HOCHSCHULE HAMM-LIPPSTADT

Praktische Informatik

Vorlesung 01

Benutzeroberflächen und Frameworks

Inhalt

- Bedienkonzepte
- Qualität von Benutzerschnittstellen
- Grafische Benutzeroberflächen
- Frameworks

Bedienkonzepte von Programmen

- Alle unsere Programme besaßen bislang ein bestimmtes Bedienkonzept.
  - Mit Hilfe der Kommandozeile konnte das Programm mit dem Benutzer interagieren, z.B. Daten einlesen oder ausgeben.

- Neben diesem zeichenorientierten Bedienkonzept existieren viele weitere Möglichkeiten für eine **Mensch-Maschine-Interaktion**.
  - Spracheingabe
  - Bedienung über die Maus in einer grafischen Benutzeroberfläche
  - Web-basiert in einem Browser
  - App auf einem Smart-Phone

Qualität von Benutzerschnittstellen

- Es existieren viele verschiedene Arten, wie die Benutzerschnittstelle zwischen einem Programm und den menschlichen Nutzern gestaltet werden kann.
  - Alle Arten haben für bestimmte Bereiche ihre Vor- bzw. Nachteile.

- Wichtig: Die Nutzer der Anwendung müssen die Software **ergonomisch** bedienen können.
  - Der eigentliche Nutzen der Anwendung muss sich entfalten.

- Was aber bedeutet ergonomisch?

Software Ergonomie

- **Software Ergonomie** befasst sich damit, ob Computersysteme und Anwendungen benutzerfreundlich sind.

- Die DIN ISO 9241-11 definiert Anforderungen an die Gebrauchstauglichkeit:
  - Effektivität: Wie gut macht ein System, was es tun soll?
  - Effizienz: Wie schnell wird ein Kommando, dass vom Benutzer kommt ausgeführt?
  - Zufriedenheit: Wie fühlt sich ein System für einen Benutzer an?

- Daraus wurden verschiedene Kriterien für gute Dialoge in Computerprogrammen abgeleitet.

Kriterien für gute Computerdialoge

Aufgabenangemessenheit	•Geeignete Funktionalität, Minimierung unnötiger Interaktionen.
Selbstbeschreibungsfähigkeit	•Wie einfach kann man als neuer Benutzer erkennen, wie das System funktioniert?
Steuerbarkeit	•Kann der Benutzer den Ablauf des Dialogs beeinflussen?
Erwartungskonformität	•Verhält sich das Programm bei den selben Interaktionen immer gleich?
Fehlertoleranz	•Unerkannte Fehler verhindern nicht das Benutzerziel. •Erkannte Fehler sind leicht zu korrigieren.
Individualisierbarkeit	•Anpassbarkeit der Anwendung an den jeweiligen Benutzer und seine Fähigkeiten.
Lernförderlichkeit	•Minimierung der Erlernzeit durch z.B. Analogien und Metaphern. •Anleitung des Nutzers.

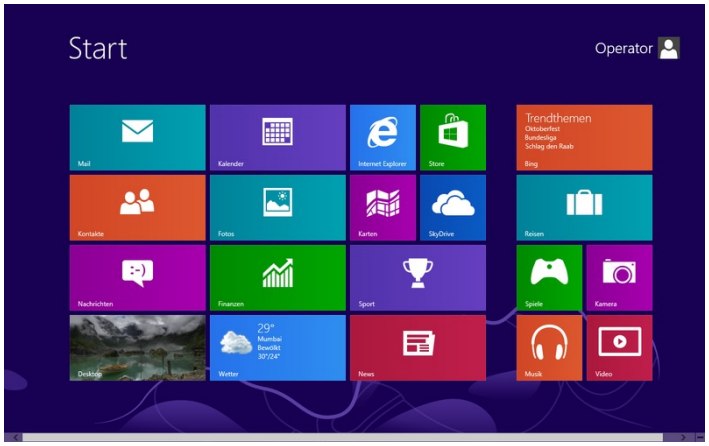
Grafische Benutzeroberflächen

- Eine **grafische Benutzeroberfläche** (*engl. Graphical User Interface, GUI*) ist eine Softwarekomponente (i.d.R. Teil eines Betriebssystems) und dient der Interaktion des Benutzers mit dem Rechner.
  - Mensch-Maschine-Interkation über grafische Symbole.
  - Maus als Eingabegerät.

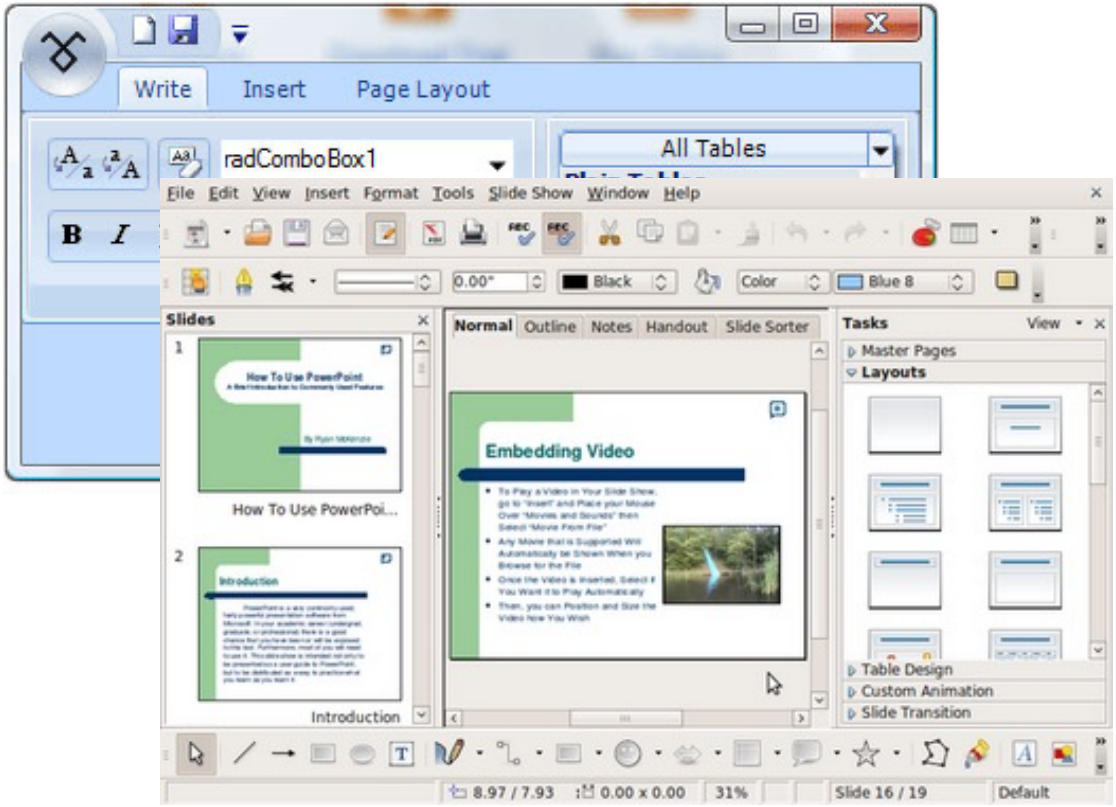
- Alle Betriebssysteme und Anwendungsprogramme, die im heute eine Rolle spielen, verfügen über eine grafische Benutzeroberfläche.

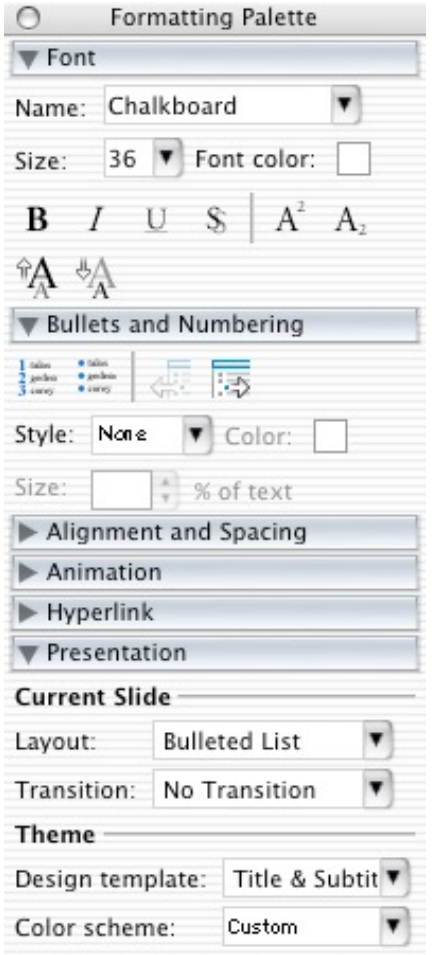
- Die meisten Anwendungen beachten dabei die Grundregeln für gute Dialoge in Programmen.
  - Als Programmierer ist es unsere Aufgebe, die Regeln einzuhalten.

Beispiele von grafischen Benutzeroberflächen



Beispiele von GUIs in Anwendungsprogrammen





Prinzipien moderner Computerdialoge

- GUIs sind deshalb auch für Gelegenheitsnutzer leicht bedienbar, weil sie Elemente benutzen, die man aus seinem Alltag bereits kennt.
  - Eine GUI bildet die Welt und bekannte Objekte in gewisser Weise nach, Analogie.

Desktop	Drag-and-drop	Objektorientierung
• Analogie zu einem echten Schreibtisch. • Ablage von Dokumenten auf Stapeln (Ordnern).	• Verschieben von Elementen mit dem Mauszeiger, der zu einer Hand wird. • Dateien löschen über drag-and-drop auf den Mülleimer. • Mac OS X: Programm deinstallieren via drag-and-drop.	• Benutzer wählt erst das zu manipulierende Objekt, dann die Funktion • Manipulationen sind spezifisch für die jeweiligen Objekte: Man kann Dateien zwar in Ordner schieben, aber nicht anders herum.

Computerdialoge, Formulare und Fenster

- Ein **Computerdialog** bzw. **Formular** dient der Unterstützung eines einzelnen Anwendungsfalls des Programms.
  - Ein Formular wird in einem **Fenster** der grafischen Benutzeroberfläche angezeigt.
  - Nur ein Fenster kann jeweils den sog. **Focus** besitzen und somit im Vordergrund stehen.
  - Ein Fenster kann auch **modal** angezeigt werden, d.h. so lange dieses Fenster geöffnet ist, kann der Focus nicht zu einem anderen Fenster wechseln.

- Die Bedienung geschieht primär über die Maus bzw. durch Berührung des Sensorbildschirms (Smartphones, Tablets, …).

- Ein Dialog ist über **grafische Elemente**, sog. **Steuerelemente**, GUI-Komponenten bzw. **Widgets** bedienbar.

Interaktionselemente eines Computerdialogs

- In einem Computerdialog werden **Steuerelemente** so angeordnet, dass eine ergonomische Oberfläche entsteht.

- Annähernd jede Oberfläche benutzt dabei die folgenden Steuerelemente:
  - Schaltflächen (engl. Buttons)
  - Text- bzw. Bezeichnungsfelder (auch mehrzeilig)
  - Auswahlkästchen (engl. Checkbox)
  - Optionsfelder (engl. Radiobuttons)
  - Bildlaufleisten (engl. Scrollbars)
  - Status- und Menüleisten
  - Listen- und Kombinationsfelder
  - Fortschrittsbalken, Registerkarten

Wann ist aber eine GUI gut?

- Einfache Antwort: **Der Anwender muss sie gut finden!**
- Sie muss die technischen Basisanforderungen erfüllen.
  - Kapazität, Durchsatz, Geschwindigkeit, Reaktionsfreude: Friert nicht ständig ein…
- Sie muss die benötigten Anwendungsfälle abdecken (Lastenheft).
- Sie muss Vertrauen erzeugen
  - Die Aufgabe kann effizient erledigt werden.
  - Man wird nicht allein gelassen.
  - Die Daten gehen nicht verloren.
  - Ohne Absicht kann nichts kaputt gehen.

Frameworks

- Eine GUI technisch zu realisieren kann sehr aufwändig sein.
  - Alle Interaktionselemente müssen auf dem Bildschirm gezeichnet werden.
  - Interaktion, wie z.B. Mausklicks, müssen abgefangen und korrekt verarbeitet werden.

- Daher nutzt man **sog. Rahmenwerke (engl. Frameworks)**, welche die Funktionalität bereits zur Verfügung stellen.
  - Ein Framework stellt dem Programmierer über ein **Appliction Programmable Interface (API)** Klassen/Funktionen bereit.

- Ein Framework ist mehr, als nur eine Bibliothek.
  - Ein Framework zwingt den Programmierer, Probleme auf bestimmte Weise zu lösen.

GUI Frameworks

- Es gibt eine große Auswahl von GUI Frameworks für unterschiedlichste Programmiersprachen:
  - C++ → Qt, Gtk, ...
  - ObjectiveC/SWIFT → CoCoA, ...
  - C# → WindowsForms, WPF, Xamarin, GTK#, MAUI, ...

- Wir werden uns für WPF entscheiden.
  - Mit WPF lassen sich unter Windows “normale“ Desktop-Anwendungen entwickeln.
  - Mit Hilfe von Xamarin lassen sich damit zudem Apps für iOS und Android entwickeln.
  - Wird aktuell durch MAUI abgelöst, ist aber noch nicht so weit (leider).

Wir haben heute gelernt…

- Dass unterschiedliche Bedienkonzepte für die Mensch-Maschine-Interaktion existieren.
- Welche Kriterien eine qualitativ hochwertige Benutzerschnittstelle ausmacht.
- Was eine grafische Benutzeroberfläche ist und woraus sie besteht.
- Wie uns Frameworks bei der Entwicklung einer grafischen Benutzeroberfläche helfen.