GTK+ Programmierung mit C#

VALUG - 13. März 2011

Wolfgang Silbermayr

13.03.2011

Inhalt

Pakete installieren

- Debian, Ubuntu ([sudo] apt-get install *<paketname>*):
  - Mono: `mono-devel`
  - GTK-Sharp: `libgtk2.0-cil-dev, libglade2.0-cil-dev`
  - Monodevelop: `monodevelop`
  - API-Dokumentation: `monodoc-browser, monodoc-gtk2.0-manual`
- Windows:
  - Mono: `http://mono-project.com/`
  - GTK-Sharp: `http://mono-project.com/` oder `http://monodevelop.com/`
  - Monodevelop: `http://monodevelop.com/`
- Allgemein:
  - API-Dokumentation online: `http://go-mono.com/docs/`

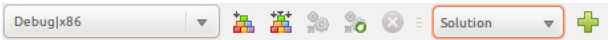
Projekte und Projektmappen

- Projekt:
  - Beinhaltet eine oder mehrere Quellcode-Dateien
  - Ist in einer einzigen Programmiersprache
  - Wird bei C# mit der Endung `.csproj` gespeichert
- Projektmappe (Solution):
  - Kann mehrere Projekte beinhalten
  - Die Projekte können in unterschiedlichen Programmiersprachen geschrieben sein
  - Die Projekte können voneinander abhängen
  - Wird mit der Endung `.sln` gespeichert

Neues Projekt anlegen

- Monodevelop starten
- **Neue Projektmappe beginnen**
- Kategorie **C#** wählen
- Projekttyp **Gtk#-2.0-Projekt** wählen
- Name eingeben (z.B. *Example*)
- Auf den Folgeseiten die Standardeinstellungen beibehalten

Projekt bauen und ausführen



- Projekt bauen (F7)
- Projektmappe bauen (F8)
- Anwendung ausführen (Control+F5)
- Anwendung debuggen (F5)

Basis: GObject

- Objekt-System, implementiert in C
- C bietet nativ kein Klassensystem, Implementierung größtenteils in Makros
- Nachteil: sehr viel Code zu schreiben, viel Routine nötig
- Vorteil: Bindings in viele gängige Sprachen (C++, C#, Java, Python, Perl, PHP etc.)
- Die Bindings fühlen sich für die jeweilige Sprache „natürlich" an
- Junge C#-ähnliche Programmiersprache „Vala" erzeugt als Zwischenschritt GObject-Code in C

GTK+

- Widget-System
- GTK-Widgets sind in GObject geschrieben
- Bietet Buttons, Texteingabefelder, Toolbars etc.
- Lizenz: LGPL
- Läuft plattformunabhängig auf Windows, Linux, MacOS uvm.
- Gnome-Desktop und XFCE basieren auf GTK+
- Umfangreiche Theming-Möglichkeiten

Einige einfache Widgets

- Button
  - Darstellung:

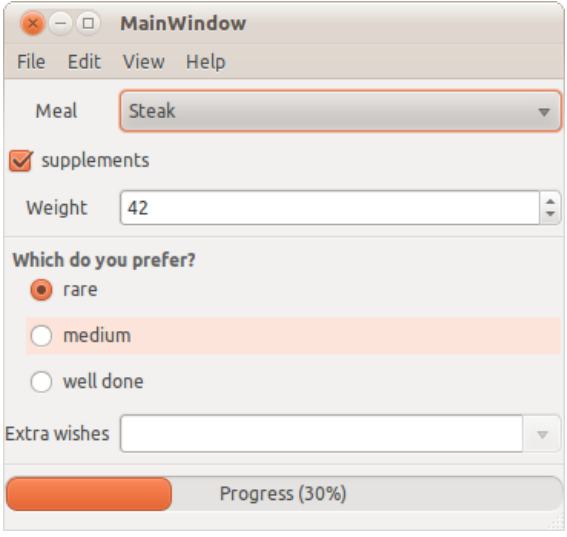
  - Beschriftung: Text und/oder Icon, oder beliebiges anderes Widget
  - Löst beim Klicken ein Event aus
- Label
  - Darstellung:

    Label
  - Zur Beschriftung anderer Elemente
  - Kann einfaches Markup verwenden (z.B. <b>)
- Entry
  - Darstellung:

  - Zur einzeiligen Texteingabe

Viele weitere Widgets verfügbar



Layouting

- Im Gegensatz zu Standard Windows-Applikationen: keine freie Platzierung
- Platzierung in Container-Widgets:
  - Bin

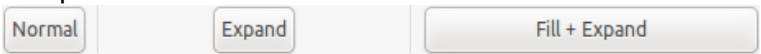
    
  - HBox, VBox

    
  - Table

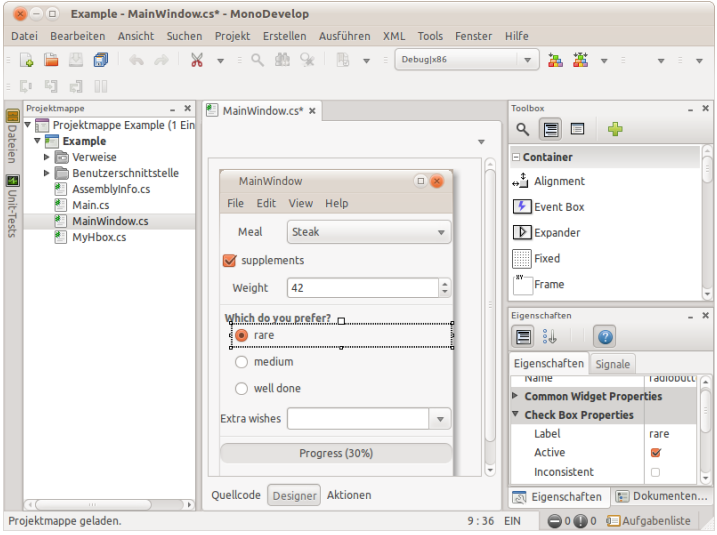
    
- Container-Widgets können ineinander geschachtelt werden

Fill, Expand

- **Fill**: Das Widget füllt den zur Verfügung gestellten Platz aus
- **Expand**: Das Widget dehnt sich aus und benützt den gesamten verfügbaren Platz
- Beispiel:



Selbst probieren



Klassen, Objekte, Variablen

- Klassen
  - Logische Bündelung von Funktionalität
  - Modellieren oftmals reale Dinge
  - Jede Klasse existiert nur ein Mal
- Objekte
  - Sind Instanzierungen von Klassen
  - Zu einer Klasse können mehrere Objekte existieren
  - Können Eigenschaften speichern
- Variablen
  - Sind vom Prinzip her „Links" (Referenzen) zu Objekten
  - Wenn sie auf kein Objekt zeigen, sind sie mit `null` initialisiert

Klassen, Objekte, Variablen

- Klassen
  - Logische Bündelung von Funktionalität
  - Modellieren oftmals reale Dinge
  - Jede Klasse existiert nur ein Mal
- Objekte
  - Sind Instanzierungen von Klassen
  - Zu einer Klasse können mehrere Objekte existieren
  - Können Eigenschaften speichern
- Variablen
  - Sind vom Prinzip her „Links" (Referenzen) zu Objekten
  - Wenn sie auf kein Objekt zeigen, sind sie mit `null` initialisiert

```
// the class
class MyClass {
  public void DoSomething() {
    // usually some code would be
    // written here
  }
}

// ----- usage: -----
public void DoSomethingWithMyClass {
  // declare a variable of type MyClass
  MyClass myClass = null;

  // instantiate a MyClass object
  // and assign it to the variable
  myClass = new MyClass();

  // call a method of the MyClass object
  myClass.DoSomething();
}
```

Properties

- Dienen dazu, Eigenschaften von Objekten nach außen verfügbar zumachen
- Können wahlweise Setter und Getter beinhalten
- Die Setter und Getter werden wie Methoden verwendet
- Setter: `value`-Keyword für den übergebenen Wert
- Getter: muss den gewünschten Ergebnis-Wert per `return` zurückgeben

Properties

- Dienen dazu, Eigenschaften von Objekten nach außen verfügbar zumachen
- Können wahlweise Setter und Getter beinhalten
- Die Setter und Getter werden wie Methoden verwendet
- Setter: `value`-Keyword für den übergebenen Wert
- Getter: muss den gewünschten Ergebnis-Wert per `return` zurückgeben

```
class MyClass {
  // the private variable
  private int strength;

  // the public property
  public int Strength {
    get { return strength; }
    set { strength = value; }
  }
}

// ----- usage: -----
class MyOtherClass {
  public void DoSomething() {
    MyClass myClass = new MyClass();
    // get the property's value
    Console.WriteLine("The␣strength␣is␣{0}",
        myClass.Strength);
    // set the property's value
    myClass.Strength = 10;
  }
}
```

Delegates, Events

- Delegates
  - Prototypen für Methoden
- Events
  - Können ausgelöst werden, um interessierte Objekte über Vorgänge zu informieren
  - Haben eine bestimmte Aufrufform, definiert durch einen Delegate
  - Auslösen funktioniert wie Methodenaufruf
  - Sind keine Methoden angehängt, ist der Delegate `null`; Daher vor Aufruf immer prüfen

Delegates, Events

- Delegates
  - Prototypen für Methoden
- Events
  - Können ausgelöst werden, um interessierte Objekte über Vorgänge zu informieren
  - Haben eine bestimmte Aufrufform, definiert durch einen Delegate
  - Auslösen funktioniert wie Methodenaufruf
  - Sind keine Methoden angehängt, ist der Delegate `null`; Daher vor Aufruf immer prüfen

```
// method signature
delegate void CountChangedDelegate(int count);

class MyClass {
  private int count;
  public event CountChangedDelegate CountChanged;
  public void Increase() {
    count++;
    if (CountChanged != null) {
      CountChanged(count);
    }
  }
}

// ----- usage: -----
class MyOtherClass {
  private void OnCountChanged(int count) {
    Console.WriteLine("Count is now {0}", count);
  }
  public void DoSomething() {
    MyClass myClass = new MyClass();
    // attach the method to the event
    myClass.CountChanged += OnCountChanged;
    myClass.Increase();
    // detach the method from the event
    myClass.CountChanged -= OnCountChanged;
  }
}
```

Generische Kollektionen

- Zum Speichern von mehreren Objekten gleichen Typs
- Der Typ wird nach Wunsch festgelegt
- Prominentes Beispiel: `List<T>`, wobei `T` der Typ der Liste ist

Generische Kollektionen

- Zum Speichern von mehreren Objekten gleichen Typs
- Der Typ wird nach Wunsch festgelegt
- Prominentes Beispiel: `List<T>`, wobei `T` der Typ der Liste ist

```
using System.Collection.Generic;

class MyClass {
  public void DoSomething() {
    // usually something would happen here
  }
}

// ----- usage: -----
class MyOtherClass {
  public void DoSomething() {
    List<int> intList = new List<int>();
    intList.Add(4);
    intList.Add(10);
    intList[0] += 10; // intList[0] is now 14

    List<MyClass> myClassList =
        new List<MyClass>();
    myClassList.Add(new MyClass());
    myClassList.Add(new MyClass());
    myClassList[1].DoSomething();

    foreach(MyClass myClass in myClassList) {
      myClass.DoSomething();
    }
  }
}
```


MainWindow
http://www.pro-linux.de/rss/index1.xml
Load
USA: Android ist Smartphone-Marktführer
Neues Scrollbar-Design für Ubuntu
Flash 10.3 Beta freigegeben
Apache 2.4 nähert sich
John Sullivan wird neuer FSF-Geschäftsführer
Nokia verkauft Qt-Geschäft
OpenAPC 1.4 mit Redundanz und Unterstützung für komplexe 2D/3D-Steuerung
Ubuntu 11.10 soll Oneiric Ocelot heißen
Nokia verkauft Qt-Geschäft
Nokia verkauft laut einer eigenen Mitteilung das Geschäft mit kommerziellen Qt-Lizenzen und dem zugehörigen Support an Digia. Das Unternehmen will zwar künftig weiterhin in die Entwicklung von Qt investieren, konzentriert sich aber nur noch auf die Kerninteressen.
Open in browser

Fragen?
Danke!