

About Faces

Zusammenfassung Chapter 25 - 32
von Christian Fritsche - www.d-sided.de

Chapter 25

Window Behaviors

PARC and the Alto

Parc Alto war der 1. Computer mit einem grafischem Interface. Außerdem besaß dieser: eine Maus, rechteckige Fenster, Scrollbars, Pushbuttons, Metapher Desktop, OOP, Drop-down Menues.
Aus dem Alto entwickelte sich der erste Mac.

PARC's Principles

Vermeidung von „Modes“

Ein Mode ist ein Programmzustand, der von der Norm abweicht und damit Irritationen beim Anwender auslöst.

Überlappende Windows (Mac)

Inhalte in rechteckige Windows darzustellen hat die Vorteile: Display, Foto, Film, Papier ist auch rechteckig => die meisten Datenoutputs von Menschen sind in rechteckiger Form.

Überlappende Windows symbolisieren übereinanderliegende Papiere auf dem Schreibtisch.

Probleme:

- Bei 3 oder mehr offenen Fenstern wird es unübersichtlich
- Klickt man ein Pixel daneben, befindet man sich direkt in einem anderen offenen Programm

Lösung:

- Taskbar (Microsoft)

Tiled Windows (Microsoft)

anstatt Fenster überlappend darzustellen, ging Microsoft einen anderen Weg, in dem die Fenster nebeneinander dargestellt wurden.

Problem: enormer Platzverbrauch. Wir heutzutage nicht mehr eingesetzt.

Full-Screen Applications

bei überlappenden Fenstern ist es schwierig zu navigieren. Linux entwickelte deshalb virtuelle Desktops. Windows entwickelte die Taskbar, die sehr pixelfreundlich ist und es erlaubt zwischen mehreren Fullscreenanwendungen zu wechseln. Mac beharrt bis heute auf überlappende Fenster und ermöglicht keine Fullscreenanwendungen.

Multipaned Applications

mehrere Fenster gehören zu einer Anwendung, die durch bewegbare Splitter getrennt sind (Outlook).
Adoptiert auch im Web mittels Frames.

Choosing your Window

Ein Programm sollte unbedingt nur so viele Fenster haben, wie unbedingt notwendig.

Vergleich: Programm=Haus, Fenster=Raum:

A dialog box is another room; have a good reason to go there!

Es sollte nicht aus Einfachheitsgründen jede Funktion sein eigenes Fenster haben, weil es aus Programmierer Sicht logisch ist. Immer das Programm aus der Sicht des Benutzers betrachten.

Build functions into the window where they are used!

Necessary rooms

Programmfunktionen, die außerhalb des üblichen Workflows liegen, sollten auch gesondert dargestellt werden. Man unterscheidet zwischen Funktionen und Dialog Boxen. Funktionen sind Werkzeuge, die essentiell sind (z.B. Pinsel in Photoshop). Dialog Boxen sind speziell und werden nicht immer gebraucht. Sie sollten schnell zugänglich sein; denn jede Dialogbox bringt ihre eigenen Funktionen mit sich. Funktionen, wie Speichern sollten hingegen in keine eigene Dialogbox; denn wenn der User speichern will ändert sich sein Hauptziel. Bsp. in Photoshop ist erst das Ziel eine Grafik zu speichern, dann ist ist das Ziel dieses zu speichern. Angesprochene Dialog Boxen dienen dazu das erstere Ziel zu erreichen.

Windows Pollution

Nie für jede einzelne Funktion ein eigenes Fenster öffnen, dies führt zum Chaos! (s. AOL Navigator)

The utility of any interaction idiom is context-dependent!

Mit Visual Basic ist es einfach neue Fenster zu erstellen. Aber das es leicht ist etwas zu tun bedeutet nicht, dass es auch gutes Design ist!

Window States

Die Fenster von Microsoft Window haben 3 Statusse: Minimized, Maximized, Pluralized.

- Minimized: Fenster verkleinern, ausblenden (bei Windows in die Taskbar)
- Maximized: Fenster wird auf den ganzen Bildschirm gestreckt.
- Pluralized: Fenster wird "restored", d.h. in den vorherigen Status zurückgesetzt.

Why minimize?

Bei Systemen ohne Taskbar führt das zu großen Problemen; denn wie stellt man das Fenster wieder her? Tastenkombination wie "Alt+Tab" sind eher unbekannt. Besonders ist das die einzigste Funktion bei Windows, bei dem ein mehrmaliges drücken der gleichen Tasten etwas bewirkt. Nicht konform.

Seit der Taskbar ist diese Funktion aber allen zugänglich.

Vorteil: Der aktuelles Status des Programms bleibt erhalten. Es kehrt in seinen früheren Status zurück.

Why pluralize?

- Drag and Drop zwischen 2 Programmen ist möglich.
- Cut and Paste über die Zwischenablage zwischen Programmen ist übersichtlicher.
- Heutzutage sind die Bildschirmauflösungen so hoch, dass man auf eine Vergrößerung auf Full-Screen nicht unbedingt angewiesen ist

MDI versus SDI

- MDI = multiple document interface
mehrere Dokumente können innerhalb einer Anwendung gleichzeitig geöffnet werden. Dies wurde als Wundermittel gegen viele Probleme angesehen. Auch weil es wenig Ressourcen benötigt. Über das Windowsmüenue kann zwischen den Dokumenten hin und her gewechselt werden. Heutzutage wird es von Microsoft nicht mehr weiter verwendet. Sondern es wird wieder SDI genommen.
- SDI= single document interface
Eine Anwendung kann lediglich ein Dokument geöffnet haben. Wenn man mehrere Dokumente gleichzeitig benötigt, muss jedesmal das Programm neu gestartet werden. Das verbraucht mehr Ressourcen, ist aber durch die heutige Hardware kein Problem mehr. Dieser Weg ist übersichtlicher, da jedes Dokument in der Taskleiste zu sehen ist und schnell zwischen Dokumenten gewechselt werden kann. Heutzutage wird beides miteinander vereint. Der Weg über die Taskleiste ist genauso möglich, wie über den Windowmenuepunkt.

Chapter 26

Using Controls

Controls lassen sich in folgende Kategorien einteilen:

- Imperative - initialisiert Funktionen
- Selection - selektiert Optionen, Daten
- Entry - Dateneingabe
- Display - direkte Manipulation der Programmvisualisierung

Avoiding Control-Laden Dialog Boxes

Es ist sehr einfach Dialog Boxes zu programmierung. Aber:

A multitude of control-laden dialog boxes doth not a good user interface make!

Imperative Controls

Imperative Controls rufen sofortige Aktionen hervor!

Push-Buttons

Hat meistens einen 3D Look, so wird die Aufforderung eines Klickens deutlich erhöht. Beim Klicken wird abermals die Grafik geändert. Auch ändert sich der Mauszeiger. Gutes Design und oft im Web kopiert!

Cursor hinting isn't enough!

Butcons

Es ist ein Push-Button innerhalb der Toolbar ohne Text, rechteckig mit symbolischer Bedeutung. Sie können außerdem mit Tooltips versehen sein. Sie existieren seit Windows 3.0 und sind heute standard.

Vorteile: (in der Theorie):

- leicht zu handhaben
- immer sichtbar
- schnell (im vgl. zu einer dropdownbox)
- leicht zu merken
- gleich in verwandten Programmen

Nachteile:

- Icon oft nicht eindeutig (da sie Verben veranschaulichen und nicht Nomen)
Lösung: Microsoft entwickelte Tooltips

Selection Controls

Hiermit ist keine Action verbunden. Es kann eine einfache Wahl (Ja/Nein) oder eine Gruppe von Auswahlen dargestellt werden.

Checkboxes

- Checkboxes waren eine der ersten visuellen Kontrollen, die entwickelt wurden
- Sie haben eine starke Klickaufforderung
- Sehr einfach und elegant
- Beschreibungstext bei Checkbox ist zwingend
- Checkboxes sind immer quadratisch. Diese Form sollte NIE geändert werden.
- Die Checkbox kommt in Toolbars in der Form von Butcons zum Einsatz (bold, italic, unterstrichen)

Flip-flop buttons: A selection idiom to avoid

Beschriftung zeigt immer den nächsten Zustand an, so weiß man nicht in welchem Zustand man sich zurzeit befindet. Sie sollten unbedingt vermieden werden!

Radio buttons

- Die Auswahl mit Radiobuttons ist immer exklusiv. D.h. es ist immer nur eine Auswahl einer Gruppe möglich.
- eine Gruppe besteht aus minimal 2 Punkten
- ein Punkt ist immer ausgewählt
- sie verbrauchen mehr Platz als Checkboxes aber sie zeigen dem User alle Möglichkeiten auf einmal (sie haben eine lehrende Rolle)
- Radiobuttons sind immer rund, dass sollte NIE geändert werden!
- Radiobuttons kommen auch in Toolbars in Form von Butcons zum Einsatz (links-, rechtsbündig, zentriert)

Combutcons

Ein Combutcon ist Dropdown-Version einer Radiobuttonauswahl.

Combutcons kommen auch in Toolbars in Form von Butcons zum Einsatz (Rand in Excel)

Varianten von Combutcons werden in Adobeprodukten verwendet. Hier haben Butcons kleine Dreiecke im Butcon. Dieses Dreieck zeigt nach rechts. D.h. bei längerem Drücken öffnet sich eine Auswahl.

Bei Microsoftprodukten wird auch ein Dreieck verwendet, das nach unten zeigt z.B. bei der Farbwahl.

List controls

List controls sind Listen, die gescrollt werden können, von denen mehrere oder alle Punkte ausgewählt werden können. Dazu muss CTRL oder SHIFT gedrückt werden. Das ist aber nur erfahrenen Usern bekannt. Seit Windows 95 können die einzelnen Punkte aber auch mit Icons oder Checkboxes versehen werden!

Distinguish important text items in lists with graphic icons

Die einzelnen Punkte können in manchen Programmen auch per Drag and Drop verschoben werden. Diese Möglichkeit sollte man immer in Betracht ziehen. Viele Programme würden hiervon profitieren. Gut werden sie bsp. in Microsoft Outlook im Email-Versender-Menue eingesetzt.

Sie sind nicht statisch. Die Anzahl der einzelnen Punkten kann abhängig von der Usereingabe/Usersystem variieren (z.B. Schriften)

Ordering Lists

Viele Programme ermöglichen es Listen zu sortieren (z.B. nach Datum, Name, Extension) oder durch Drag and Drop die Reihenfolge zu verändern. Aber warum ist es nicht möglich z.B. nach Wichtigkeit zu sortieren?

Horizontal Scrolling

Text horizontal zu Scrollen schrecklich und sollte nie verwendet werden

Never scroll text horizontally!

Comboboxes

Eine Combobox ist eine Kombination von einer Listbox und einem Entryfield (nur Windows). Es ist ein Pop-Up List control.

Vorteile:

- platzsparend
- dynamisch
- Platz für viele Informationen
- drag and drop wird unterstützt

Nachteile:

- unterstützt keine Mehrfachauswahl

Entry Controls

Ermöglichen es dem User Informationen selbst einzugeben, anstatt sie nur auszuwählen.

Bounded and unbounded entry controls

bounded entry: Daten sind bsp. in einem bestimmten Bereich eingeschränkt (Slider mit Zahlen von 1-200). Diese Einschränkung sollte aber wohlüberlegt sein und nicht willkürlich. Eine Monatseingabe sollte die Zahlen von 1-12 aufzeigen und nicht von -1000000 - +1000000!

unbounded entry: Eingabe ist nicht beschränkt. Das kann schnell zu falschen Eingaben führen. Eingabemöglichkeiten, die im nachhinein auf Richtigkeit prüfen sind ebenfalls unbounded.

spinners

spinners werden für die Eingabe von numerischen Zahlwerten verwendet. Hierzu kann die Maus aber auch die Tastatur verwendet werden. Es besteht aus 2 Dreiecken auf der rechten Seite, wovon eins nach oben und eins nach unten zeigt, sowie einem Entryfield auf der linken Seite. Somit verschwimmt hier die Unterscheidung zwischen bounded und unbounded.

Unbounded entry: Text edit controls

Das Textentry Control ist die häufigste uneingeschränkte Eingabemöglichkeit. Zur Eingabe von numerischen Werten sollte diese allerdings nicht verwendet werden.

Validation

Unbounded Controls sollten nachher immer geprüft werden. Bei häufigen Fehleingaben führt dies aber schnell zu Frustration bei dem User. Dabei muss das Feld mit der falschen Eingabe deutlich hervorgehoben werden.

Use bounded controls for bounded input!

Active and Passive Validation

Active Validation: Fehlerkorrektur erfolgt in Echtzeit. So werden Spaces, Tabs oder Zahlen sofort Korrigiert.

Passive Validation: Fehlerkorrektur erfolgt erst nach vollständiger Eingabe.

In beiden Fällen muss ein visuelles Feedback erfolgen!

Clue Boxes

Eine Clue Box ist wie ein Tooltip das erscheint, wenn eine Fehleingabe erfolgt ist.

Handling out of bounds data

Bsp. wenn in einem Entryfield eine Zahl eingegeben wird und man gibt statt der Zahl "9" das Wort "Neun" ein, sollte nicht die Fehlermeldung kommen: "Bitte geben Sie eine Zahl ein!"; denn das wurde ja erfüllt. Eine andere Möglichkeit wäre es auch, das Wort zu erkennen und es dann in eine Zahl zu konvertieren.

Units and measurements

Sollte die Eingabe von Einheiten erforderlich sein, sollten verschiedene Varianten erlaubt sein.

Muss bsp. ein Wert in Inches eingegeben werden, sollte folgendes möglich sein: 5i, 5in, 5 inches.

Using Text Edit Controls for Output: Bad Idea

Textentry Control fordert zur Dateneingabe auf. Dieses sollte nie zum Datenoutput verwendet werden.

Use non-editable (display) controls for output-only text!

Display Controls

Display Controls werden benutzt, um visuell Informationen zu präsentieren.

Text controls

Die wohl einfachste Ausgabemöglichkeit erlauben Textfelder.

Das Problem ist dabei, dass sie meistens dort eingesetzt werden, wo editierbare Textfelder besser ange-

bracht gewesen wären. Warum sollte man nicht dort, wo man die Information sieht, sie auch ändern dürfen?

Those darned scrollbars

Scrollbars verbrauchen sehr viele Pixel und werden unnötig oft benutzt. Oft, weil Designer/Programmierer keine bessere Idee haben.

Scrollbars geben über den Slider ein visuelles Feedback über die Länge und die aktuelle Position des Textes.

Aber sie sollten eigentlich noch viel mehr können:

- Anzeige: Wieviele Seiten hat der Text
- aktuelle Seitenzahl beim Scrollen
- Ersten Satz oder Bild einer Seite beim Scrollen
- Möglichkeit geben zum springen zu: Seiten/Kapitel/Abschnitte/Keywords
- Möglichkeit geben um ans Ende und an den Anfang zu springen
- Möglichkeit geben um Bookmarks zu setzen

Probleme bei Scrollbars:

Die Maus muss genau platziert werden, d.h. Konzentration wird vom eigentlichen Ziel abgelenkt, da die zwei äußersten rechten Pixel nicht Teil der Scrollbar sind.

Sliders and dials

Sie sind eine bounded und nicht proportionale Kontrollmöglichkeit. Oft ist eine Scrollbar sinnvoller. Sie sind sehr klein und platzsparend, erfordern aber präzisen Umgang mit der Maus.

Thumbwheels

Thumbwheels ist eine Variante von dials, nur leichter im Umgang. Einsatz finden sie oft in 3D Programmen.

Splitters

Splitters eignen sich gut um eine Anwendung aufzuteilen. Jeder Teil ist wenn gewollt scrollbar. Manchmal sind Drawer (Tabs) eine bessere Wahl.

Drawers and levers

Drawers (Tabs) sind Panes, die mit einem Klick geöffnet oder geschlossen werden können. Sie eignen sich gut für Funktionen und Controls, die weniger oft benutzt werden. Sie sollten aber nie übermäßig verwendet werden und sinnvoll eingesetzt werden. Von einem Mehrreihigen Einsatz wird dringend abgeraten.

Chapter 27

Menus: The Pedagogic Vector

The Command-Line interface

vor 1984 (erster Mac) gab es noch kein grafisches User Interface. Daten wurden über eine Command Line eingegeben. Benutzer waren in erster Linie Programmierer.

Die Command Line ist schneller als eine Eingabe per Maus. Aber man musste sich alle Befehle merken.

Sequential Hierarchical Menus

Ist ein Menue, dass auf dem Display dargestellt wird. Per Zahlen lassen sich die Optionen auswählen. So braucht man sich nicht alles merken oder man Infos zu einzelnen Befehlen nachschauen (Help).

Hat man einen Punkt ausgewählt kann ein neues Menue das alte ersetzen ... Es ist in einer Baumstruktur aufgebaut. Man konnte allerdings nicht zurück und man hatte sich schnell verewählt. Daher wurde oft eine

Sicherheitsabfrage eingebaut die jedesmal nachfragt, ob man sich sicher sei.

The Lotus 1-2-3 Interface

Der nächste Entwicklungsschritt ging von Lotus aus mittels eines tiefverzweigten hierarchischen Menues. Hier konnte das Menue und die Anwendung gleichzeitig dargestellt werden.

Das Menue umfasste mehrere hundert Auswahlmöglichkeiten. Diese waren auch durch Tastaturkürzel erreichbar. Poweruser merkten sich nur noch Tastaturkürzel um Menüpunkte auszuwählen (z.B. /-c-g-x).

Drop-Down and Pop-Up Menus

Ein Pop-Up Fenster ist rechteckig und überdeckt das zurzeit aktive Fenster bis die Operation abgeschlossen ist. Diese Technik liegt den Drop-Down Menues zugrunde. Dabei kann das Drop-Down-Menue auch hierarchisch aufgebaut sein und solange aktiv bis nur noch eine Auswahlmöglichkeit besteht. Wichtig: Es kann immer nur ein Punkt ausgewählt werden.

Dialogboxen erlaubten es folgend, Menue Punkte aus den Drop-Down Menues auszusondern und zu Gruppen zusammen zufassen. Jede Gruppe wurde also nur noch durch ein Menüpunkt repräsentiert.

Menus Today: the Pedagogic Vector

Die moderne GUI brachte fundamentale Änderungen: direkte Manipulation und Toolbars (1989).

Die Butcons in der Toolbar repräsentieren die am häufigsten verwendeten Funktionen. Diese sind in erster Linie an erfahrene User gerichtet, die die Funktionen kennen. Anfänger sollten sich in erster Linie an das Drop-Down-Menue richten.

So entwickelten sich Menues und Dialogboxen immer mehr zu lehrende Bestandteile als für den täglichen Einsatz. Unterstützend wirkt dabei, das jede Dialogbox ein Cancelbutton hat und so zum Experimentieren anregt, da man alles rückgängig machen kann.

Provide a pedagogic vector with menus and dialogs

Ein Anfänger, der ein Programm zum ersten mal startet sollte sich daher alle Funktion im Menue und die Dialogboxen genauer ansehen.

Die Toolbar mit den Butcons ist der falsche Weg, da sie an erfahrene User gerichtet sind und keine Beschreibung mit sich führt.

Das Menue erlaubt es hingegen erfahrene User vergessene Funktionen wieder zu entdecken. Somit erfüllt das Menue folgende Aspekte:

- lehrt was möglich ist
- wo finde ich welche Funktion
- was für ein Shortcut hat die Funktion
- das Menue muss alle Funktionen darstellen. Was hier nicht steht kann das Programm nicht

Daher müssen Menüpunkte präzise formuliert sein. Hier werden noch viele Fehler gemacht. So sollte es nicht "Open" heißen, sondern "Open Document", statt "arrange" besser "arrange icons".

Weitere Informationen zu Menüpunkten können außerdem in der Statusbar dargestellt werden.

Chapter 28

Using Menus

Menus sind die vielleicht ältesten GUI Elemente. Diese akzeptieren wir ohne nachzufragen, weil es so viele Programm benutzen. Aber ist das so auch richtig?

Standard Menus

File und Edit ist immer links angeordnet und Help immer rechts. Das hat sich so entwickelt ist aber falsch!
File: Aufgrund des Filesystems auf der Festplatte

Edit: An Anlehnung eines Clipboard

Help: Hier findet ein hilfeschuchender User wohl am wenigsten Hilfe

Alles ist nach Funktionen benannt. Besser ist eine Benennung nach Zielen. Man muss das mentale Model des Users betrachten. Was denkt er? Was will er machen?

Richtet sich die Anwendung an Kinder oder Computerexperten?

The File Menu

Dieses sollte nicht auf Files ausgerichtet sein, sondern auf das Dokument und so entsprechend auch heißen.

The Edit Menu

Dies sollte nicht als Container für alle Funktionen gesehen werden, die man sonst nicht unterbekommt. Diese sollten dann in ein Optionen oder Preferences Menue.

The Windows Menu

Hier sollten nur Funktionen betreffend des Fensters dargestellt werden, wie Anordnung, Viewing, Switching.

The Help Menu

Dieses muss wesentlich umfangreicher gestaltet werden als bisher, z.B. fehlt in der Regel eine Auflistung der Shortcuts.

Optional Menus

The View Menu

Hier sollte dargestellt werden, wie der User die Daten sieht und optionale visuelle Hilfen, wie Lineale, Vorlagen oder Hilfslinien.

The Insert Menu

Ist lediglich eine Erweiterung des Editmenues. Nur sinnvoll wenn man mehrere Funktionen noch hat.

The Setting Menu

Hier müssen alle Einstellungen untergebracht sein. So auch die Druckereinstellungen (sonst unter File).

The Format Menu

ist optional, da hier exklusive Eigenschaften von visuellen Objekten eingestellt werden können und nicht Funktionen. Hier müsste z.B. das Page Setup eingestellt werden können (sonst unter File).

The Tool Menu

Hier sollten sehr mächtige Funktionen unter gebracht sein, die nur Poweruser anwenden.

Problematic Menu Idioms

Cascading Menus

Ein Menue ist kaskadierend, wenn innerhalb eines Drop-down-Menues ein Menüepunkt keine Funktion ist, sondern eine weitere Menueebene mit neuen Funktion bzw. weitere Menueebenen enthält.

Diese Technik scheint für Programmierer bei komplexen Programmen unumgänglich.

Dieses bringt viele Nachteile mit sich:

- keine klare Gruppierung
- schwer zu merken
- exakte Mausnavigation von Nöten

Hierarchische Menues, die bereits in ihrem Grab lagen, werden somit wieder belebt. Diese sollten nur verwendet werden, wenn sonst alle Mittel ausgeschöpft sind. Funktionen, die oft gebraucht werden, sollten NIE in ein kaskadierendes Menue!

Microsoft macht von dieser hierarchischen Menuestruktur starken Gebrauch im Startmenue nur um Doppelclicks zu verhindern. Ist das sinnvoll?

Expanding Menus

Werden seit Windows 2000 von Microsoft eingesetzt. Es sind Menues, die nur die häufigsten Funktionen anzeigen. Alle anderen erst nach Klick auf einen Pfeil am Ende des Menues. Dieses soll laut MS pädagogische Gründe haben. Dieses ist es aber aus folgenden Gründen nicht:

- Menüanordnung ändert sich
- verwendet man einen seltenen Menüpunkt, wird er danach lange eingeblendet
- man versteckt den Programmumfang
- man muss erneut Klicken

Gut ist, dass man die Funktion abstellen kann. Leider ist sie defaultmäßig an. Microsoft muss lernen, dass der Platz für häufig verwendete Funktionen die Toolbar ist.

Bang Menus

Bang Menues gibt es nicht mehr. Es sind Funktion direkt auf der obersten Ebene, ohne dass sich ein Menü ausklappt (z.B. Compile).

Menu Item Conventions and Variants

Disabling menu items

Funktionen, die momentan nicht möglich sind, werden eingegraut und somit als solches gekennzeichnet.

Disable menu items when they are inapplicable!

Checkmark menu items

Checkmarks schalten Funktion an oder aus und sind im Menü durch Haken symbolisiert. Bsp. können Lineale an oder aus sein. Sie sollten nur in einfach Programmen eingesetzt werden. In komplexen ist der Platz zu wertvoll. Da sollten solche Funktionen innerhalb einer Dialog Box mit weiteren Funktionen dargestellt werden.

Flip-flop menu items

Der Funktionstitel ändert sich je nach Funktion. Bsp. Hide Statusbar - Show Statusbar. Dieses spart zwar Platz ist aber sehr verwirrend und sollte NIE verwendet werden. Besser Checkmarks verwenden.

Icons on menus

Icons im Menü sind sehr sinnvoll. Man erkennt die Funktion schneller. Evtl. allein anhand des Icons. Außerdem ist der Lerneffekt größer, wenn das selbe Icon z.B. auch in der Toolbar als Butcon verwendet wird.

Use parallel visual symbols on parallel command vectors!

Accelerators (Shortcuts)

Tastaturkürzel um Funktionen aufzurufen. Es sollte beachtet werden:

- Standards beachten
- täglichen Einsatz beachten
- Erreichbarkeit zeigen
- Möglichkeit bieten, eigene Shortcuts zu definieren

Mnemonics

Mnemonics sind Shortcuts mit denen man das Menü öffnet und dort eine Funktion mittels des unterstrichenen Buchstabens aufruft (bsp. ALT+O). Sie werden immer durch ALT initialisiert.

Mnemonics sind nicht optional. Sie müssen vorhanden sein!

The Windows System Menu

Ist das Icon in der oberen linken Ecke eines Windowsfensters. Es ist ein Relikt aus Windows 3.1. Es könnte ohne weiteres eliminiert werden.

Chapter 29

Using Toolbars and Tooltips

Toolbars: Visible, Immediate Functionality

Eine Toolbar ist:

- eine Collection von Butcons.
- immer sichtbar
- einzeilig
- führt häufig verwendete Funktionen auf
- sind schnell
- nicht geeignet für Anfänger
- nicht vollständig

Toolbars provide experienced users fast access to frequently used functions!

Toolbars and Toolbar Controls

Icons versus text on toolbars

Text beschreibt eine Funktion genau und präzise. Text hat eine lehrende Rolle und gehört ins Menue. Viele Programme (bsp. IE) beschreiben die Butcons zusätzlich mit Text. Das ist falsch!

Bei Butcons wird vorausgesetzt, dass die Funktion bekannt ist. Es muss nicht lehrend sein aber eindeutig. Außerdem wird unnötig viel Platz verschwendet.

Explaining Toolbar Controls

Das größte Problem von Toolbars ist, wie man sie den Anfänger am besten erklärt.

Balloon help: A first attempt

Balloon help sind nur nervig und mittlerweile nicht mehr im Einsatz. Es ist eine große Sprechblase, die die Funktion genau erklärt. Das mag für den Anfänger nett sein, für den täglichen Einsatz aber völlig ungeeignet.

ToolTips

ToolTips sind clever und effektiv. Sie erklären nur den Zweck einer Funktion. Bestehen aus einem einzeiligen Satz oder Wort und erscheinen nach einer kurzen Verzögerung. Sie brachten den Durchbruch für die Toolbar. Erklärungen in der Statusbar sind bei weitem nicht so gut, wie Tooltips. Erfahrene User sollten dennoch die Möglichkeit haben, sie ausstellen zu können.

Use ToolTips with all toolbar and iconic controls!

Evolution of the Toolbar

Als Designer die Möglichkeiten der Toolbar erkannten, wurden nicht mehr nur Butcons verwendet, sondern sie wurde vielfältig erweitert. Zuerst wurde die Combox (bsp. Schriftart) integriert. Nachher folgten zuvor beschriebene Idiome.

Menus on toolbars

Mittlerweise befinden sich Menues in der Toolbar, was schon sehr ironisch erscheint (bsp. Undo). Ob dieses logisch ist, ist noch offen. Allerdings rückt somit das Standardmenue immer weiter in den Hintergrund und die Toolbar entwickelt sich zur primären Steuerung.

Moveable toolbars

Keiner fördert die Toolbar mehr als Microsoft. Im aktuellen Office lassen sich die Toolbars vollständig individualisieren. Es kann die Sichtbarkeit und die Position eingestellt werden. Außerdem können sie ange-dockt und attached werden. Auch kann man Toolbars als "floating Toolbar" darstellen. Steht unzureichend Platz für eine Toolbar zu Verfügung, erscheint ein Erweiter-Button in der Toolbar (>>).

Customizable toolbars

Durch viele möglichen Einstellungen der Toolbars, besteht die Möglichkeit, dass unerfahrene User sich ihre Oberfläche selbst unbedienbar machen. Dieses muss aber in Kauf genommen werden. Die Toolbars müssen sich immer zum Defaultlayout zurückstellen lassen.

The Windows Taskbar: Special Purpose Toolbars

Die Windows Taskbar existiert seit Windows 95. Sie ist gut geeignet für Multitasking, hat ein konfigurierbares Startmenue, eine Quickstartleiste, den System Tray und lässt sich frei an den Rändern positionieren. Die Dockbar auf dem Mac ist bei weitem nicht so effizient, wie die Taskbar. Der schwerwiegendste Punkt ist, dass die Dockbar keinen Unterschied zwischen aufrufenden Programmen und aktuell laufenden Prozessen macht.

The Start menue

Das Startmenue ist der Eintrittspunkt in die Programme über ein kaskadierendes Menue. Dieses ist immer auf dem Screen zu sehen.

Seit Windows XP ist das Startmenue ausführlicher geworden. Leider führen dort die Kategorien eher zu Verwirrungen als zum besseren Verständnis.

Quick Start toolbar

ist ein Gebiet, das frei anpassbar ist. Dort dienen Butcons zum Aufruf von Programmen. Viele Programme installieren dort automatisch ihre eigenen Butcons. Dies sollte nie ohne Aufforderung geschehen.

Window buttons

Diese präsentieren alle laufenden Anwendungen. Durch Klick werden sie minimiert oder wieder hergestellt. Dieses funktioniert aber nur bis zu einer gewissen Anzahl an laufenden Anwendungen.

Seit XP kann man gleiche Anwendungen gruppieren. Dieses hat aber zwei große Probleme:

- es lässt sich nicht sagen, welche Anwendungen offen sind
- die Abbildung zwischen des aktuellen Window Button und der aktuell geöffneten Anwendung ist nicht 1:1 sondern 1:n.

Dieses lässt sich Gott sei Dank ausstellen!

The Status Area (System Tray)

Gibt Informationen über Hardware und Hintergrundprozessen, sowie zu Zeit und Datum.

Chapter 30

Using Dialogs

Dialogboxen spielen eine sekundäre, unterstützende Rolle.

Dialogs Suspend Normal Interaction

Dialogboxen unterbrechen den normalen Arbeitsbetrieb und kehren nach ihren Beenden zum Punkt des Aufrufs zurück.

Put primary interactions in the primary window! Dialogs break flow!

Dialogboxen sind der Ort für spezielle Interaktionen. Solche, die im primären Fenster nur irritieren würden. So können bsp. spezielle Settings hier erklärt werden. Hierfür bieten Dialogboxen genügend Platz. Dialogboxen sprechen zugleich zwei Usergruppen an: Erfahrene Nutzer nutzen Dialogboxen wegen den erweiterten Funktionen. Und unerfahrene Nutzer lernen mit Hilfe der Dialogboxen das Programm kennen.

Eine Dialogbox muss sein:

- kompakt
- mächtig
- schnell
- klar strukturiert
- selbsterklärend

Dialog Box Basics

Dialogboxen müssen bestimmten Konventionen entsprechen. Sie haben einen Besitzer, i.d.R. das aufrufende Programm, es kann aber auch Windows selbst sein. Wichtig ist, dass jede Dialog Box mindestens einen Exit Button hat.

Modal dialog boxes

Es existieren zwei verschiedene Dialogboxen: modal oder modeless.

Modale Dialogboxen sind die üblichen:

- das aufrufende Programm stoppt und wird nach Bearbeitung der Box fortgesetzt. Deswegen werden sie auch application modal boxes genannt. Es gibt auch system modal boxes. Diese halten das ganze System an.
- alle anderen Programmfunktionen sind deaktiviert
- andere Programme können gestartet werden
- Dialogboxen sind für Designer und User leicht zu verstehen
- sie besitzen terminierende Symbole

Never create a system modal dialog!

Modeless dialog boxes

Modeless Boxen sind eher die Ausnahme, werden aber zurzeit von Adobe und Macromedia Produkten stark verwendet.

- das aktuelle Programm wird nicht gestoppt, d.h. sie bleiben nach Bearbeitung offen.
- sie besitzen terminierende Symbole
- schwer verständlich, weil Ziel oft unklar
- Sie verbrauchen viel Platz. Die Größe sollte gut überlegt sein.

Ein Beispiel stellt "Find and Replace" dar.

Modeless dialog issues

Modeless Dialog Boxen sind visuell den modalen Dialog Boxen sehr ähnlich, funktional aber sehr verschieden. Daher ist es für Benutzer oft schwierig sie als solches zu erkennen. Oft wird das Verhalten einer modalen Box erwartet.

Two solutions for better modeless dialogs

1. evolutionärer Ansatz:

- differenziere modeless und modale Dialogboxen visuell! Hier muss ein Standard gefunden werden!
- Konsistenz bei terminierenden Symbolen muss geschaffen werden. Hier gibt es zuviele unterschiedliche Begriffe für das Gleiche: Close, Apply, Done, Accept, Yes, OK, ... Außerdem sollte das terminierende Symbol in der Titelleiste immer vorhanden sein.
- Buttonlabels sollten sich nicht dynamisch ändern. Bsp. von Cancel zu Apply oder Close.
- Closebuttons sollten immer gleich positioniert sein und die gleichen Ausmaße haben.

2. radikaler Ansatz:

Toolbars mit Butcons sind nichts anderes sind also modeless dialog boxes. Der Unterschied ist nur der, dass sie visuell nichts mit Dialogboxen gemein haben. Auch können hier einzelne Butcons leicht eingegraut werden, wenn man diese zurzeit nicht benutzen kann. Dies ist alles von den Usern akzeptiert. Außerdem benötigen sie keinen Exitbutton und sind äußerst platzsparend, es entfällt bsp. der title bereich. Auch verdecken sie nicht den aktuellen Arbeitsbereich. Oft lassen sie sich, bei nicht Gebrauch, am Seitenrand andocken und somit wieder leicht reaktivieren. Sie würden alle Nachteile von modeless dialog boxes beseitigen.

Goal-Directed Dialog Boxes

Es existieren 4 verschiedenen Arten von Dialogboxen: property, function, process und bulletin.

Property dialog boxes

Hier werden Einstellungen und Eigenschaften angezeigt, verändert. I.d.R. für eine aktuelle Markierung.

Function dialog boxes

kontrollieren eine bestimmte einzelne Funktion (bsp. Drucken). Hier können Funktionsdetails eingestellt werden.

Process dialog boxes

diese werden vom Programm selbst gestartet (bsp. Ladebalken, Alerts).

Programs must inform the user when they are about to become unresponsive!

Jeder Process dialog muss folgendes erfüllen:

- zeige an, dass ein länger andauernder Prozess abläuft
- zeige an, wie lange der Prozess noch dauert
- zeige an, dass dieses normal ist
- eine Cancel Option muss vorhanden sein

bulletin dialog boxes

Hier werden Fehlermeldungen angezeigt.

Chapter 31

Dialog Etiquette

Wie müssen sich Dialogboxen richtig verhalten?

Politeness as a dialog Virtue

Property und Function Dialogboxen erscheinen, wenn der User sie aufruft. Process und Bulletin Dialogboxen erscheinen, wenn der Programmablauf es erzwingt. Daher müssen sie unterschiedlich präsentiert werden.

Erstere können daher mehr Platz verbrauchen, während letztere kleiner und kompakter dargestellt werden sollten.

The Title Bar

Jede Dialogbox sollte bewegbar sein. Dies ist nur mittels der Titlebar möglich.

All dialog boxes should have title bars!

Was soll in der Titlebar stehen: Funktionsname oder aufrufendes Programm?
Keines von beiden!

Use verbs in function dialog title bars!

Bezieht sich die Funktion auf eine Auswahl, sollte diese mit in den Titel aufgenommen werden.
Öffnet man eine Property Dialog Box, sollte im Title eine Beschreibung des Objektes stehen, für das Einstellung gesetzt werden (bsp. für einen Ordner Backup: "Backup Properties").

Use object names in property dialog title bars!

Transient Posture

Dialog Boxen kann man als kurzlebige eigenständige Programme innerhalb von Programmen bezeichnen. Trotzdem verbrauchen sie Platz von ihrem aufrufenden Programm.

Dialogs should be as small as possible, but not smaller!

Anordnung von Schaltflächen: OK immer rechts, OK und Cancel immer unten!

Reduce Excise

Know where you are needed

- Dialog boxes müssen sich ihre letzte Position merken und dort wieder geöffnet werden
- Der letzte Status muss abgebildet werden

Know if you are needed

Eine Dialog Box sollte nur dann aufrufbar sein, wenn ihre Funktion Sinn machen, ansonsten nicht.

Terminating Commands for Modal Dialogs

Modale Dialog Boxen schließen sich auf drei Wege:

- Cancel (Ablehnen)
- Ok (Zustimmen)
- CloseBox in der Titlebar (Ablehnen)

Diese dürfen in ihren Bezeichnungen nicht geändert werden!

Offer OK and Cancel buttons on all modal dialog boxes!

Benutze außerdem niemals diese Wörter in anderen Zusammenhängen!

Never use terminating command words in dialog text!

Folgende Eigenschaften sollten bei Ok und Cancel berücksichtigt werden:

- müssen schnell erkennbar sein
- sollte nicht mit anderen Schaltflächen zusammen stehen
- Ok ist wichtiger als Cancel besonders für erfahrene User
- Buttons sollten unten rechts platziert sein
- OK ganz rechts und Cancel links von OK

The Closebox

Closebox sind für Anfänger schwierig zu deuten. Wird OK oder Cancel ausgeführt?

Don't put close boxes on modal dialogs!

The Help button

Der Help Button sollte nicht bei terminierenden Buttons stehen, sondern besser mit in der Titlebar (mittels "?").

Keyboard Shortcuts

Shortcuts sollten programmübergreifend gleich sein. Sie sind besonders für Poweruser wichtig. Diese haben auch einen großen Einfluss auf neue User.

Tabbed Dialogs

Tabbed Dialogs erfreuten sich schnell großer Beliebtheit und sind heute standard. Diese sollten immer horizontal oben dargestellt werden und niemals vertikal seitlich.

Tabübergreifende Funktionen sollten visuell getrennt unterhalb des Tabinhalts sichtbar sein.

Put terminating buttons on the untabbed area of a tabbed dialog!

Breadth versus depth

Tabs erlauben es sehr viele Informationen in eine Dialogbox unterzubringen. Tabs sollten daher sinnvoll ausgewählt werden und zusammen gehören.

Stacked tabs: Dialog abuse

Stacked Tabs sind Tabs die in mindestens zwei horizontalen Reihen übereinander angeordnet sind. Dieses sollte immer vermieden und besser auf zwei Dialog Boxes verteilt werden.

All idioms have practical limits. Don't stack tabs!

Expanding Dialogs

Expanding Dialogs sind Dialogboxen, die einmal Basiseinstellungen darstellen und durch Klick mehr, tiefer gehende Funktionen zeigen. Also ein Beginner- und ein Advanced-Modus besitzen.

Sie sind mittlerweile weit verbreitet. Allerdings ist es oft, dass wenn man das Programm in den Beginnermodus schaltet, alle Dialogboxen erweitert sind. Das ist falsch! Außerdem sollten immer Buttons zum erweitern und zum verkleinern dargestellt werden. Für diesen einen Fall sind Flip-Flop Buttons sinnvoll.

Cascading Dialogs

Cascading Dialog Boxes sind Dialog-Boxes, die durch Dialog Boxes geöffnet werden. So können sich zahlreiche Boxes öffnen, die in der geöffneten Reihenfolge abgearbeitet werden müssen. Sie führen zum Chaos und sind nicht mehr weit verbreitet. Drei kaskadierte Dialog Boxes sollten für alle Zwecke reichen.

Dynamic Dialogs

Es sind Dialogboxen, die je nach Usereingabe unterschiedlich aussehen oder unterschiedliche Einstellungen haben. Dieses führt oft zu Verwirrungen!

Chapter 32

Creating Better Controls

Direct Manipulation Controls

Noch viel zu oft werden Entry Fields verwendet an dessen Stelle besser boundy entry controls stehen sollten. Click and Drag Controls sollten auch Entry Fields vorgezogen werden.

Extraction Controls

Text Edit Felder sind sehr stupide. Sie könne nur im nachhinein auf ihre Richtigkeit hin überprüft werden. Besser wären spezielle angepasste Eingabefelder sog. Extraction Controls. Bsp. sollten für Adresseingaben Adress Controls bestehen. Diese sollten ein einzigen Textfeld aufweisen, in das die Anschrift eingegeben werden kann. Die Eingabe wird später nach Straße, Ort, PLZ, ... automatisch geordnet.

Dieses würde in drei Schritten ablaufen:

- Text wird zuerst wortwörtlich wiedergegeben
- Jede Zeile wird separiert
- jede Zeile wird in die entsprechende Kategorie geparkt

So kann jeder, wie gewohnt seine Adresse eingeben. Sicherlich wird es eine Fehlerrate geben. Diese sollte aber gering ausfallen.

Teilweise wurde dieses Konzept in Outlook bei der Adresseingabe umgesetzt.

Visual Controls

Viele GUIs bestehen heutzutage noch aus Text, Radiobuttons, Lists, Comoboxes, ... Besser wäre es aber vielmehr mit Bildern zu arbeiten. Ein gutes Beispiel ist die Layer Darstellung in Photoshop. Hier werden die meisten Funktion visuell dargestellt.

Hiermit lassen sich komplexe Sachverhalte einfacher, schneller und direkter vermitteln und darstellen.

Die Gründe warum dieses heute noch nicht zum standard gehört ist einfach. Programmierer sind selten in der Lage ein gutes visuelles Interface zu erstellen. Designer, die das könnten, sind teuer. So wird oft das Geld gespart und auf die Windowsboardmittel zurückgegriffen. Nicht bedacht wird dabei oft, dass die Kosten für Designer später durch weniger Customer Support um ein vielfaches wieder wett gemacht werden.

Im Web wurden große Fortschritte mit immer neuen Techniken erzielt. Diese Techniken könnten auch bald für die GUI Erstellung zum Standard gehören. Macromedia Flash stellt dabei die interessanteste Technologie dar.