Applikationsentwicklung mit XForms

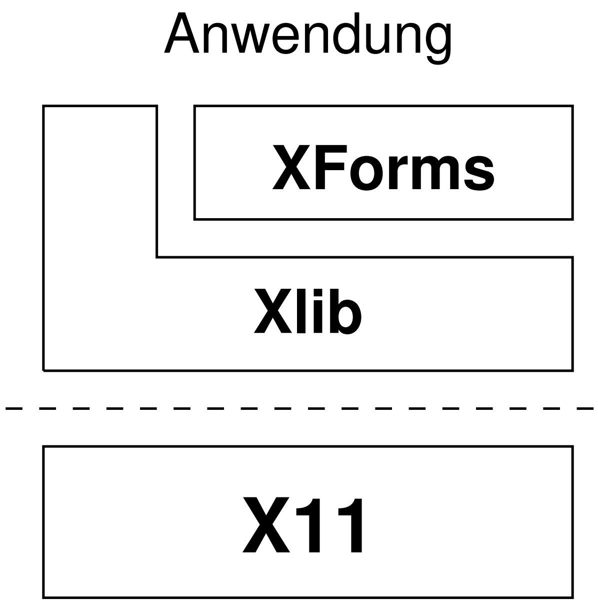
Nils Magnus, Michael Kleinhenz
Unix-AG, Universität Kaiserslautern
{magnus, kleinhen}@unix-ag.uni-kl.de

V1.2 – 13. Mai 1997

Zusammenfassung

Die XForms-Library stellt eine Reihe von grafischen Benutzerelementen zur Verfügung, die insbesondere mit dem dazugehörigen Interfacebuilder `fdesign` leicht zu einer eigenen Applikation zusammengestellt werden können. Zusammen mit der GNU-Entwicklungsumgebung bietet sich hier eine komplette Programmieroberfläche für Programme mit einem professionellen Look & Feel.

Benutzeroberflächen

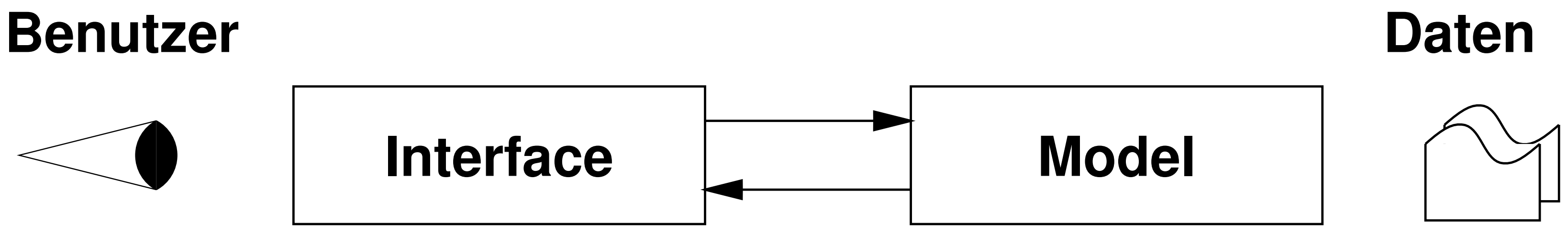


- verteiltes Fenstersystem
- Client-Server-Architektur
- verschieden mächtige “Schichten”

Graphical User Interfaces (GUIs)

- OSF/Motif
- Tcl/Tk
- Qt
- XForms

Programme mit Benutzeroberflächen



- Schnittstellen für Datenaustausch

- unabhängige Events

- mehrere Fenster

⇒ anderer Kontrollfluß

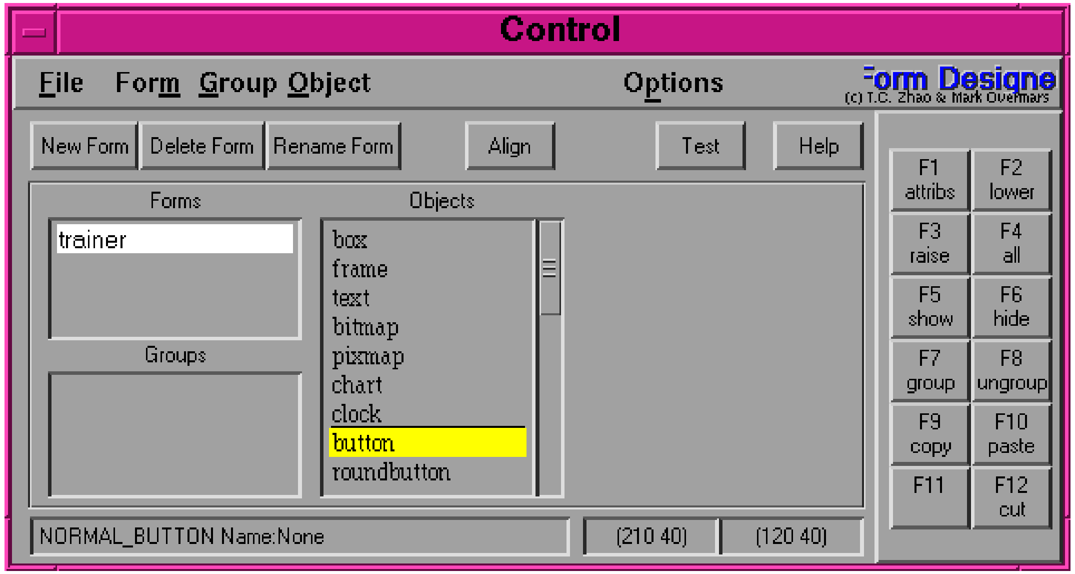
Vorteile von XForms

- Einfache Bedienung
- Unterstützung von C als Programmiersprache
- Weitere Anbindung an `perl` und `phython`
- Interfacebuilder mitgeliefert
- Gute Dokumentation
- frei erhältlich

Aufbau von XForms

- XForms ist eine Bibliothek, die zu eigenen Routinen dazugebunden wird

- Da die Erzeugung von Interface-Definitionen aufwendig ist, wird dies grafisch von `fdesign` übernommen



`fdesign`

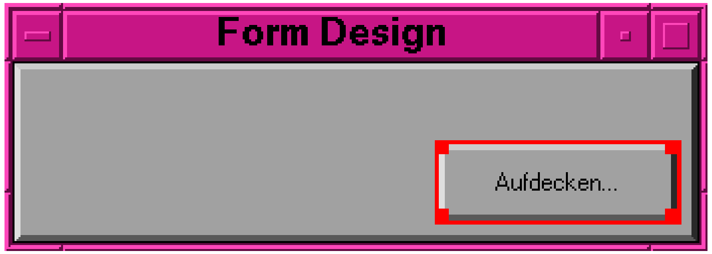
- Entwurf von Oberflächen mit der Maus
- Arbeit ähnlich einem Zeichenprogramm
- Große Auswahl an Oberflächenelementen steht zur Verfügung:
  - Knöpfe
  - Eingabefelder
  - Rahmen
  - Menus
  - ... vieles mehr

Beispielanwendung

Implementation des Interfaces für einen einfachen Vokabeltrainer.

Anforderungen:

- Anzeige einer Vokabel
- Eingabefeld für Überset-zung
- Knopf zum Aufdecken



Mögliche Erweiterungen:

- Menuleiste zur Steuerung
- Auswahlmenu für aktuelle Lektion

Beispiel-Model

Die eigentliche “Funktionalität” steht für dieses Beispiel schon zur Verfügung. In der Datei `model.c` sind folgende Funktionen bereits implementiert:

```
void LadeLektion(char *filename);
```

Paare von deutschen und englischen Wörter werden aus der übergebenen Datei in den Hauptspeicher geladen.

```
const char *deutschesWort(void);
```

Gibt einen Zeiger auf das aktuelle deutsche Wort zurück.

```
const char *englischesWort(void);
```

Gibt einen Zeiger auf das aktuelle englische Wort zurück.

```
void naechstesWort(void);
```

Schaltet auf das nächste Wort um; nach dem letzten Wortpaar geht es am Anfang weiter.

Bedienung von `fdesign`

- neuem Form Namen geben
- Auswählen der Oberflächenelemente
- Plazieren mit der Maus
- Rechte Maustatste aktiviert Element, mittlere Maustaste vergrößert/verschiebt
- F12/Cut entfernt Elemente

“Ergonomische” Plazierung von:

- Textfeld
- Eingabefeld
- Aufdecken-Taste

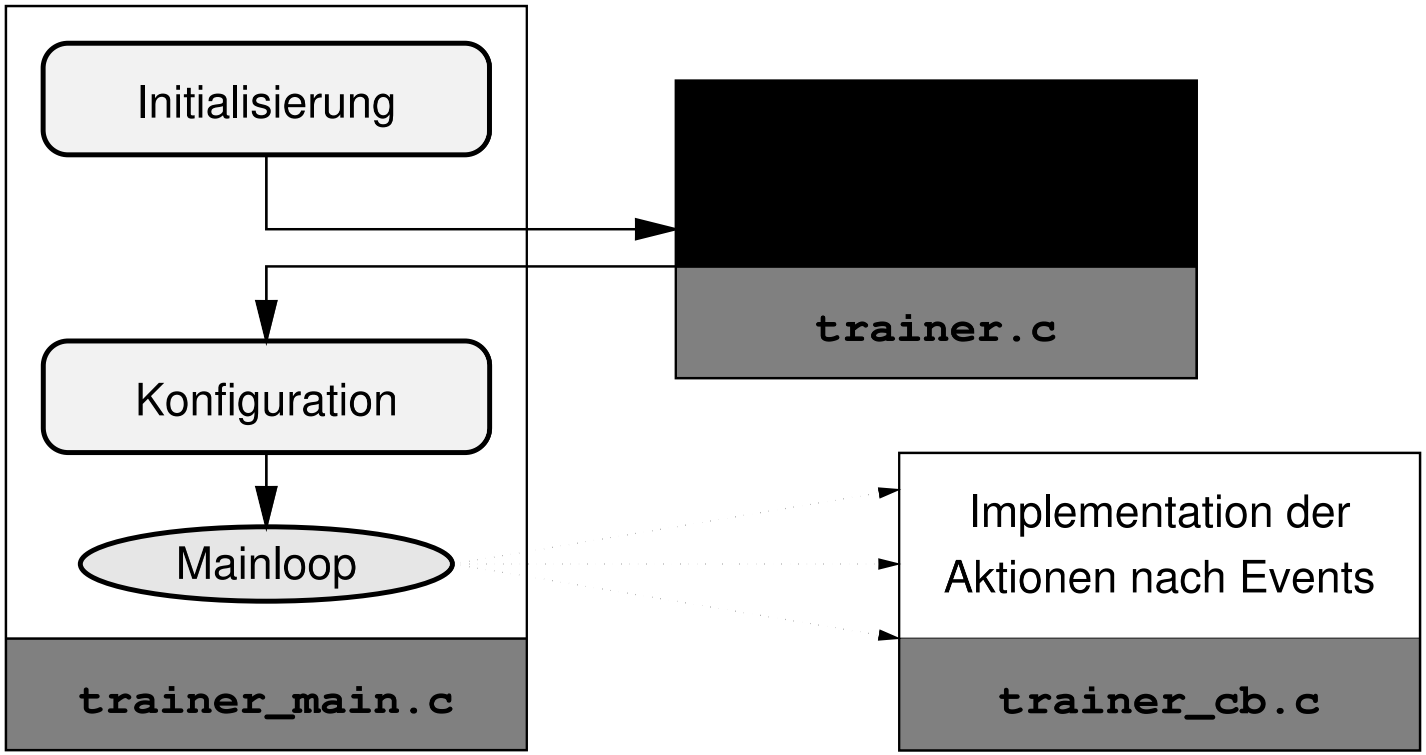
Testen und Konfigurieren

- Mit dem Test-Button kann das erzeugte Interface ausprobiert werden

- Im Attribut-Menu (F1 oder Doppelklick mit rechter Maustaste) können vielfältige Parameter geändert werden, die das Erscheinungsbild beinflussen:
  - Art der Umrandung
  - Schriftart und -größe
  - Farben

Schnittstelle zwischen Interface und Model

Nach dem Generieren der C-Dateien kann nun die Verbindung des Interfaces mit dem Model in Angriff genommen werden:



Konzept der Callbacks

- wenn Element “aktiviert” wird
  $\rightarrow$ Funktion aufrufen

- jedes aktivierbare Element muß daher einen Namen haben
  $\rightarrow$ Eintrag von Namen und Callback im Attrib-Menu

- der Callback-Funktion wird das Objekt übergeben, das den Aufruf veranlaßt hat

```
void solveButtonCB(FL_OBJECT *ob, long data)
{
    fl_set_object_color(ob, FL_BLACK, FL_WHITE);
}
```

Exkurs: Menuzeile

- Auswahl von Oberflächenelement “Menu”
- rechts bzw. linksbündig am oberen Rand plazieren
- mit Rahmen (Frame) gruppieren:
  - Umrandung: `UP_FRAME`
  - “Unter” die Menus legen (F2/Lower)
- Funktion testen
- Eintragen von Callbacks für jedes Menu
- Menueinträge werden später implementiert

Entwicklungsumgebung: Emacs

Der Emacs ist ein sehr mächtiger Editor und fungiert als “Schaltstelle” unserer Entwicklungsumgebung:

- großer Funktionsumfang, wir werden nur wenige Features nutzen
- Umgang mit mehren Dateien (sog. Buffern) in mehreren Fenstern sog. Frames)
- direkter Aufruf von Compiler und Debugger
- Sprung zu Fehlerstellen im Quellcode, auch über mehrere Dateien hinweg
- farbige, syntaxsensitive Darstellung von Code
- automatisches Einrücken von Code
- Vorlage von Templates

Bedienung des Emacs

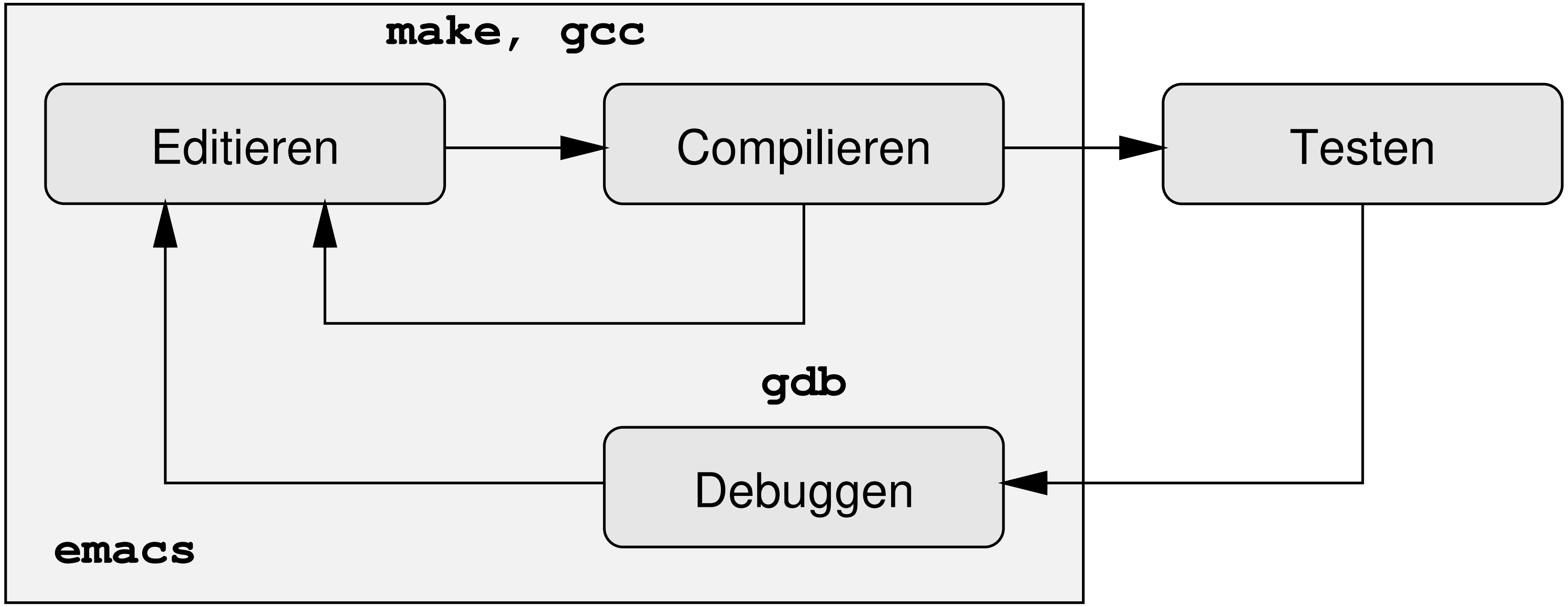
Der GNU-Emacs kann über Menus mit der Maus oder mit Tastaurkommandos gesteuert werden:

- **Laden** von Dateien: C-x C-f (Ergänzung der Kommandozeile mit TAB, bzw. Menu durch zweimaliges TAB, Auswahl mit mittlerer Maustatste)
- **Speichern** der aktuellen Datei: C-x C-s. In neue Datei: C-x C-W. Speichern aller Buffer: C-x s.
- Wechseln von **Buffern:** C-x b (Ergänzung mit TAB). Sprung in anderen Buffer im selben Frame: C-x o. Alle außer aktuellen Frame ausblenden: C-x 1.
- **Compilieren:** F8. Sprung zum nächsten Compilerfehler: C-x ‘
- **Einrücken** einer Zeile Code: TAB

Initialisierung und Anpassung des Interface

- `fdesign` erzeugt eine Datei, die die Funktion `main()` enthält.
- Initialisierung der XForms-Library
- Aufruf der Interfacedefinitionen von `fdesign`
- eigene zusätzliche Anpassungen des Interfaces:
  - Hinzufügen der Menudefinition

Entwicklungszyklus



- make: Makefile enthält Anweisungen, wie welches File zu compilieren bzw. zu linken ist.
- gdb: Debugger zum Schrittweisen Ausführen von Programmen

Übung

Deaktivieren von Text- und Eingabefeld:

- `fl_deactivete(object);`

Editieren des Callbacks:

- Einfügen von `puts(‘‘hello, world!’’);`

Compilieren und Testen

Vorgefertigte Widgets

Einige hilfreiche und häufig benötigte Bedienelemente (“Widgets”) sind schon fertig in XForms vorhanden, unter ihnen:

- Fileselectionbox:

```
const char *filename = fl_show_fselector("Select Lesson",
                                         ".",
                                         "*.les",
                                         "lesson1.les");
```

- Ausgabefenster für Hinweise und Warnungen:

```
fl_show_message("Lesson is reset", "", "");
fl_show_alert("Error!", "Can't open file:", filename, 0);
```

Fertigstellung der Funktionalität

Callback für **Eingabefeld**:

1. Eingabe eines Wortes

2. Check, ob Wort korrekte Übersetzung ist (`strcmp()`)

   - Wenn richtig: → nächstes Vokabelpaar
   - Wenn falsch: → Fehlerbox aufpoppen lassen

Callback für **Aufdeck-Button**:

1. Aufdecken gedrückt

2. Wort in Textfeld einfügen