

# Einführung

## Themenüberblick



# Agenda

- 1. Historie und Entwicklung**
2. Typen grafischer Oberflächen
3. Standardprobleme GUI
4. Programmiermodelle für GUI



# Historie der Benutzerschnittstellen

1960      1970      1980      1990      2000      2010      2020



# Lochstreifen (1960)



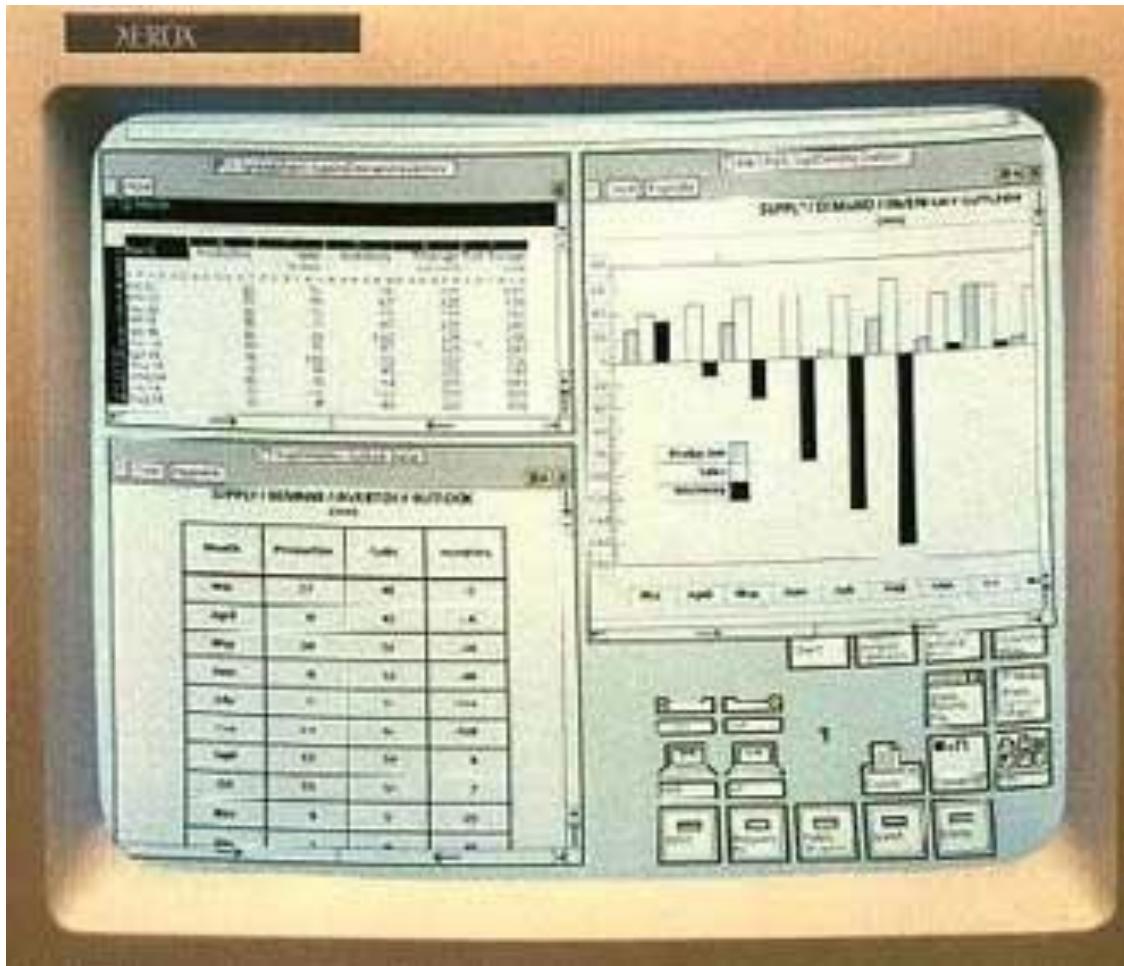
# IBM 3270 (1971)



- Die Mutter aller betrieblichen Informationssysteme
- Tastaturbedienung
- Textoberfläche mit grafischen Elementen (2 Helligkeitsstufen, blinken ...)



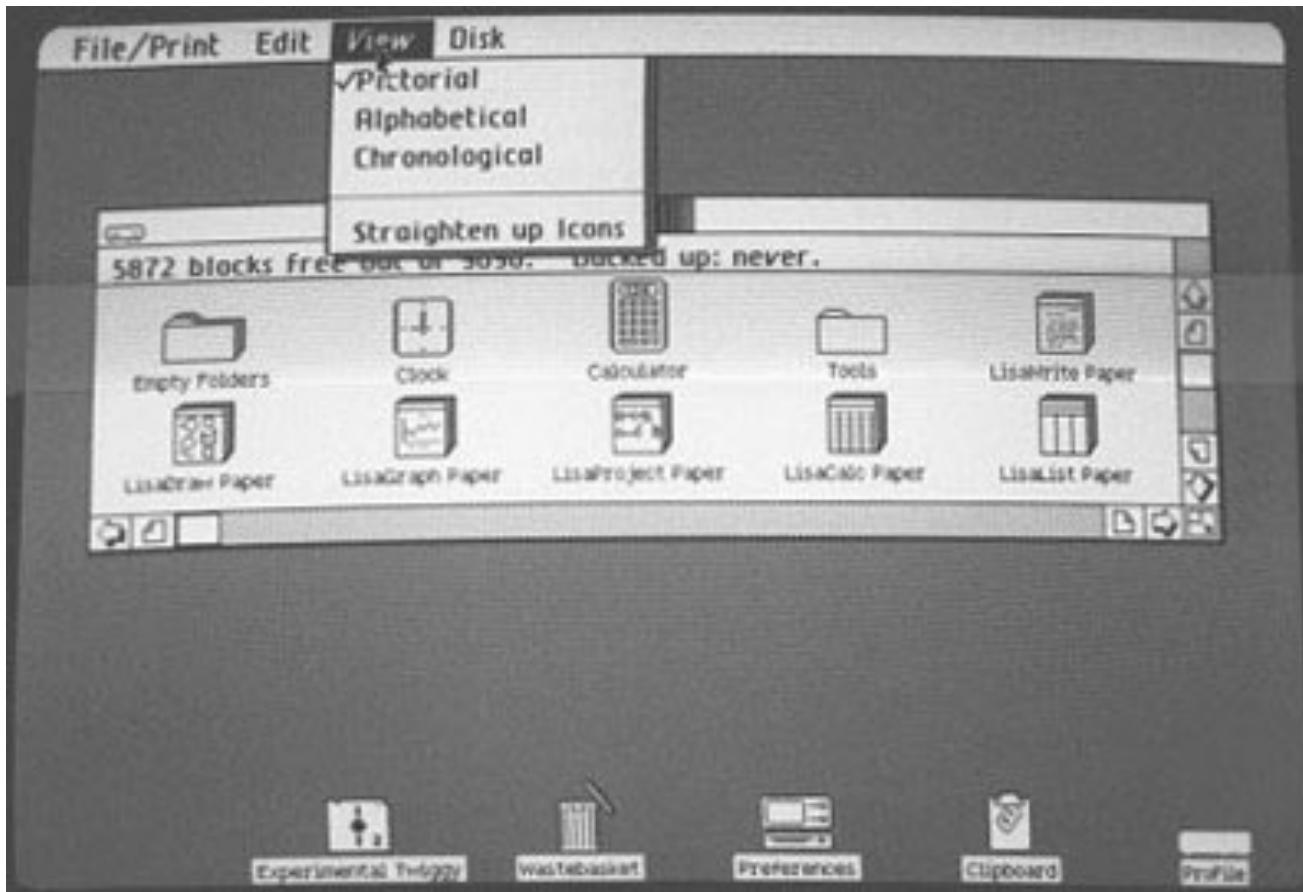
# Xerox Star (1981)



- Basiert auf den Forschungsarbeiten bei XEROX ab 1970
- Tastatur + Mausbedienung
- Grafische Software-Entwicklungsumgebung mit Smalltalk



# Apple LISA (1983)



- Steve Jobs hat die XEROX GUI für Apple kopiert
- Tastatur + Mausbedienung
- Software-Entwicklung in C und Pascal

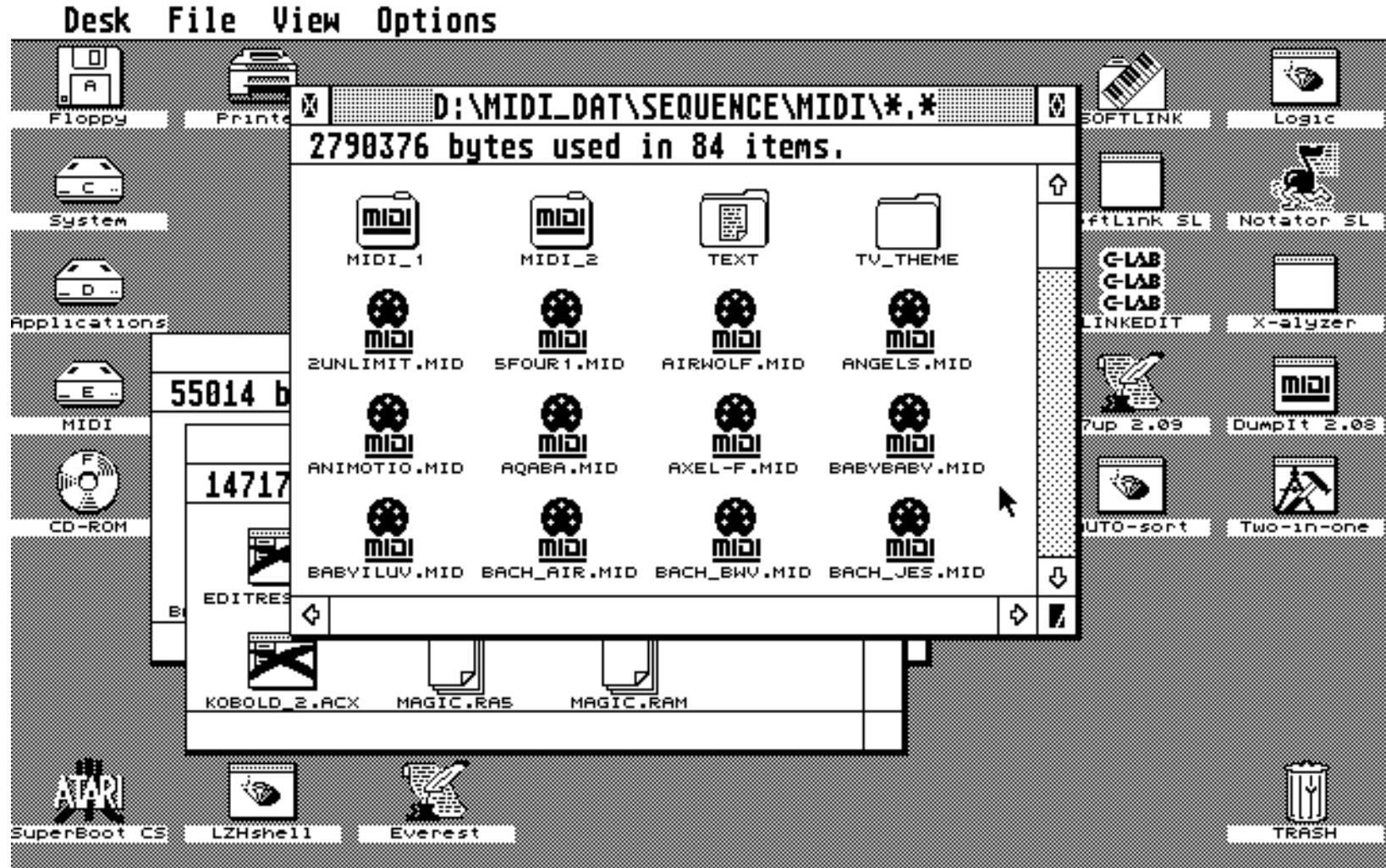


# Apple Macintosh (1984)



- Mini Bildschirm
- Kostengünstig
- Software-Entwicklung in C und Pascal

# Atari ST - GEM (1984)

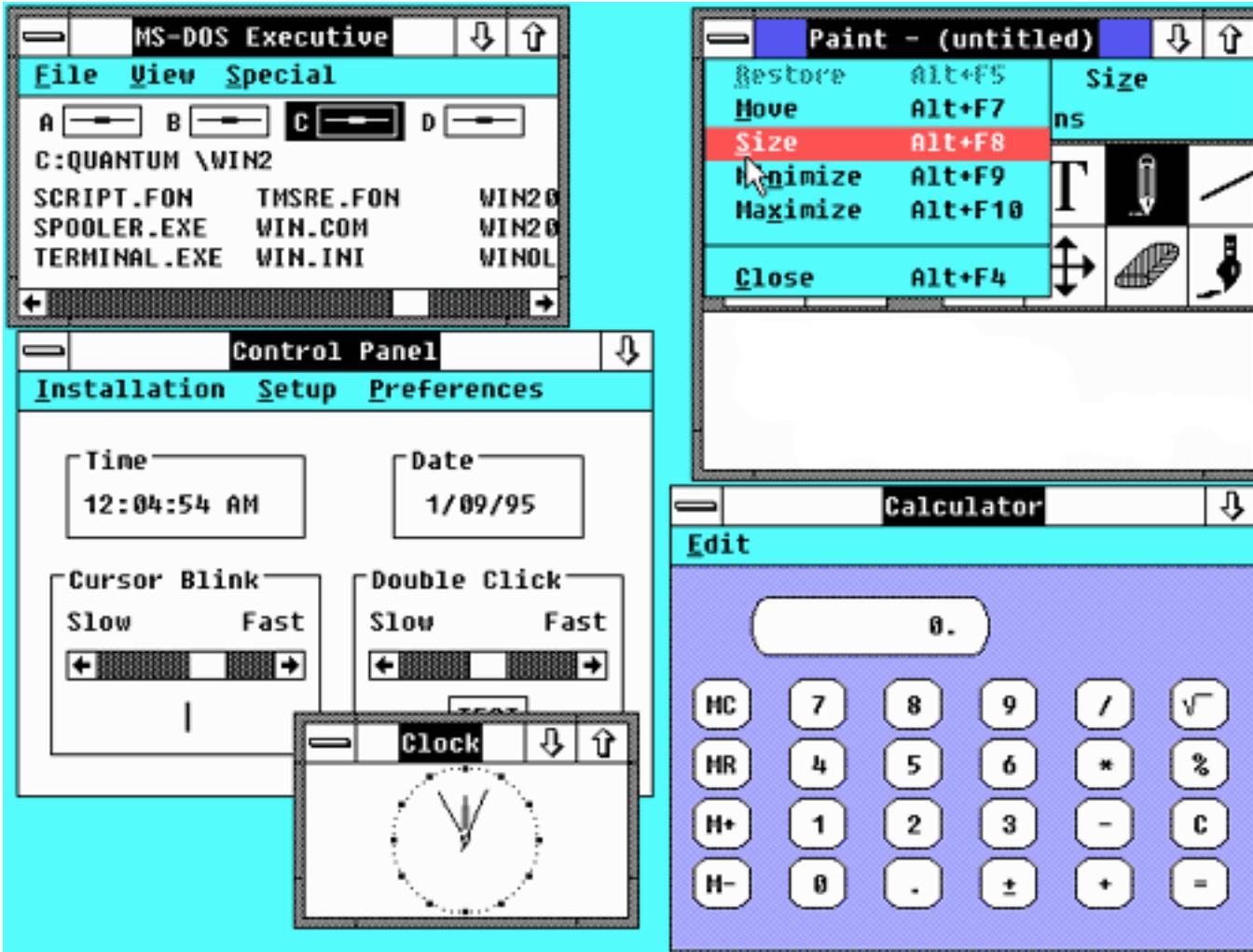


■ Verwendung im Kreativbereich

■ Tastatur + Mausbedienung

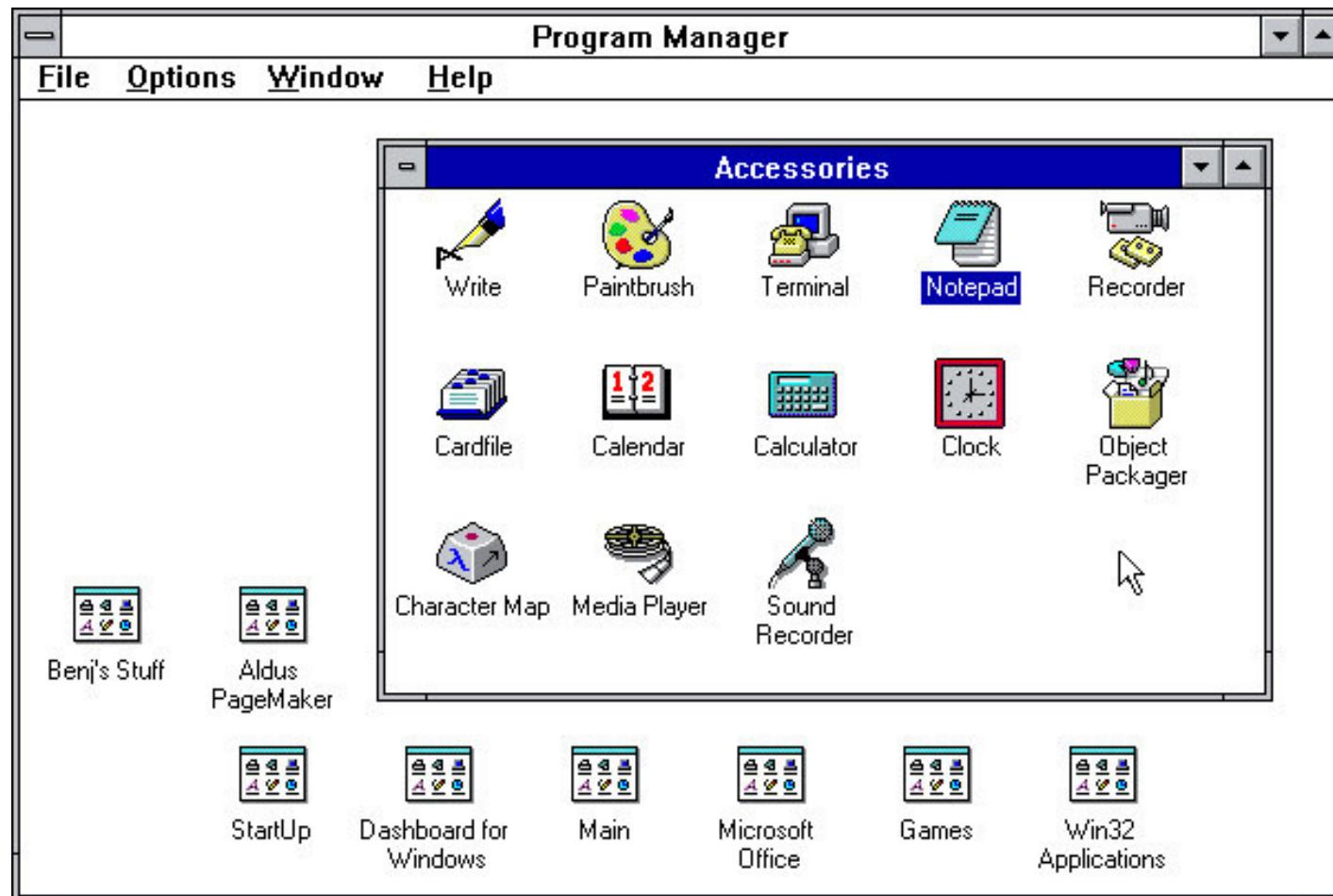
■ Software-Entwicklung in C

# Microsoft Windows 2 (1987)



- Wenig erfolgreich
- Software-Entwicklung in C

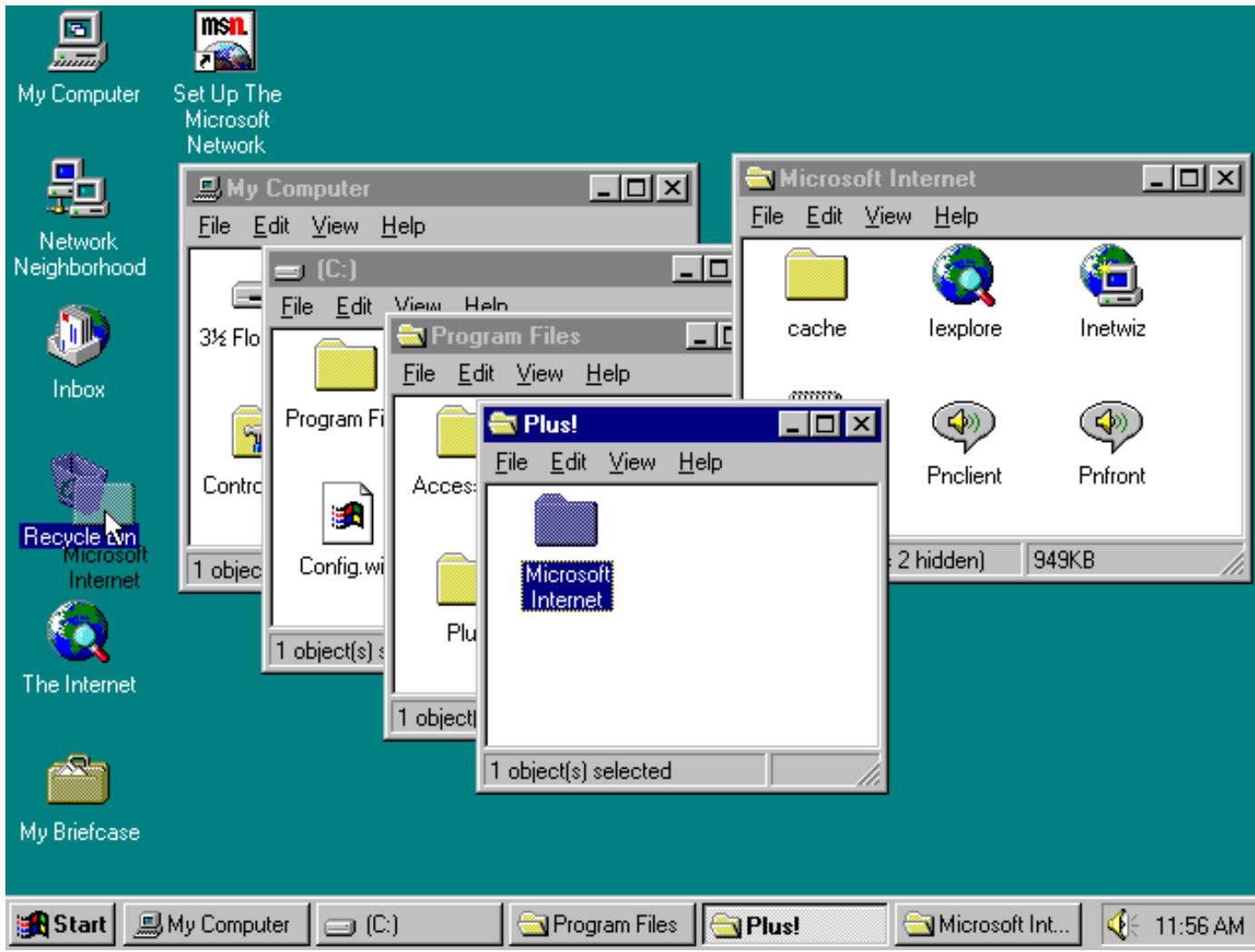
# Windows 3.x (1990)



- Erfolgreich
- Killeranwendungen: Winword + Excel
- Software-Entwicklung in C und C++



# Windows 95 (1995)



■ Sehr erfolgreich

■ Software-  
Entwicklung in C  
und C++

# 1995

- Bis Mitte der 1990 Jahre hatte jede grafische Umgebung eigene Oberflächenbibliothek – Die Programmiersprache war i.R. „C“
  - Microsoft: Windows-API
  - Apple: C-API
  - Unix: X-Windows
- Plattformunabhängige GUI Entwicklung war extrem aufwendig
- C eignet sich aufgrund seines Sprachdesigns (z.B. Größen von Datentypen hängen von der Hardware ab) nur bedingt für plattformunabhängige Programmierung



# 1995 - Heute

## - JAVA

- Die JAVA-Programmiersprache und -Plattform ist verfügbar auf allen gängigen Systemen. JAVA ermöglicht mit Java Swing und JavaFX auch den Bau von grafischen Oberflächen (mit Einschränkungen)

## - .NET

- Microsoft entwirft mit .NET eine OS unabhängige Plattform für Windows Systeme und schafft mit Windows Forms und später WPF die Basis für plattformunabhängige GUIs

## - IOS / OSX

- Apple schafft mit COCOA die Basis für IOS und Mac OSX – Portabilität wird hier nicht über eine virtuelle Maschine hergestellt sondern mittels Crosscompiler

## - Diverse

- Diverse Hersteller bieten plattformunabhängige Bibliotheken an (z.B. Trolltech QT)

## - HTML/JS

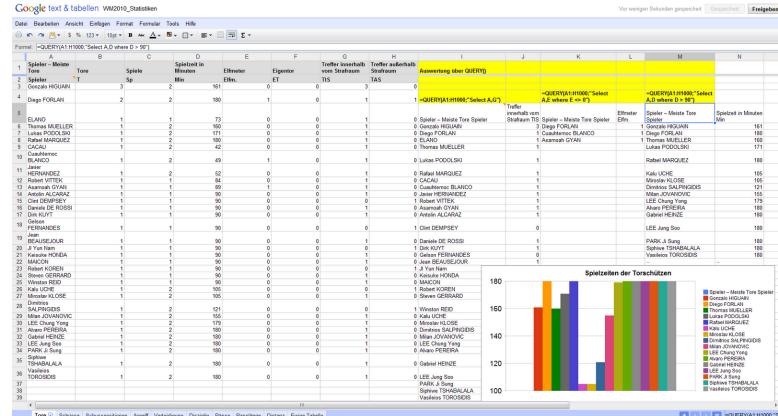
- Der Browser wird zunehmend als Enterprise UI Platform eingesetzt



# Die Zukunft



- Mehr mobile Anwendungen
  - Mehr Kanäle
  - Der Browser wird zum OS
  - Internet der Dinge
  - Gutes Design und Usability wird zur Pflicht
  - 3D

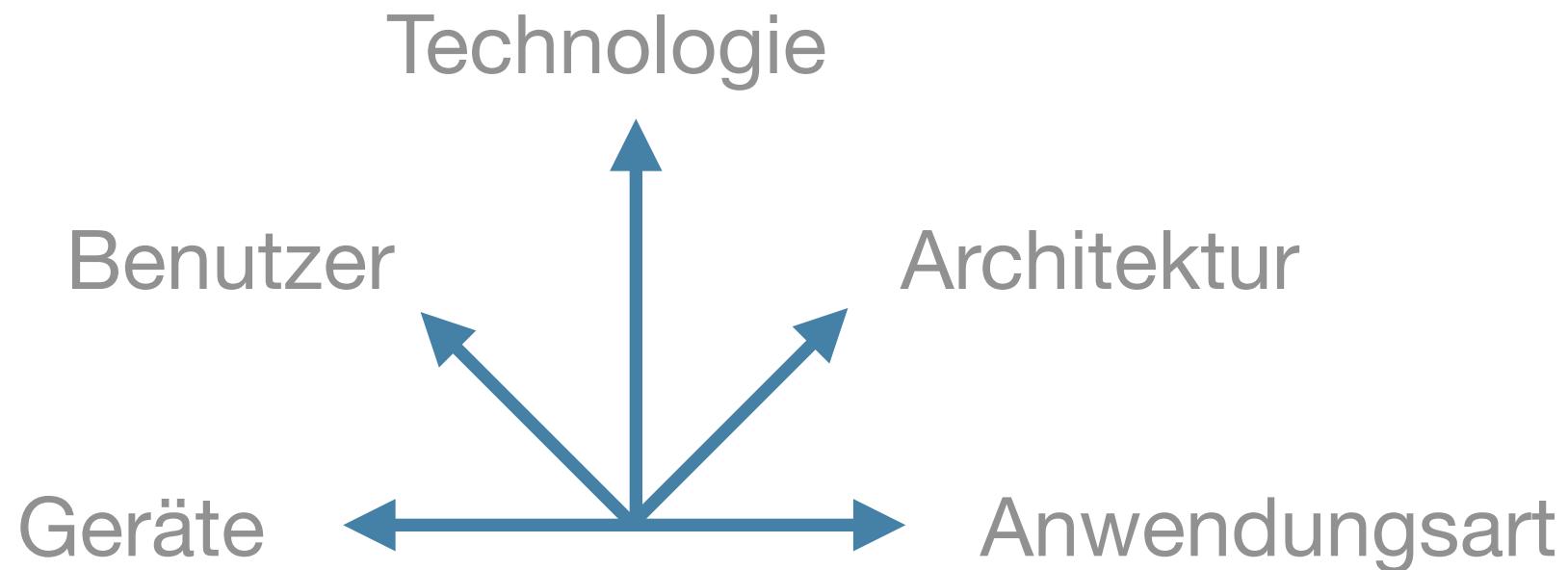


# Agenda

1. Historie und Entwicklung
2. **Typen grafischer Oberflächen**
3. Standardprobleme GUI
4. Programmiermodelle für GUI



# Einflussfaktoren beim Entwurf grafischer Oberflächen



# UI Technologien

- Native Oberflächen
  - Windows API, MacOS Cocoa, iPhone SDK, ...
- Interpretierte Oberflächen
  - Skriptsprachen (z.B. TCL/TK)
  - Browser (JS + HTML5)
- Oberflächen auf Basis virtueller Maschinen
  - .NET WPF
  - JavaFX
- Textbasierte Oberflächen
  - Terminal (z.B. IBM 3270, Unix Shell ...)



# UI Architekturtypen

- **FAT-Client**
  - Oberflächen und Geschäftslogik werden am Client ausgeführt. Remotezugriffe finden nur auf die Datenhaltung/Datenbank statt.
- **Thin-Client**
  - „Dummer Client“ - Die Oberfläche wird am Server erzeugt. Dialogsteuerung und UI-Logik ist ebenfalls serverseitig.
- **Rich-Client**
  - Der Client führt die Oberflächenlogik aus. Dialogsteuerung und UI-Logik wird am Client ausgeführt. Die Businesslogik ist am Server.
- **Smart-Client**
  - Internet Ausprägung eines Rich-Client (RIA)



# UI Anwendungsarten

- Experten Oberfläche zum Arbeiten
  - Excel, Videobearbeitung
- Mobile Oberflächen
  - Tablets, Telefone
- Spezialgeräte
  - Oberflächen für Industrie 4.0
  - Autosteuerung, Navigationsgeräte, Fernseher
- Oberflächen für den Gelegenheitsnutzer oder Konsumenten
  - Kaufportale (Amazon, Ebay ...)



# Der Benutzer und das Endgerät ist beim Entwurf und der Realisierung grafischer Oberflächen entscheidend

- Experte vs. Gelegenheitsnutzer
- Spezial-Schnittstellen vs. Consumer-Devices
- Mobile first?
- Maus und Tastatur?
- Auflösung
- Accessibility für Benutzergruppen mit eingeschränkten Fähigkeiten



# Textbasierte Oberflächen sind die „Mutter aller betrieblichen Informationssysteme“ z.B. IBM 3270

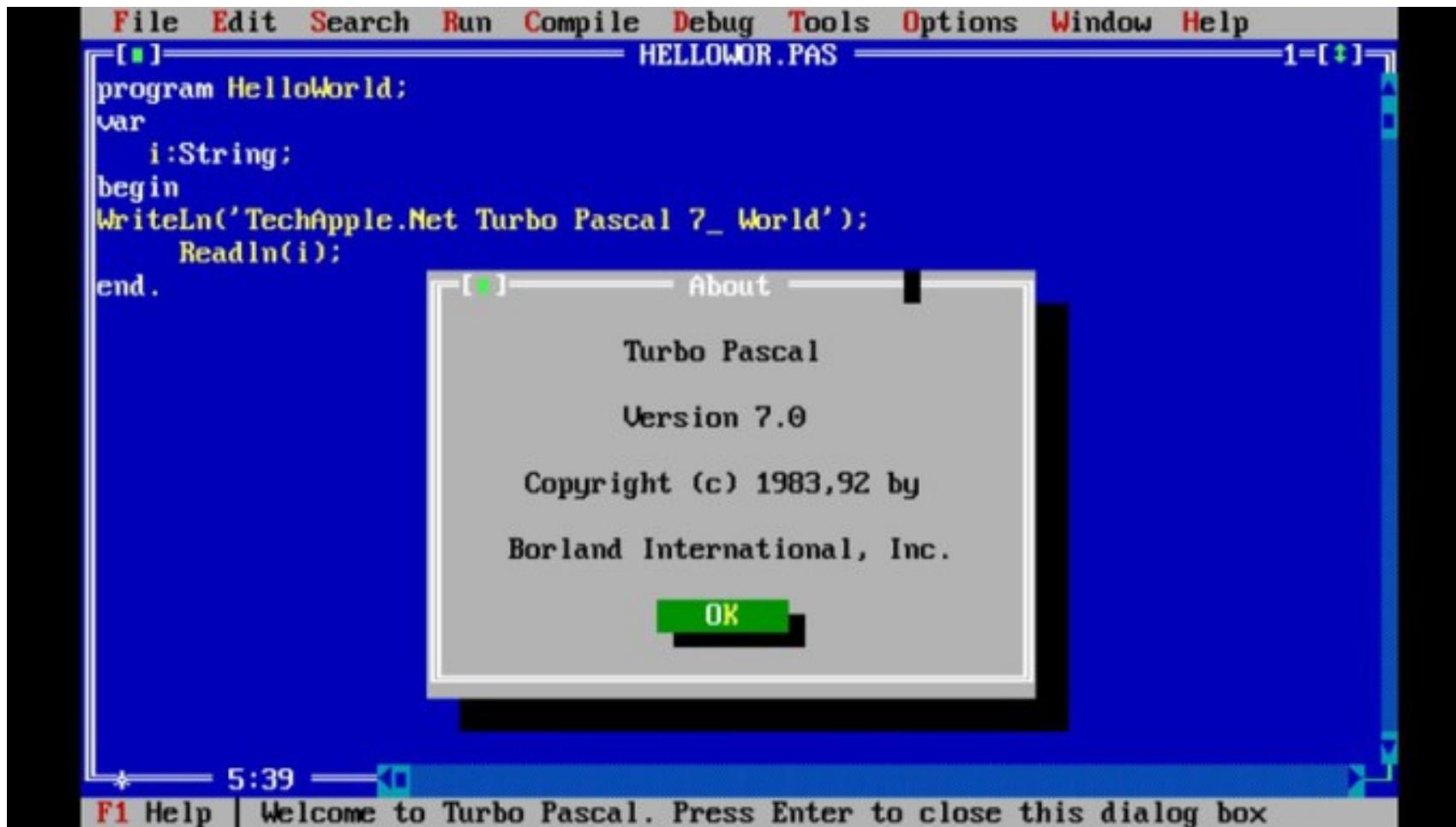
- IBM 3270 (-Terminals)
- Zugriff auf IBM-Host
- Eigenschaften
  - **Textbasierte Oberfläche 24 x 80 Zeichen**
  - Bedienung nur mit Tastatur  
(Blind bedienbar)
  - Sehr schnell
  - Navigation über Kürzel
- Als „Middleware“ verwendet

The screenshot shows a window titled "Sitzung A - Terminal". The menu bar includes "Datei", "Bearbeiten", "Übertragung", "Darstellung", "Funktionen", "Fenster", and "Hilfe". The main area displays a form with fields like "Maske", "Spar/Vsnr", "Ngnr", "Datum", and "BEPL1". A message in red text reads "bitte ein Objekt -einschließen! (Lotsen-Nr.: "30") ----- D TEST". Below this, there's a section titled "Grunddatenbild 1 - Bestand" with fields for "Kdnr", "Grkdnr", "Geskz", "Sachb...", "ORG.Ber..", "Dok.vom..", "AGT.....", "Storno...", and various dates and codes. The bottom of the screen has a status bar with "Bestand Antrag X Navigation:" and "Lotsen-Nr.:", and a footer with "MB a" and "14/014".

Screenshot aus: Vorlesungseinheit „Spezifikation großer Informationssysteme Benutzungsoberflächen“, Rolf Lewandowski, TU-Wien

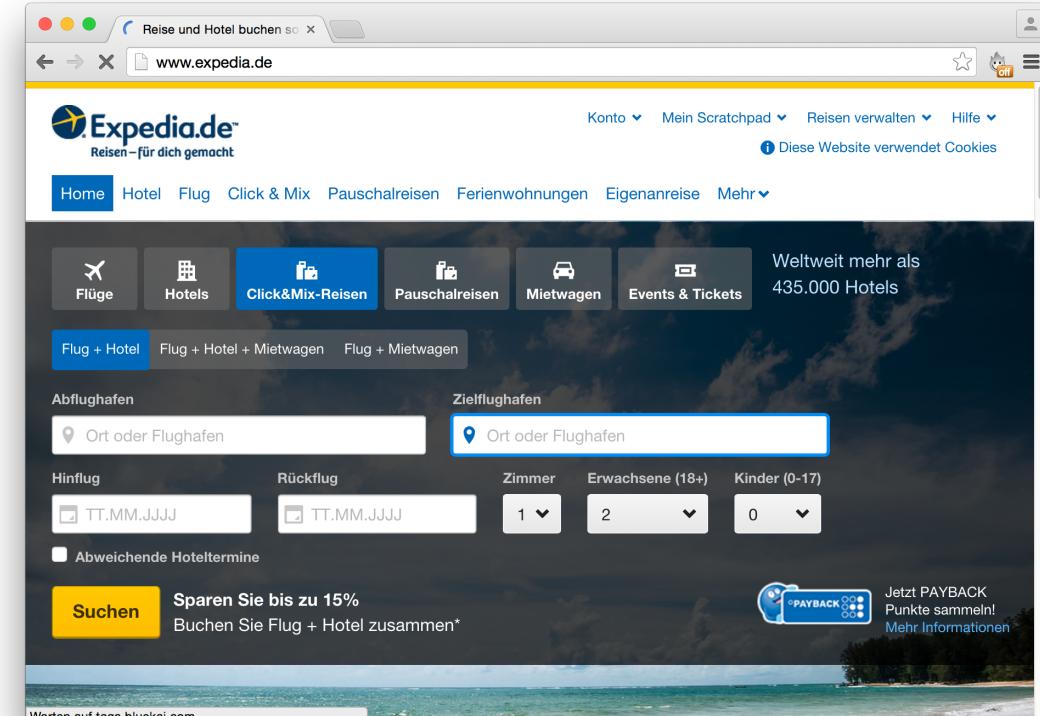


# Text-GUIs mit Mausbedienung

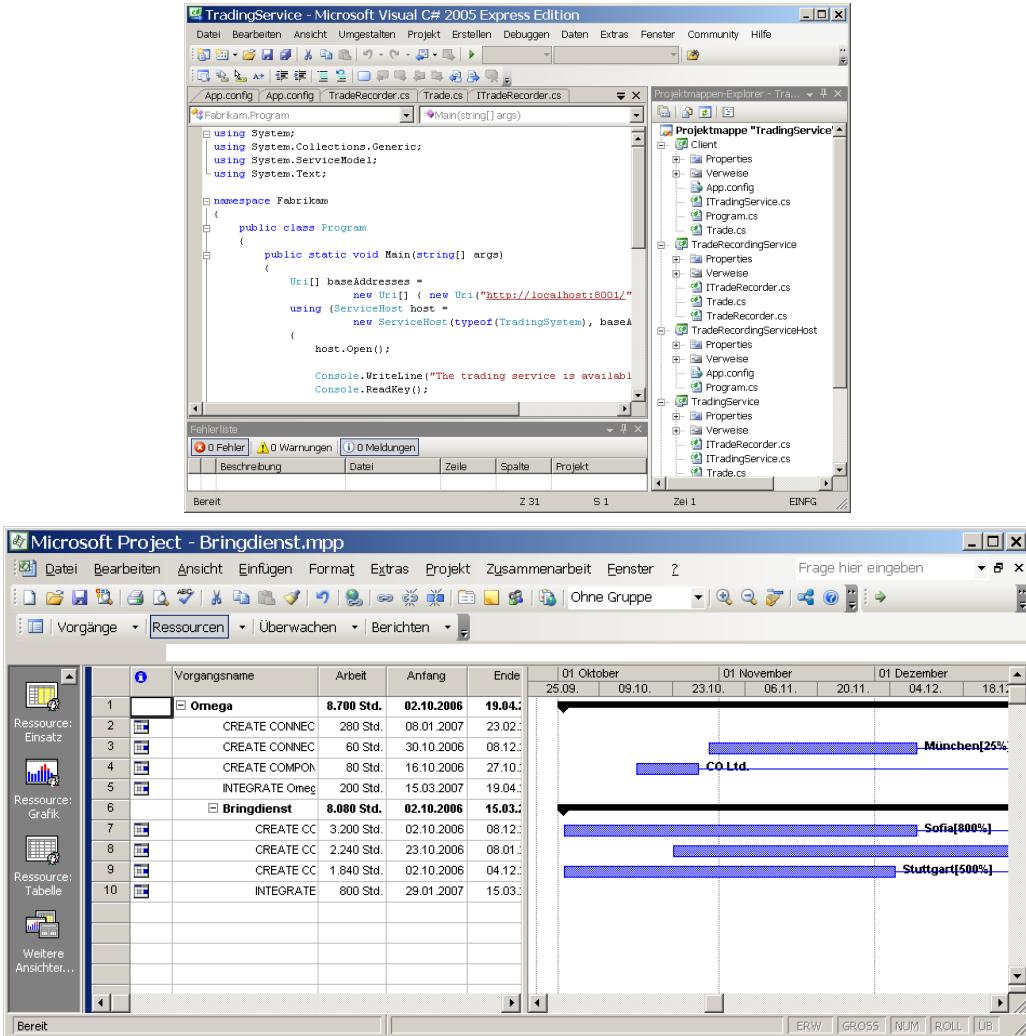


# Web-Oberflächen

- Beispiele: Flugbuchung bei Lufthansa.de, Buchkauf bei Amazon.de
- Technische Plattform: **Browser** (Chrome, Safari, Firefox...)
- Eigenschaften
  - Bedienung mit Maus, Tastatur, Finger (Mobil)
  - Häufig formular-basiert
    - Strukturierte Eingabe von Informationen
  - Der Trend: Single-Page Anwendungen
    - HTML5 und JS ist die technische Basis
- Sehr weit verbreitet  
(E-Commerce, B2B, B2C)



# Native-Oberflächen auf dem Desktop

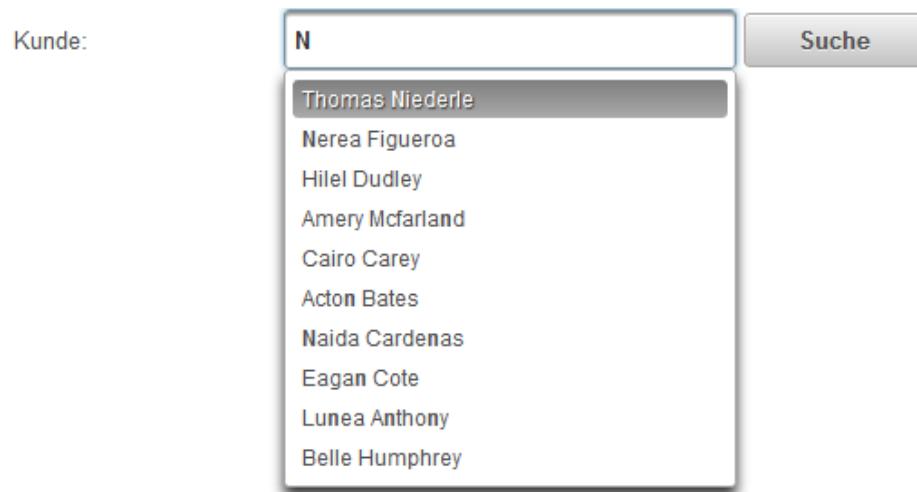


- Beispiele: Word, Eclipse, iTunes, Photoshop, ...
- Technische Plattform: **Betriebssystem** wie Linux, (bzw. Laufzeitumgebung wie Java VM, CLR, ...)
- Eigenschaften
  - **Hohe Interaktivität**
  - Häufig Planungs- / Kreative Oberfläche
  - Vielfältige Gestaltungsmöglichkeiten
  - Bedienung mit Maus, Tastatur, TouchPads, usw.



# Smart Clients im Web

- Trennung von nativen Oberflächen und Weboberflächen verschwindet zunehmend
- Beispiele: JQuery-UI, XY-Faces ...



# Grafische Oberflächen auf Tablets und Smartphones

- Grafische Oberflächen auf mobilen Geräten
- Technische Plattformen: Android, Windows, IOS
- Eigenschaften
  - Bedienung Berührung, über Stift, Tastatur, Sprache
  - **Kleines Display**
  - Ergonomie besonders wichtig



# Grafische Oberflächen im Auto

- Steuerung und Integration der  
> 1000 Fahrzeugfunktionen  
(Vereinfachung des Cockpits)
- Technische Plattform: Java VM (z.B. JavaFX) / C/C++
- Eigenschaften
  - Muss während der Fahrt bedienbar sein
  - Bedienung: Sprache, iDrive, Tastensteuerung
  - Ergonomie besonders wichtig



Quelle der Abbildung: [www.bmw.de](http://www.bmw.de)



# Agenda

1. Historie und Entwicklung
2. Typen grafischer Oberflächen
- 3. Programmiermodelle für GUIs**
4. Standardprobleme GUI



# Standardprobleme GUI

- Grafische Gestaltung
- Dialogzustand (enable/disable von Funktionen)
- Benutzerführung
  - Navigation, Menüs, Wizards
  - Backbutton, Undo/Redo
  - Bedienung ohne Maus, Tab-Reihenfolge, Fokus-Management
- Internationalisierung (I18N - internationalization, L10N - localization)
- Fehlerbehandlung, Eingabevalidierung
- Berechtigungen (Wer darf das? Wer darf diese Daten sehen?)
- Online Hilfe, kontextsensitive Hilfe



# Es gibt zwei Programmiermodelle zum Bau von grafischen Oberflächen

- Programmierte Oberflächen
  - GUI Frameworks (Java Swing, MFC, Qt, ...)
  - APIs von Betriebssystemen (Win 32 API, X11, ...)
- Deklarative / Markup Oberflächen
  - HTML – Oberflächen
  - XAML (WPF im .NET)
  - FXML (JavaFX)
- Viele UI Technologien verwenden BEIDE Ansätze
  - View: Deklarativ
  - Controller/Model: Programmiert



# Agenda

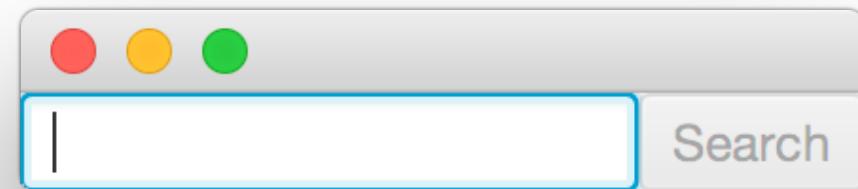
1. Historie und Entwicklung
2. Typen grafischer Oberflächen
3. Standardprobleme GUI
4. **Programmiermodelle für GUIs**



# Programmierte Oberflächen

## Am Beispiel Java-FX

```
public class SearchApplication extends Application {  
  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
        SearchApplication.launch(args);  
    }  
  
    @Override  
    public void start(Stage primaryStage) throws Exception {  
        Scene scene = new Scene(new SearchView());  
        primaryStage.setScene(scene);  
        primaryStage.show();  
    }  
}
```

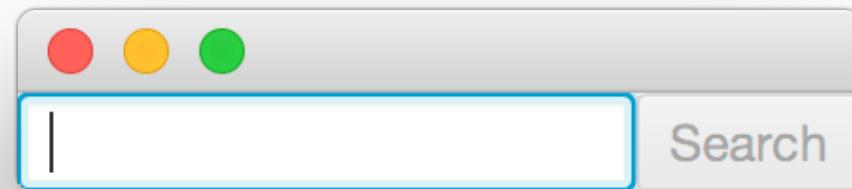


```
public class SearchView extends HBox {  
  
    private final Button searchButton;  
    private final TextField searchText;  
  
    public SearchView() {  
  
        searchText = new TextField();  
        searchText.setPromptText("Enter a Expression");  
  
        searchButton = new Button("Search");  
        searchButton.disableProperty().bind(  
            searchText.textProperty().isEmpty()  
        );  
  
        getChildren().addAll(searchText, searchButton);  
    }  
}
```



# Deklarative Oberflächen

## Am Beispiel Java-FX



```
public class SearchApplication extends Application {  
  
    @Override  
    public void start(Stage stage) throws Exception {  
        Parent root = FXMLLoader.load(getClass().getResource("SearchView.fxml"));  
  
        Scene scene = new Scene(root);  
  
        stage.setScene(scene);  
        stage.show();  
    }  
  
    /**  
     * @param args the command line arguments  
     */  
    public static void main(String[] args) {  
        launch(args);  
    }  
}
```

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<?import java.lang.*?>  
<?import java.util.*?>  
<?import javafx.scene.*?>  
<?import javafx.scene.control.*?>  
<?import javafx.scene.layout.*?>  
  
<HBox maxHeight="-Infinity"  
       maxWidth="-Infinity"  
       minHeight="-Infinity"  
       minWidth="-Infinity"  
       xmlns="http://javafx.com/javafx/8"  
       xmlns:fx="http://javafx.com/fxml/1">  
  <children>  
    <TextField fx:id="searchText" promptText="Enter a expression" />  
    <Button fx:id="searchButton" mnemonicParsing="false" text="Search" />  
  </children>  
</HBox>
```

