricht dem Hintergrund in einem Proportional Gadget, also der Bereich auf dem sich der Schieberegler hin und her bewegt.
- ‘BG Slider Container’  
entspricht dem Hintergrund in einem Slider Gadget.

Für jedes aktivierte Image kann auf der rechten Seite das Aussehen nahezu beliebig eingestellt werden. MUI bietet dabei einige verschiedene Möglichkeiten an, die mit dem Cycle-Gadget (oder mit der Karteikasten-Gruppe) angewählt werden können.

- Pattern

Ein Pattern ist ein wenig kompliziertes Muster, einige davon sind bereits in MUI eingebaut. Pattern eignen sich hauptsächlich als Hintergrund-Image, können aber auch für einige Standard-Images, etwa für den Prop-Gadget Knopf sinnvoll verwendet werden.

- Builtin

Für (fast) alle Standard-Images ist in MUI jeweils ein Typ fest eingebaut. Diese eingebauten Images sind aus Vektoren zusammengesetzt und können sich daher in ihrer Größe jeweils an den verwendeten Zeichensatz anpassen.

- Pen

Hier kann einfach ein ausgefülltes Rechteck in einer bestimmten Farbe gewählt werden. Für die Bestimmung dieser Farbe stehen wiederum verschiedene Möglichkeiten zur Auswahl. Entweder kann einer der aus den System-Voreinstellern bekannten System-Pens oder ein direkter Eintrag in der Colormap des Bildschirms referenziert werden. Letzteres ist vor allem in Zusammenhang mit einer installierten MagicWorkbench von Martin Huttenloher von großer Bedeutung. Die Einführung von zwei zusätzlichen Pens (Halfshine und Halfshadow) bietet ungeahnte Möglichkeiten beim Oberflächen-Design. Eine dritte, nur unter Kickstart 3.0+ vorhandene Möglichkeit eine Farbe zu bestimmen bietet das Farbrad. Hier kann einfach eine beliebige Farbe vorgegeben werden, MUI versucht dann, diese falls benötigt zu allokalieren. Das ganze ist nur dann sinnvoll, wenn der entsprechende Bildschirm auch genug unbenutzte Farben zur Verfügung stellt.

- Boopsi

Ein Boopsi-Image ist eigentlich ein Programm, das immer dann aufgerufen wird wenn es gezeichnet werden soll. Solche Images liegen als “shared library” vor und sollten sich im Ordner ‘`sys:classes/Images`’ auf der System-Partition befinden. Bei MUI liegen einige dieser Images bei die insbesondere für den Schieberegler eines Proportional-Gadgets gedacht sind.

- Brush

Das sind in einer speziellen Farbpalette abgespeicherte IFF Brushes. Diese Farbpalette erlaubt MUI, die Images an die Farben des Bildschirms anzupassen, auf dem diese dargestellt werden sollen. Auf der MUI-Diskette werden viele solcher Images mitgeliefert, natürlich können auch mit einem entsprechenden Malprogramm eigene erstellt werden. Brushes sind nicht in der Größe veränderbar, erlauben aber durch ihr pixelgenaues Design bestmögliche Qualität.

- Alien (nur Kickstart 3.0 oder höher)

Ab Kickstart 3.0 gibt es im Amiga Betriebssystem die sogenannten ‘Datatypes’. Damit ist es möglich, beliebige Bild-Dateien, seien es nun IFF, GIF oder sonstige Formate, mit einigen weni-

gen Befehlen einzuladen. MUI unterstützt diese Datatypes und erlaubt damit die Verwendung von beliebigen Bildern als Hintergrund oder Standard-Image in allen Applikationen.

Zum Einstellen des aktiven Images genügt ein Doppelklick auf den gewünschten Eintrag in einer der Listen. Bei der großen Anzahl kann das jedoch schnell in einer großen Klickorgie enden, deswegen bietet der 'Guess' Button die Möglichkeit, viele Images auf einmal zu konfigurieren. Dabei müssen zunächst in der linken alle einzustellenden Images mittels Multiselect angeählt werden, nach einem Druck auf 'Guess' ordnet MUI dann diesen Images automatisch entsprechende Einträge aus der gerade angezeigten rechten Liste zu.

### 3.7 Pens-Seite

MUI verwendet beim Zeichnen der Benutzeroberflächen nicht direkt die System-Pens sondern definiert hier um flexibel und erweiterbar zu sein eigene Stifte. Vorhanden sind dabei die folgenden Einträge:

Shine

identisch mit dem entsprechenden System-Stift.

Halfshine

verwendet in diversen eingebauten Rastern und beim Rahmen-Design XEN.

Background

identisch mit dem entsprechenden System-Stift.

Halfshadow

verwendet in diversen eingebauten Rastern und beim Rahmen-Design XEN.

Shadow

identisch mit dem entsprechenden System-Stift.

Text

identisch mit dem entsprechenden System-Stift.

Fill

identisch mit dem entsprechenden System-Stift, wird in MUI nur selten um diverse der eingebauten Raster zur erzeugen.

ActiveObj

Diese Farbe wird für die Umrandung von aktiven Objekten verwendet.

Die Einstellung der Pens erfolgt dabei wie auf der Images-Seite mit der entsprechenden Gruppe.

## 3.8 System-Seite

Auf der System-Seite finden sich einige Einstellungen, welche die Zusammenarbeit von MUI mit dem Betriebssystem betreffen. Dies sind die einzigen Einstellungen, die auch für nicht registrierte Benutzer verfügbar sind.

An oberster Stelle findet sich hier ein String-Gadget mit Popup-Button, in dem der Name eines Public-Screens eingestellt werden kann, auf dem sich die Fenster einer Applikation öffnen sollen. Über die Popup-Liste erreicht man den in MUI integrierten Screen-Manager, mit dem solche Public-Screens erzeugt werden können.

Zum Refresh von MUI-Fenstern stehen zwei Möglichkeiten zur Auswahl, die mit dem **‘Window Refresh’** Gadget eingestellt werden können. Der **‘smart’** Refresh braucht mehr Chip-Memory, ist dafür aber schneller als der **‘simple’** Refresh.

Beim Neuzeichnen eines Fensters bietet MUI ebenfalls zwei verschiedene Möglichkeiten an, die mit dem **‘Redraw’** Gadget eingestellt werden können. In der Einstellung **‘fast/ugly’** wird vor einem Neuzeichnen (nach einer Resize-Operation) zunächst der komplette Fensterinhalt gelöscht, das nachfolgende Zeichnen geht dann etwas schneller als im **‘slow/nice’** Modus wo das Löschen unterbleibt.

Bei **‘Startup’**- und **‘Shutdown’**-Command können zwei Befehle eingetragen werden, die automatisch vor dem Start bzw. nach dem Ende einer Applikation ausgeführt werden sollen. Hier könnte zum Beispiel mit einem externen Screen-Manager ein Public-Screen für die Applikation geöffnet werden, sofern die Fähigkeiten des integrierten Screen-Managers nicht ausreichen. Oder ein simpler echo-Befehl könnte in einem Logfile mitspeichern, wann eine Applikation gestartet und beendet wurde.

Mit dem **‘Iconify-Hotkey’** kann man eine Tastenkombination festlegen, mit der eine bestimmte Applikation ikonifiziert (und wieder hervorgeholt) werden kann. Das Format für den Eintrag entspricht dem Standard Input-Description Format der commodities.library.

Wenn **‘Iconify-Gadget’** angeschaltet ist, dann bekommt jedes Fenster der Applikation im Rahmen ein zusätzliches Gadget, das dann bei Betätigung den Iconify-Prozeß auslöst.

Normalerweise wird für eine ikonifizierte Applikation auf der Workbench ein AppIcon erzeugt. Durch Doppelklick auf dieses AppIcon kann dann die Applikation wieder aktiviert werden. Bei ausgeschaltetem **‘Iconify-Icon’** erscheint kein solches Icon, zum Aktivieren bleibt nur noch die

Möglichkeit eines eventuell konfigurierten Hotkeys oder ein Benutzen des Commodities Exchange Programms.

Im Listview am rechten Rand der Seite können alle Tasten konfiguriert werden, die zum Steuern von MUI Applikationen verwendet werden. Die Einträge in der Liste sprechen für sich, das Format ist wie üblich das Standard-Format für die Input Event Beschreibungen der `'commodities.library'`. Änderungen werden hier nur nach Bestätigung durch die **Return**-Taste im String-Gadget akzeptiert. Vorher wird der eingegebene String allerdings noch auf Korrektheit überprüft und gegebenenfalls zurückgewiesen.

Besondere Beachtung verdient der **'Press'**-Key. Bei dieser Taste ist MUI darauf angewiesen, auch beim Loslassen entsprechende Aktionen durchführen zu können. Deswegen muß diese Taste als Qualifier-Beschreibung unbedingt den String **'-upstroke'** enthalten.

### 3.9 Public-Screen-Manager

Beim Starten einer Applikation sucht MUI nach dem für diese Applikation eingestellten Public-Screen. Falls dieser nicht vorhanden ist, wird in der Liste des MUI-Screen-Managers nach einem entsprechenden Eintrag gesucht und der Bildschirm falls gefunden mit den eingestellten Spezifikationen geöffnet.

Den Public-Screen Manager erreicht man durch Betätigen der **'New'**- und **'Edit'**-Knöpfe in der Popup-Liste beim Einstellen des Public-Screens einer Applikation. Er läuft unabhängig vom Rest des Preferences-Programms in einem separaten Fenster, bei Bedarf können auch mehrere Manager gleichzeitig geöffnet werden.

Hier können Auflösung, Farbpalette sowie einige weitere Daten eines neuen Public-Screens eingestellt werden. Das Fenster ist deswegen auch in drei Seiten unterteilt, die mit dem Cycle-Gadget am oberen Rand umgeschaltet werden können.

Auf der **'Attributes'**-Seite finden sich im einzelnen folgende Einstellmöglichkeiten:

Im **'Public Name'** String Gadget kann man einen Namen festlegen, unter dem der Bildschirm später referenziert wird. Um Verwechslungen zu vermeiden, sollten alle im System verwendeten Public Screens unterschiedliche Namen haben. Hinter **'Title'** verbirgt sich wie erwartet der Text, der in der Titelzeile angezeigt wird und bei **'Font'** läßt sich der Default-Zeichensatz einstellen, mit dem der Bildschirm und die darauf geöffneten Fenster dargestellt werden sollen. Das **'Background'**-Gadget ist nur unter Kickstart 3.0 oder höher verfügbar und erlaubt die Konfiguration eines Hinter-

grundbildes. Hier werden wieder die **‘Datatypes’** verwendet, man kann dadurch die verschiedensten Bildformate verwenden.

Neben diesen essentiellen Werten kann ein Bildschirm noch einige andere Eigenschaften besitzen, die im folgenden aufgezählt werden:

- **‘Auto Scroll’**

Wenn der Screen größer als der sichtbare Bereich definiert wurde, dann wird er automatisch gescrollt sobald die Maus an einen Rand stößt.

- **‘Draggable’**

Screens ohne dieses Attribut können nicht verschoben werden.

- **‘Exclusive’**

Der Bildschirm kann das Display nicht mit anderen Bildschirmen teilen, er wird immer alleine dargestellt (erst ab Kickstart 3.0).

- **‘Interleaved’**

Dieses Attribut vermindert falls gesetzt das Flimmern das besonders beim Scrollen von Listen auf vielfarbigen Screens auftritt (erst ab Kickstart 3.0).

- **‘Open Behind’**

Der Screen wird unauffällig hinter allen anderen Screens geöffnet.

- **‘System Default’**

Der Screen wird zum System Default Screen erklärt. Alle Fenster die auf dem System Default Screen aufzugehen wünschen (z.B. Shell-Fenster) werden dadurch umgelenkt.

Die Auflösung und Größe eines Bildschirms läßt sich auf der **‘Display Mode’** Seite einstellen. In einer Liste werden dazu alle verfügbaren Modi angezeigt. Außerdem finden sich am unteren Fensterrand Gadgets für Breite, Höhe und Tiefe sowie eine Einstellung für den zu verwendenden Overscan-Typ.

Die Farbpalette und die Zuordnung der System-Stifte eines Bildschirms kann man schließlich mit Hilfe der **‘Palette’**-Seite bestimmen. Das hier verwendete Verfahren ist ähnlich des im System benutzten Palette-Voreinsteller-Programms. Anzumerken ist hier allerdings daß die gerade eingestellten Farben nur unter Kickstart 3.x und auch da nur dann auch wirklich sichtbar sind, wenn genug freie Farben auf dem Bildschirm zur Verfügung stehen. Die Mehrheit aller Benutzer wird momentan leider nur schwarze Kästchen sehen und ist beim Einstellen etwas auf ihre Phantasie angewiesen. Die MUI Palette-Einstellung wird erst bei mindestens 32 Farben so richtig schön.

### 3.10 CLI-Interface

Das Prefs-Programm bietet auch ein kleines CLI-Interface, das eigentlich nur dazu dient, den eingebauten Screen-Manager auch anderen Programmen zur Verfügung zu stellen. Die Syntax für den Aufruf mit Parametern lautet:

`NAME, OPEN/S, CLOSE/S`

NAME: Name eines (vorher konfigurierten) Public Screens

OPEN: Screen öffnen

CLOSE: Screen schließen

### 3.11 ARexx Port

Das Preferences Programm enthält einen einfachen ARexx-Port mit den folgenden vier Befehlen:

`'SAVE'`

Entspricht der Funktion des `'Save'`-Knopfs.

`'USE'`

Entspricht der Funktion des `'Use'`-Knopfs.

`'TEST'`

Entspricht der Funktion des `'Test'`-Knopfs.

`'CANCEL'`

Entspricht der Funktion des `'Cancel'`-Knopfs.

Natürlich können daneben auch die Abschnitt 2.5 [USE`AREXX], Seite 6 verwendet werden.



## 4 Sonstiges

### 4.1 Registration

“MagicUserInterface” ist ein umfangreiches Produkt, in das ich viel Arbeit und Zeit (und wahrscheinlich ein Semester meines Studiums) investiert habe. Ich hoffe allerdings, daß sich diese Arbeit gelohnt hat und daß schon bald viele - auf MUI basierende - Applikationen mit schönen und flexiblen Benutzeroberflächen verfügbar sind.

Damit auch der finanzielle Aspekt nicht ganz unberücksichtigt bleibt, habe ich mich entschlossen, MUI als Shareware zu vertreiben. Mit der unregistrierten Version ist das Speichern einiger Einstellungen im Preferences-Programm nicht möglich. Diese Einschränkungen beeinflussen allerdings in keinsten Weise die Funktionalität von Applikationen, alle wichtigen Parameter (etwa Fenster-Positionen, Public-Screens und System-Einstellungen) sind auch ohne Registrierung möglich. Für die übrigen Werte werden dann die integrierten Standard-Einstellungen verwendet, auch damit sind MUI Applikationen immer noch funktioneller und attraktiver als die meisten anderen Programme.

Wer allerdings die weitergehenden Möglichkeiten von MUI (verschiedene Zeichensätze, Rahmen, Images, Hintergrund-Pattern) nutzen möchte, der sollte sich registrieren lassen. Registrierte Benutzer erhalten eine Diskette mit der jeweils aktuellsten Version, zusammen mit einem Keyfile. Mit diesem Keyfile können dann sämtliche Einstellungen im Preferences-Programm abgespeichert werden und stehen dauerhaft zur Verfügung. Natürlich bleibt das Keyfile für alle zukünftigen Versionen von MUI gültig, spätere Updates können also einfach von Mailboxen oder PD-Disketten bezogen werden.

Der Preis für eine Registrierung beträgt

20.- DM (D-Mark),  
20.- SFr (Schweizer Franken),  
90.- FF (French Francs),  
15.- US\$ (US-Dollar)

oder einen Betrag in irgendeiner anderen Währung der 20.- US\$ (zwanzig!) entspricht. Zwanzig deswegen weil ich das Geld dann bei meiner Bank umtauschen und dafür Gebühren bezahlen muß.

Als schnellster und einfachster Weg sich registrieren zu lassen bietet sich an, das beiliegende Bestellformular auszufüllen und es in einem Brief zusammen mit dem Geld an die folgenden Adresse zu schicken:

Stefan Stuntz  
Eduard-Spranger-Straße 7  
80935 München  
DEUTSCHLAND

Statt Bargeld können natürlich auch Euro-Cheques verwendet werden, allerdings bitte ich von jeglicher Art von Auslands-Cheques abzusehen, diese einzulösen kostet teilweise unverschämt hohe Gebühren.

Innerhalb Deutschlands kann man die Registration auch einfach per Banküberweisung an die

Stadtsparkasse München, BLZ 701 500 00, Konto 35-169929

durchführen. Dabei ist unbedingt darauf zu achten, daß auf dem Überweisungsformular unter 'Verwendungszweck' die komplette Anschrift angegeben wird.

Alle Registrierungen werden schnellstmöglich bearbeitet und sollten im Inland innerhalb von ein bis zwei Wochen ausgeliefert werden. Bedingt durch die Postlaufzeiten kann eine Auslandsregistrierung etwas länger dauern. Deswegen gibt es für MUI in einigen Ländern Registration Sites:

– U.S.A. und Kanada

Robert Blayzor  
P.O. Box 807  
Johnstown, NY 12095-0807

Phone: (518) 883-5326 (data/fax/bbs)

electronic mail:

InterNet/UUCP: robertb@liquid.albany.ny.us  
FidoNet: 1:267/131.0  
AmigaNet: 40:714/1.0  
C-Link: 911:6150/1.0

Make all payments payable to: Robert Blayzor

Acceptable payment methods: US Money Orders & Bank/Personal Checks

Personal checks must wait 10-15 days to clear unless certified!  
(All payments in US dollars ONLY!)

## 4.2 Updates

Wann immer eine neue Version von MUI erscheint, wird das in den entsprechenden Brettern einiger Datennetze angekündigt werden. Das neue Archiv findet sich dann sicher bald in vielen Mailboxen und ist außerdem auf allen ‘**aminet**’ ftp-servern zu bekommen. Größere Updates werden auch auf PD-Disks erhältlich sein.

Wie bereits erwähnt, brauchen registrierte Benutzer weder ein neues Keyfile noch irgendwelche speziellen Programmversionen, Updates können sofort mit allen ihren Möglichkeiten genutzt werden.

Natürlich werden alle zukünftigen MUI Versionen zu den vorher erschienenen vollständig kompatibel sein. Applikationen die mit MUI 1.0 entwickelt worden sind, werden auch noch nach geraumer Zeit unter MUI 7.0 ihren Dienst versehen. Sie werden dabei automatisch von eventuellen Verbesserungen im Design oder in der Funktionalität der Oberfläche profitieren.

## 4.3 Support

Ich werde mich bemühen, alle Anfragen bezüglich MUI schnellstmöglich zu beantworten. Dabei werden die Probleme von registrierten Benutzern bevorzugt behandelt. Bitte wann immer möglich electronic mail als Kommunikationsmedium verwenden, das ist sowohl billiger als auch schneller, schont die Umwelt und erleichtert die Arbeit. Electronic mail beantworte ich normalerweise noch am selben Tag, bei Papierbriefen kann das ganze bedeutend länger dauern. Dann ist vielleicht ein Anruf eher angesagt.

Sackpost: Stefan Stuntz  
~       Eduard-Spranger-Straße 7  
~       80935 München  
~       DEUTSCHLAND

Telefon: +49-89-313-1248

e-mail: [stuntz@informatik.tu-muenchen.de](mailto:stuntz@informatik.tu-muenchen.de)

## 4.4 Danksagungen

MUI entstand aus der Enttäuschung darüber, wie wenig Unterstützung einem Programmierer beim Erstellen von Benutzeroberflächen seitens des Betriebssystems gewährt wird. Meine ersten

Denkanstöße in Richtung objektorientierter Oberflächengestaltung erhielt ich von Armin Sander, bei dem ich mich an dieser Stelle für die vielen Tips herzlich bedanken möchte. Er hat damit den Grundstein für MUI gelegt.

Ohne die tatkräftige Mithilfe meiner Betatester wäre MUI sicher niemals fertiggestellt worden. Dabei beschränkten sich ihre Aufgaben nicht nur auf bloßes Ausprobieren sondern vor allem auch auf das Erstellen der mitgelieferten Beispielprogramme. Daß aus einigen dieser Beispielprogramme richtige Applikationen geworden sind, macht das große Engagement und die Begeisterung mit der sie alle bei der Sache waren um so mehr deutlich.

Im einzelnen haben mitgeholfen:

- Stefan Becker  
... hat trotz chronischen Zeitmangels einige wertvolle Tips und Hinweise gegeben. Außerdem waren mir Teile seines ToolManager Sourcecodes bei der Entwicklung von MUI eine große Hilfe.
- Martin Berndt ... hat mich beim Lösen einiger kniffliger Probleme unterstützt.
- Robert Blayzor ... hat die englische Anleitung überarbeitet.
- Dirk Federlein ... erstellte die MUI-Applikation 'DFView'. Bei über 100 kByte Sourcecode inclusive Sprachen-Anpassung und Anleitung in drei verschiedenen Formaten wage ich es nicht mehr, nur von einem Beispielprogramm zu sprechen. Außerdem hat Dirk zu meiner großen Freude einige Teile dieser Dokumentation ins Englische übersetzt.
- Georg "gucky" Heßmann  
... hat in der Endphase noch einige Bugs aufgedeckt und das Demo-Programm 'DVIprint' beigesteuert.
- Martin Horneffer und Albert Weinert  
... von ihnen stammt das Oberon-Interface.
- Martin "XEN" Huttenloher  
... hat viele der mitgelieferten Images gezeichnet und auch beim sonstigen MUI-Design entscheidend mitgewirkt. Außerdem stammen von ihm die wunderschönen Hintergrund-Patterns, die ein kleiner Auszug seines 'MagicWB'-Pakets sind. Freunde einer ansprechenden, plastischen Workbench sollten sich sein Paket 'MagicWB' unbedingt einmal näher betrachten!
- Kai "KCommodity" Iske  
... programmierte einen der diversen MUI-Taschenrechner und hat dabei noch einige üble Bugs in MUI gefunden.
- Oliver "Mr.Coffee" Kilian  
... hat MUI auf dem guten alten (und langsamen) 68000er getestet.

- Klaus “kmel” Melchior  
... schrieb die beiden Beispiel-Programme ‘WbMan’ und ‘MUI-Exchange’ und endlose Listen von Bug-Reports. Außerdem war er beim Korrektur-Lesen der Autodocs mit Abstand am fleißigsten, malte die Icons der Demo-Programme und schrieb ein paar BOOPSI Images.
- Wouter van Oortmerssen  
... für das Amige-E Interface.
- Matthias “tron” Scheler und Markus “corwin” Stipp  
... haben die erste richtige MUI Applikation geschrieben, einen Message-Editor für das Universal Mail System (UMS). Look out for ‘IntuiNews’! Außerdem stammt von Matthias das Beispiel-Programm ‘Font’.
- Andreas “goonie” Schildbach  
... hat MUI-Design und -Funktionsumfang entscheidend beeinflußt, von ihm stammt unter anderem ein kompletter Satz MUI-Images. Außerdem schreibt er ebenfalls mehrere Applikationen, eine davon ist eine Telefon- und Anrufbeantworter-Software für ISDN. Andreas hat mich durch die wie immer endlosen Telefongespräche auch mal auf andere Gedanken gebracht.
- Wolfgang Schildbach  
... für seinen Text-Formatierungs-Code.
- Christian “Kochtopf” Scholz  
... für das Modula-Interface.
- Ibrahim “radi” Solmaz  
... der mich ebenfalls durch viele Telefonate von der Arbeit abhielt, mir aber trotzdem manchmal eine wertvolle Hilfe war.
- Henri Veistera  
... für das Assembler-Interface. Ich hätte nie gedacht, daß man MUI in Assembler programmieren kann.

Das letzte Wort gilt jedoch allen registrierten Benutzern meines File-Requesters MFR. Der Erfolg, den ich mit diesem Programm hatte, bestärkte mich darin, es auch bei MUI auf Shareware-Basis zu versuchen. Deswegen tut es mir besonders leid, daß von MFR so lange kein Update mehr erschienen ist. Die Arbeit an MUI hatte für mich aber zunächst Priorität, insbesondere weil eine nächste Version von MFR vermutlich auf MUI aufbauen wird.

## 4.5 Diskussion

- "Was bringt MUI überhaupt für Nicht-Programmierer?"

In Zukunft werden hoffentlich viele Applikationen MUI zum Erstellen ihrer Userinterfaces verwenden. Als Benutzer solcher Applikationen hat man dann vielerlei Möglichkeiten auf deren Aussehen einzuwirken und kann alle nur erdenklichen Dinge an seinen persönlichen Geschmack anpassen. Um alle Features in vollem Umfang nutzen zu können, muß man sich für MUI registrieren lassen.

- "Warum ist MUI so langsam?"

MUI ist nicht langsam. Die große Vielfalt und Flexibilität erfordert jedoch einen deutlich höheren Rechenaufwand als bei den bisher bekannten Benutzeroberflächen. Insbesondere auf langsamen Rechnern kann sich dieser Aufwand unangenehm bemerkbar machen. Beim Starten der ersten MUI Applikation nach einem Reset müssen zudem noch einige Dateien nachgeladen werden, was ebenfalls einige Zeit in Anspruch nehmen kann. Diese Dateien bleiben dann allerdings im Speicher, so daß weitere Programme relativ schnell gestartet werden können.

Natürlich könnte man bei MUI einige Dinge beschleunigen und ich versuche auch, jede neue Version etwas schneller als die vorherigen zu machen. Allerdings bin ich nicht bereit, zu Gunsten von ein wenig Geschwindigkeitssteigerung auf Features wie etwa das hohe Maß an Konfigurierbarkeit zu verzichten. Wer heute noch einen 10 Jahre alten 68000er Prozessor mit 7 Megahertz Taktfrequenz verwendet, der muß damit rechnen, daß nicht alle Programme in Hochgeschwindigkeit ablaufen. Die Software entwickelt sich hin zu immer komplexeren Produkten, deren Anforderungen einfach nicht mehr durch veraltete Hardware erfüllt werden können.

- "Warum werden meine Fensterpositionen nicht gespeichert?"

Fensterpositionen werden normalerweise nur temporär, d.h. im RAM gespeichert und sind nach einem Neustart verloren. Eine dauerhafte Speicherung kann man erreichen, indem man nach Benutzen einer Applikation das MUI Preferences Programm startet und hier das 'Save'-Gadget angewählt.

- "Wozu dient das schräg schraffierte Feld beim Einstellen der Farb-Palette?"

Hier wird normalerweise die gerade eingestellte Farbe angezeigt, allerdings funktioniert dieses Feature erst ab Kickstart 3.0 und außerdem nur dann, wenn auf dem Bildschirm noch freie Farben zur Verfügung stehen.

- "Wie kann man eigene Images malen?"

Dazu kann jedes beliebige Malprogramm verwendet werden. Als Ausgangspunkt sollte man am besten das Bild 'MBrKit.ilbm' aus dem 'Images'-Verzeichnis dieser Distribution verwenden.

- "Warum unterstützen die String-Gadgets kein Clipboard?"

Es gibt ein Utility namens 'NewEdit', daß **allen** String Gadgets im System Clipboard-Fähigkeiten verleiht. Auch MUI's Gadgets arbeiten mit diesem Utility zusammen.

## 4.6 Disclaimer

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDER AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

## 4.7 License

Diese Lizenz betrifft die Weiterverbreitung des kompletten MUI-Systems. Sie hat mit dem Benutzen von MUI in eigenen Applikationen nichts zu tun, genaue Informationen darüber finden sich im Developer-Archiv.

- This license applies to the product called “MagicUserInterface” (short “MUI”), a collection of programs for the Amiga computer, published by Stefan Stuntz under the concepts of shareware, and the accompanying documentation. The terms “Program” and “MUI” below, refer to this product. The licensee is addressed as “you”.
- You may copy and distribute verbatim copies of the program’s executable code and documentation as you receive it, in any medium, provided that you conspicuously and appropriately publish only the original, unmodified program, with all copyright notices and disclaimers of warranty intact and including all the accompanying documentation, example files and anything else that came with the original.

- Except when otherwise stated in this documentation, you may not copy and/or distribute this program without the accompanying documentation and other additional files that came with the original. You may not copy and/or distribute modified versions of this program.
- You may not copy, modify, sublicense, distribute or transfer the program except as expressly provided under this license. Any attempt otherwise to copy, modify, sublicense, distribute or transfer the program is void, and will automatically terminate your rights to use the program under this license. However, parties who have received copies, or rights to use copies, from you under this license will not have their licenses terminated so long as such parties remain in full compliance.
- By copying, distributing and/or using the program you indicate your acceptance of this license to do so, and all its terms and conditions.
- Each time you redistribute the program, the recipient automatically receives a license from the original licensor to copy, distribute and/or use the program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein.
- You may not disassemble, decompile, re-source or otherwise reverse engineer the program.
- You agree to cease distributing the program and data involved if requested to do so by the author.

## 4.8 Installer

Mit MUI wird der Installer von Commodore ausgeliefert, der die Installation des Programmpakets auf Festplatte erheblich erleichtert. Für dieses Tool gelten die folgenden Bestimmungen:

```
Installer and Installer project icon
(c) Copyright 1991-93 Commodore-Amiga, Inc.  All Rights Reserved.
Reproduced and distributed under license from Commodore.
```

```
INSTALLER SOFTWARE IS PROVIDED "AS-IS" AND SUBJECT TO CHANGE;
NO WARRANTIES ARE MADE. ALL USE IS AT YOUR OWN RISK. NO LIABILITY
OR RESPONSIBILITY IS ASSUMED.
```