Hochschule für Technik
und Wirtschaft Berlin

University of Applied Sciences

Innovativ und vielfältig: die Hochschule für Technik und Wirtschaft Berlin

Fachbereich 2

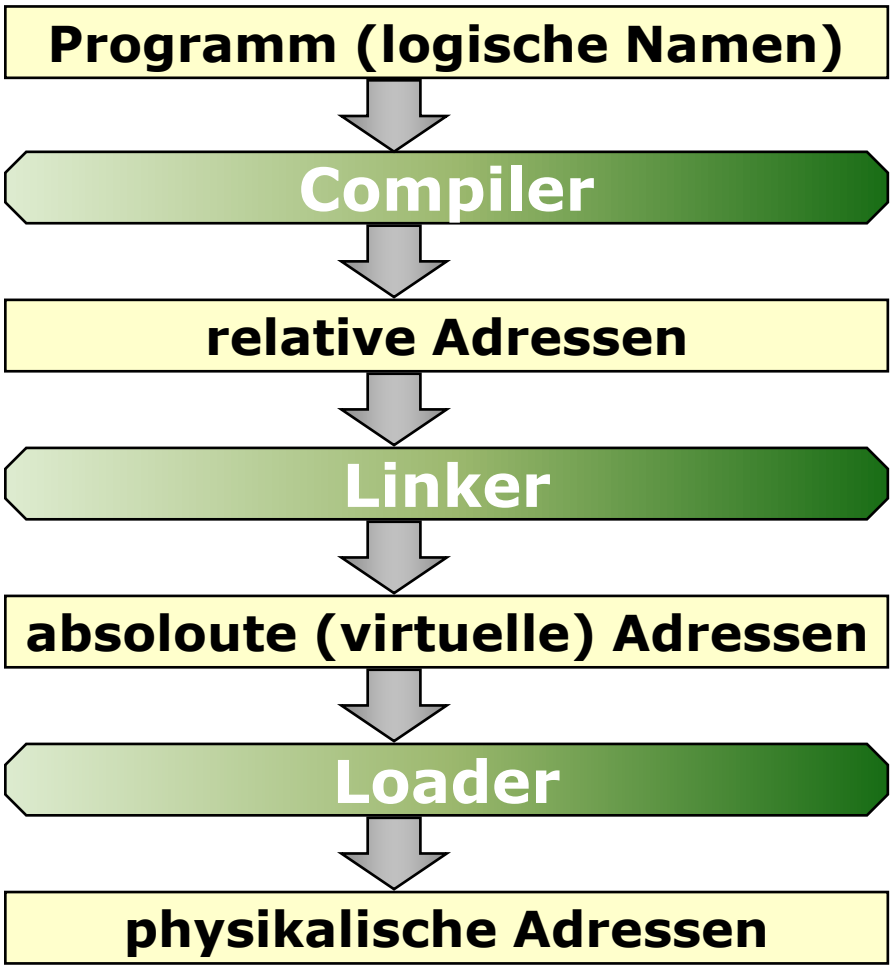
Informatik

Vorkurs **Informatik**

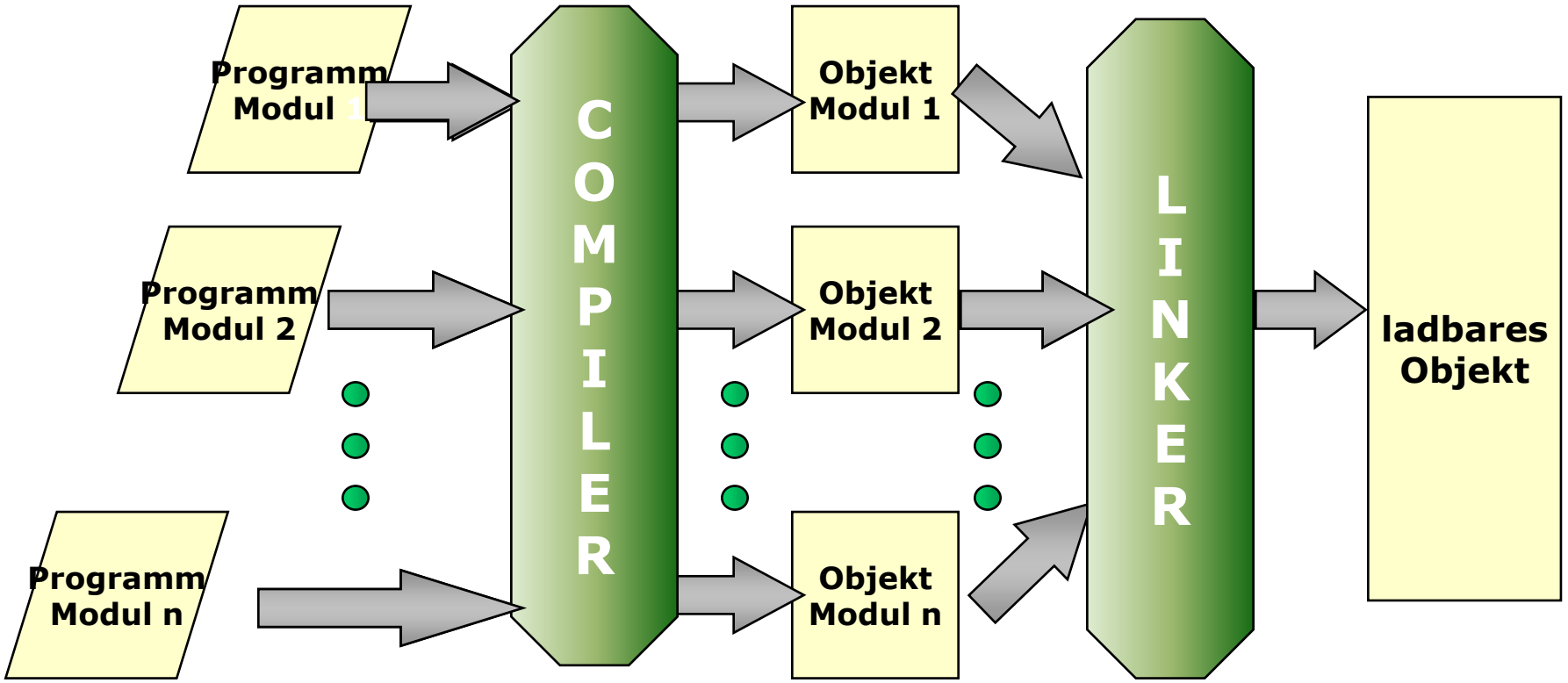
Lektion 8

PROGRAMMIEREN

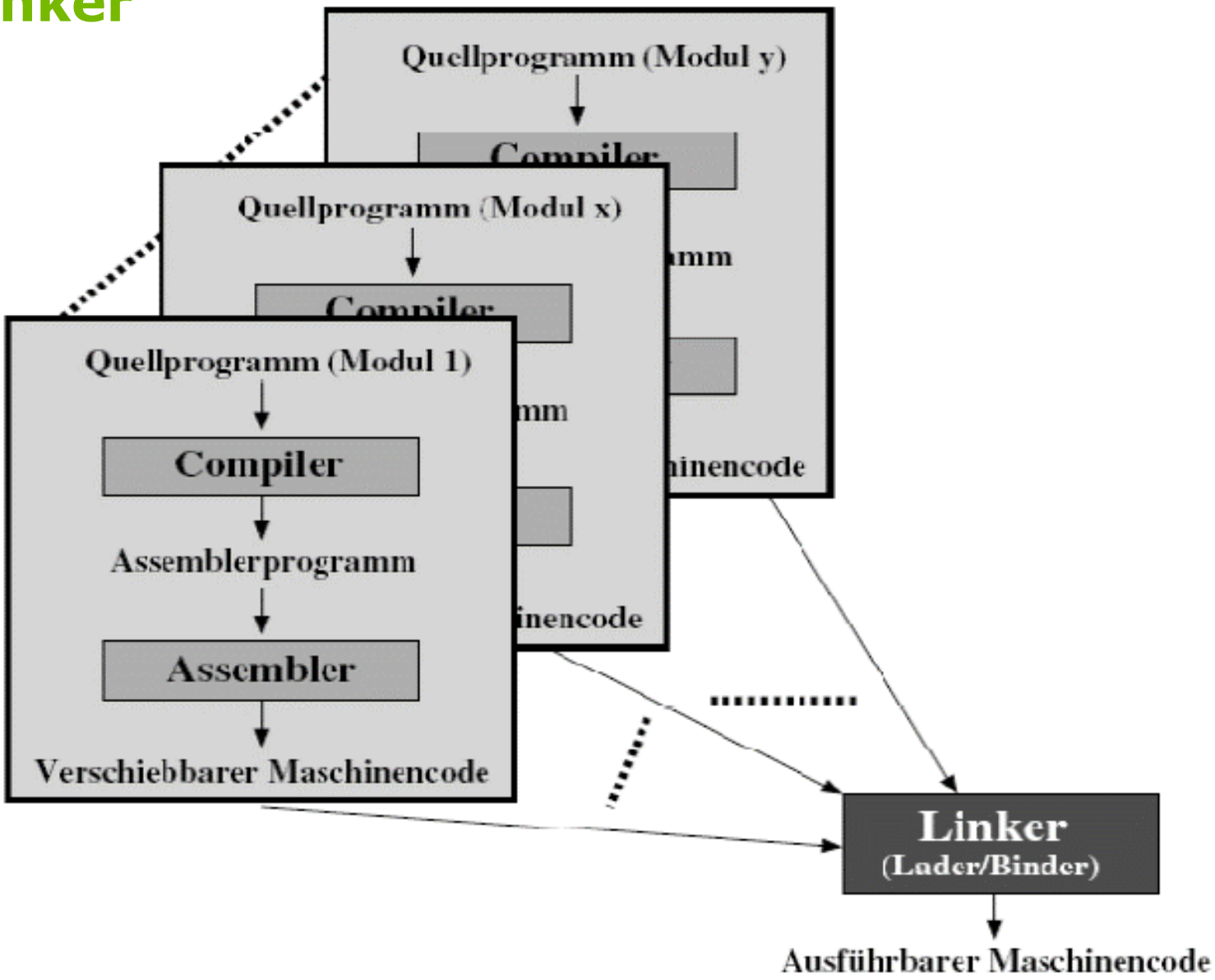
Adressumsetzung bsp C++



Erstellung eines ladbaren Objektcodes



Linker


Quellprogramm (Modul y)
Compiler
Quellprogramm (Modul x)
Compiler
Quellprogramm (Modul 1)
Compiler
Assemblerprogramm
Assembler
Verschiebbarer Maschinencode
Linker
(Lader/Binder)
Ausführbarer Maschinencode

Basis-Datentypen

char: Menge der Zeichen
int: Menge der ganzen Zahlen, die im Rechner darstellbar sind
float: Menge der darstellbaren Gleitkommazahlen mit einfacher Genauigkeit,
double: Menge der darstellbaren Gleitkommazahlen mit doppelter Genauigkeit,

Array: Zusammenfassung von zusammengehörigen Daten des gleichen Typs
String: Array von Zeichen

Variablen

```
tips = 12
10 + tips
```

```
waitress_name = “Mary Jane”
waitress_name
```

```
Python 2.7.11+ (default, Apr 17 2016, 14:00:29)
[GCC 5.3.1 20160413] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> foo = "Hello, "
>>> bar = "World"
>>> foo + bar
'Hello, World'
>>>
```

Variablen

- Gültigkeit der Variablen
  - Ort der Vereinbarung
  - im 'Programm' => globale Variable
  - im Block => lokale Variable
  - in der Funktion => lokale Variable
  - entscheidet über Benutzbarkeit
- Lebenszeit von Variablen
  - lokal: in der Funktion - nur während des Funktionsaufrufes (Block)
  - global: während der gesamten Programmausführung
  - static macht auch lokale Variable permanent

Operatoren, Ausdrücke und Anweisungen

- Formeln
  - dreiecksflaeche = grundlinie * hoehe / 2;
  - kreisflaeche = radius*radius*3.1415926;
  - umfang = 2*radius*pi;
- Unäre (einstellige) Operatoren
- Binäre (zweistellige) Operatoren

Multiplikation	*, /, % (Rest der int-Division)
Addition	+, -
Relation	<, <=, >, >=
Gleichheit	==, !=
logisches UND	&&
logisches ODER	||
Zuweisung	=, +=, -=, *=, …

Vorzeichen	+,-
Inkrement und Dekrement	++, --
Adresse	&
Negation	!

Gleichheit vs. Zuweisung

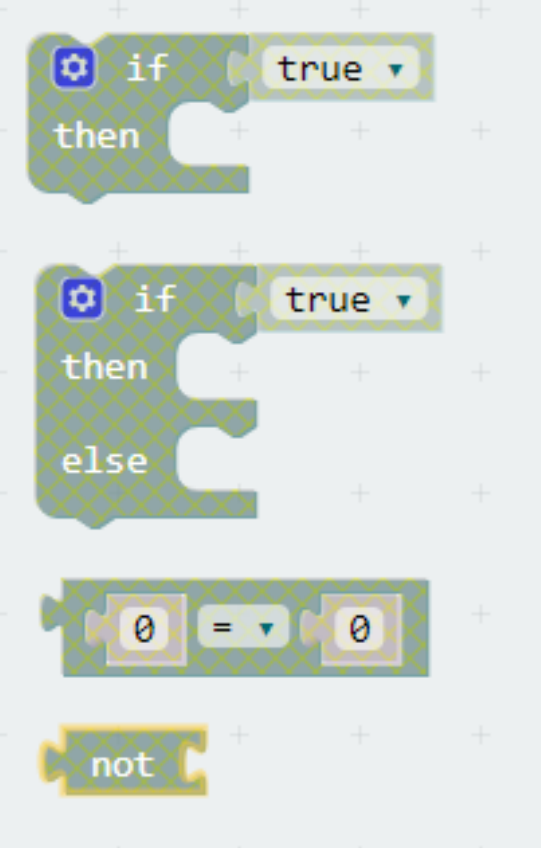
„If" und „Else"

```
word = raw_input("Please enter a four-letter word: ")
if len(word) == 4:
    print (word + " is a four-letter word. Careful now!")
else:
    print ("That's not a four-letter word. Try harder.")
```


```
if today is sunny:
    Call friends
    Walk to park
    Buy ice cream
else:
    Play video games
```

Operatoren: Entscheidung

Scratch	Java




```java
if ( Boolescher-Ausdruck )
{
    Anweisung1;
    Anweisung2;
    ...
    AnweisungN;
}
```

```java
if ( Bedingung )
{
    Anweisung1;
    Anweisung2;
}
/* Ansonsten führe Anweisung3 und Anweisung4 aus. */
else
{
    Anweisung3;
    Anweisung4;
}
```

Die while-Schleife

```
print "10"
print "9"
print "8"
print "7"
print "6"
print "5"
print "4"
print "3"
print "2"
print "1"
print "BLAST OFF!"
```

Start countdown at 10
While the countdown is bigger than 0
    Announce the countdown
    Reduce countdown by 1
Say "BLAST OFF!"

Untitled

Untitled

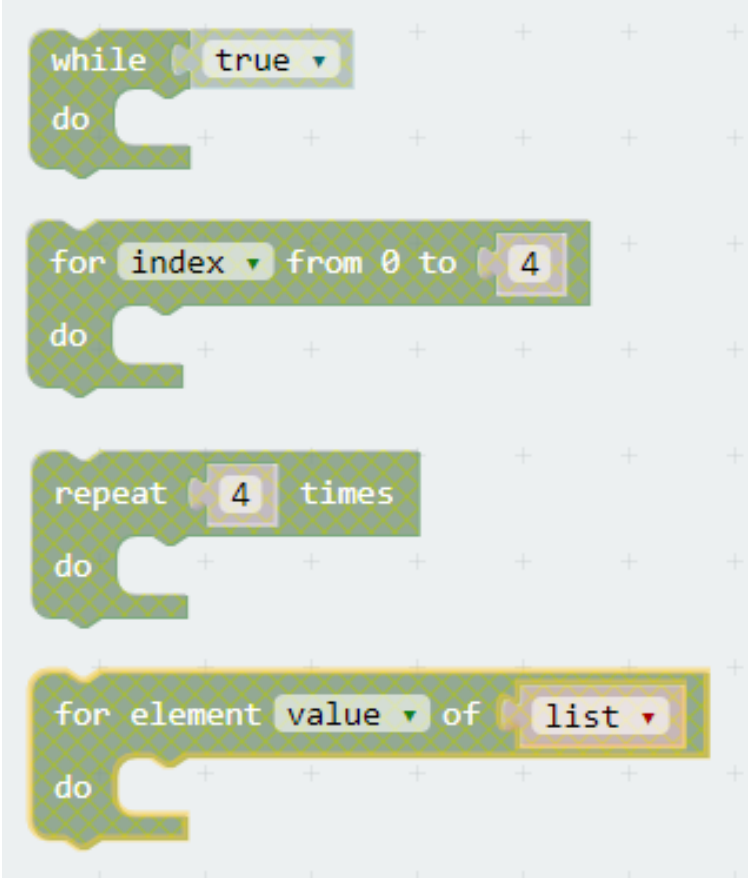
File Edit Format Run Options Window Help

```
countdown = 10
while countdown > 0:
    print countdown
    countdown -= 1
print "BLAST OFF!"
```

While True
Do something

Operatoren: Schleife

Scratch



Java

```java
// Variable i wird deklariert und initialisiert
// Abbruchbedingung wird festgelegt auf i < 5
// i wird nach jedem durchlauf um eins erhöht
for(int i=0; i<5; i++)
{
    // Die Ausgabe findet fünfmal statt (von 0 bis 4)
    System.out.println("i ist "+i);
}
```

```java
while( Boolescher Ausdruck )
{
   Anweisung;
}
```

```java
do
{
   Anweisung;
}
while( Boolescher Ausdruck )
```

For Schleife

for variable in sequence

```
for number in [2,4,6,8]:
    print number
print "Who do we appreciate?"
```

```
for x in range(1000):
    print x * x
```

Funktionen und Prozeduren

Einmal definierter Algorithmus für wiederkehrende Aufgaben, z.B.:

- Math. Funktionen:
  - Quadrat, Sinus, min(), max(), …
- Sensorabfrage
- Kreis malen

- Für Interrupts
  - onShake, onButtonPress
- Für Dekomposition
- Funktionen haben einen Rückgabewert
- Könnnen in Bibliotheken zusammengefasst werden
- Sehr oft Parameter

Funktionen

- help(pow)
- https://docs.python.org/2/library/functions.html

```
>>> abs(-10)
10
```

```
>>> negative = -10
>>> positive = abs(negative)
>>> positive
10
```

```
>>> bin(abs(-10))
'0b1010'
```

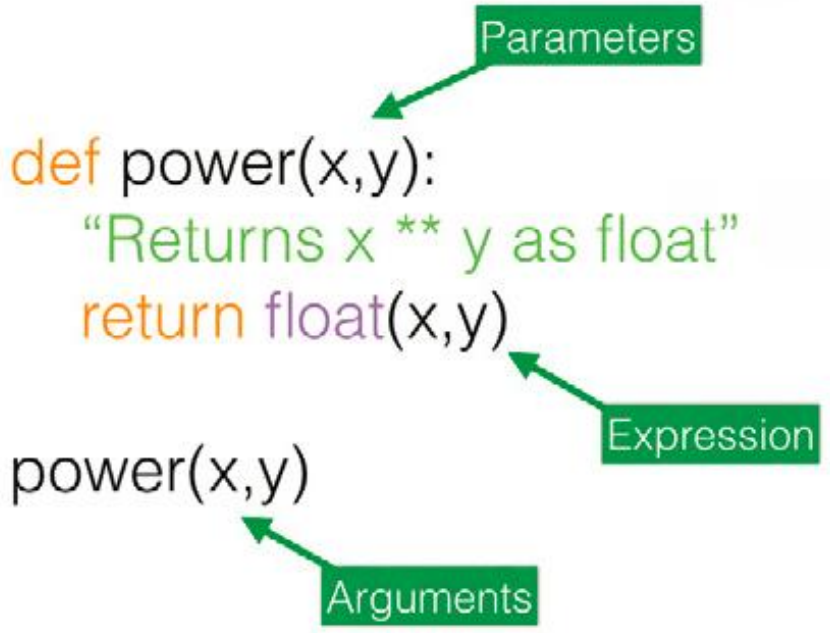
```
>>> pow(5,3)
125
>>> pow(
      pow(x, y[, z]) -> number
```

Eigene Funktionen

```
def function_name(parameters):
    function code
    return expression
```

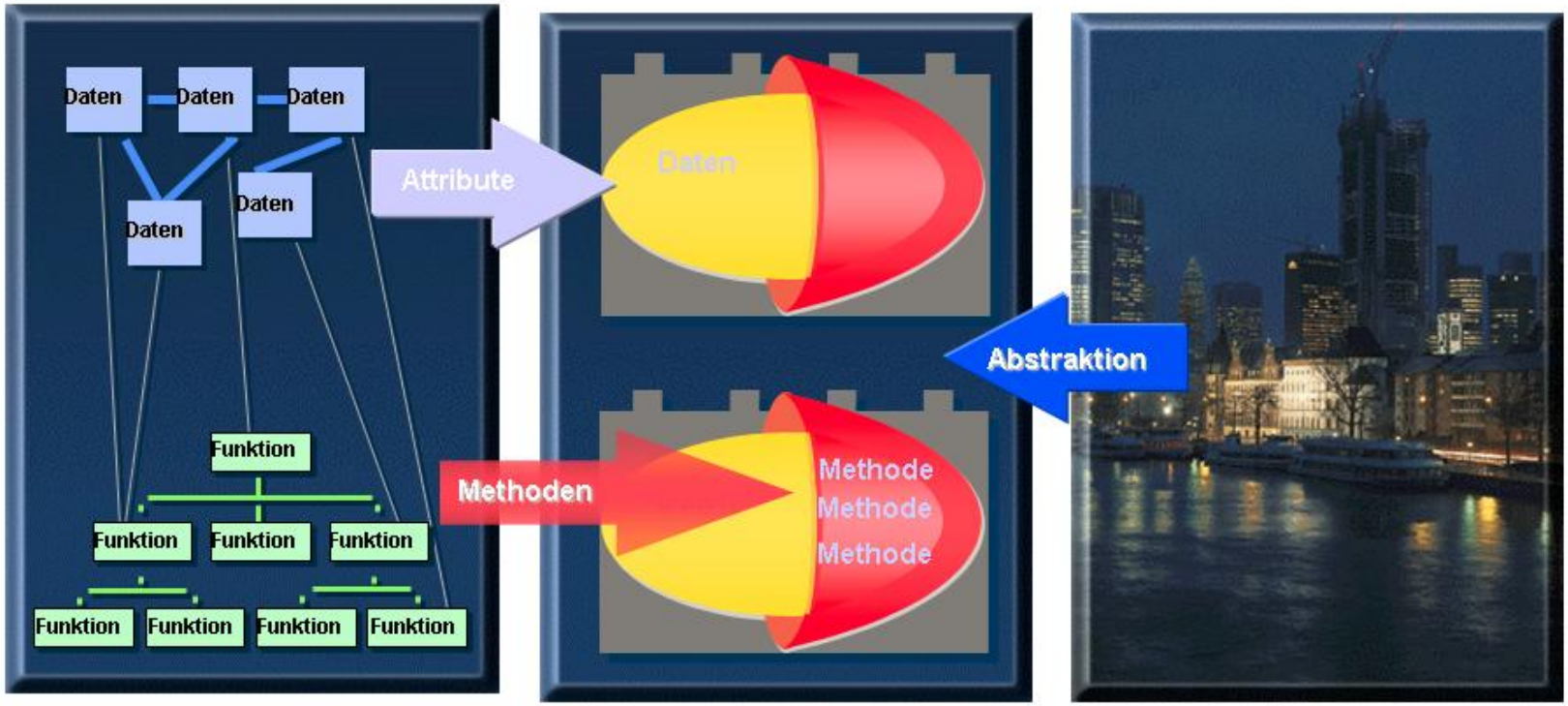
```
def square(number):
    return number * number


print (square(4))
```



Was ist Objektorientierung?

Datenkapselung



Funktionen und Daten
Datenmodell als Abstraktion der realen Welt

Software-Objekte
Objektmodell als Abstraktion der realen Welt

Objekte der realen Welt

Source: SAP

https://de.wikipedia.org/wiki/Objektorientierte_Programmierung

Objektorientierung

1. Alles ist ein Objekt,
2. Objekte kommunizieren durch das Senden und Empfangen von Nachrichten,
3. Objekte haben ihren eigenen Speicher (strukturiert als Objekte),
4. Jedes Objekt ist die Instanz einer Klasse,
5. Die Klasse beinhaltet das Verhalten aller ihrer Instanzen (in der Form von Objekten in einer Programmliste),
6. Um eine Programmliste auszuführen, wird die Ausführungskontrolle dem ersten Objekt gegeben und das Verbleibende als dessen Nachricht behandelt"

ALAN KAY: The Early History of Smalltalk (1993)

Klassen & Objekte & Instanzen

Klasse
ist die Definition der Attribute, Operationen und der Semantik für eine Menge von gleichartigen Objekten

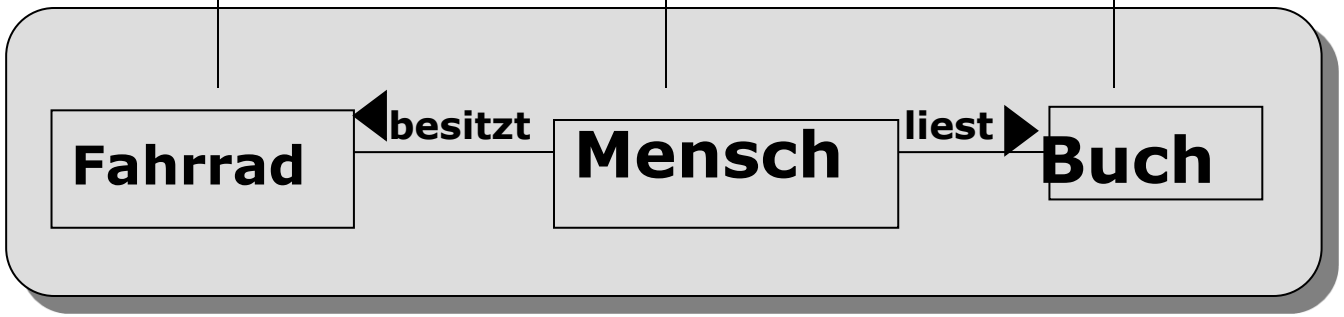
Objekt
ist eine konkret zur Ausführungszeit vorhandene und agierende Einheit mit eigener Identität und für ihre Instanzvariablen Speicher allokierende Instanz, die sich entsprechend dem Protokoll ihrer Klasse verhält

Instanz
Für den Hausgebrauch können Instanz, Objekt und Exemplar synonym betrachtet werden

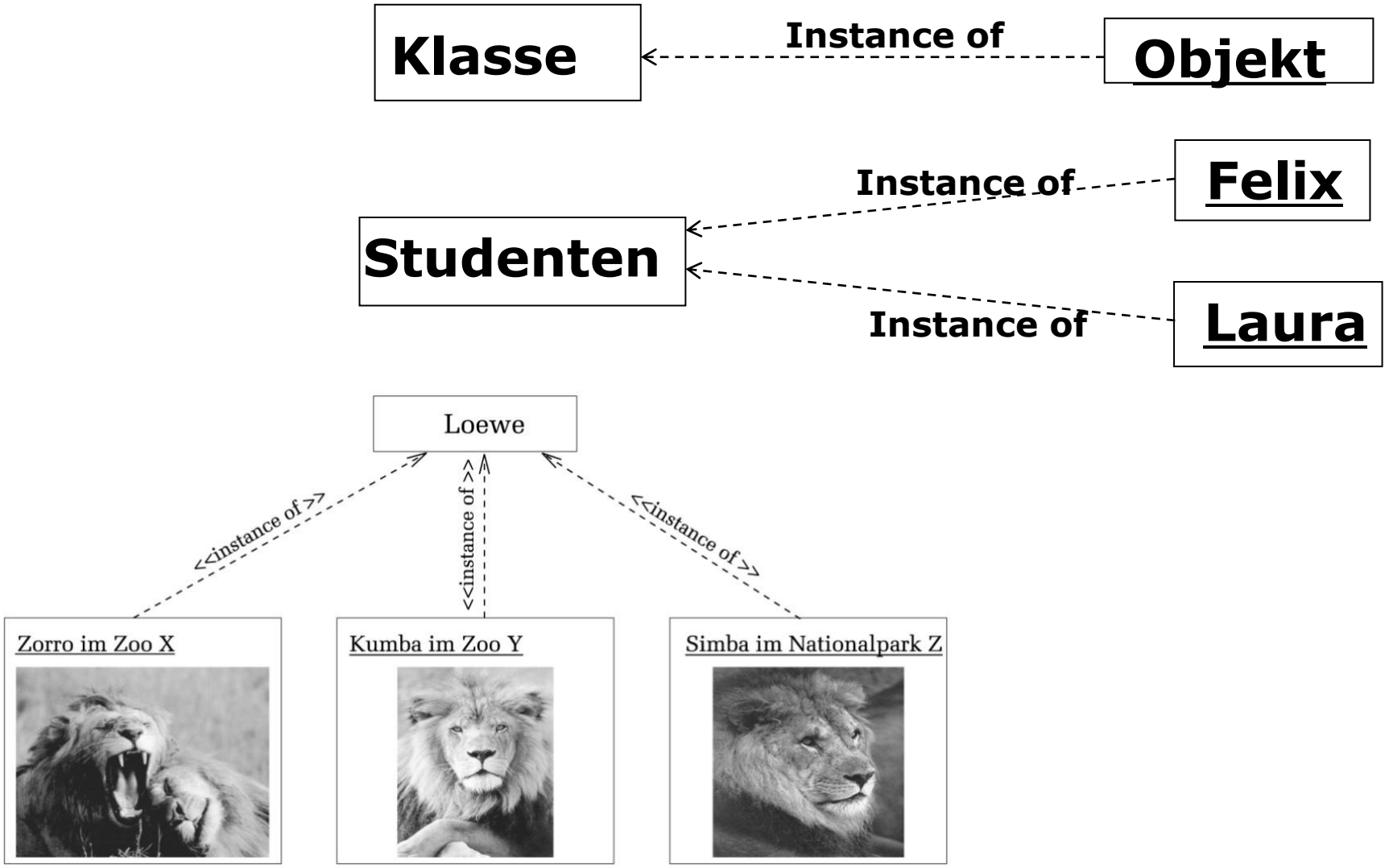
Realität – Modell (Beziehungen)

Realität

Modell



Objekt-Klassen-Beziehung



Objekt´-Orientierung

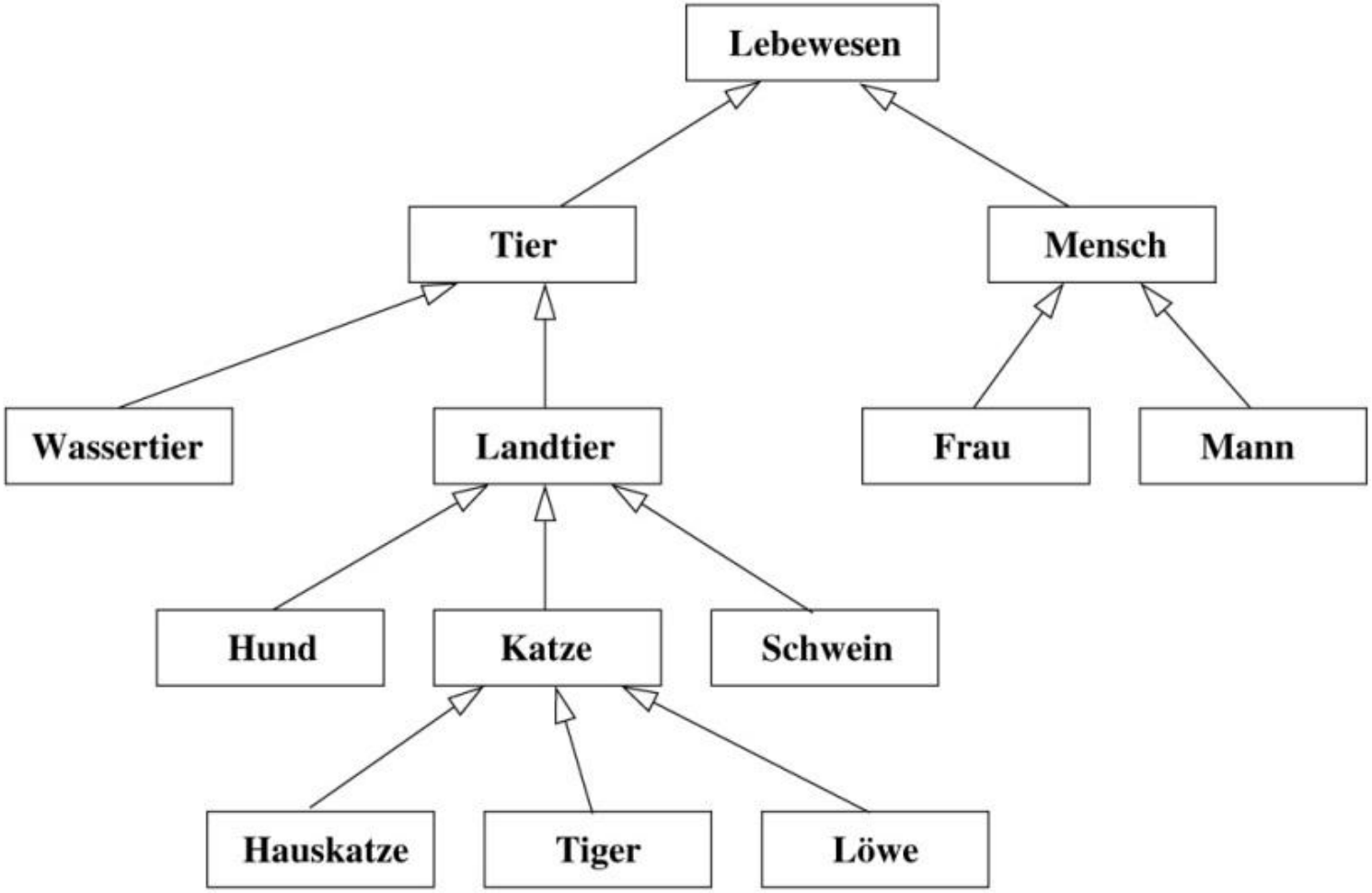
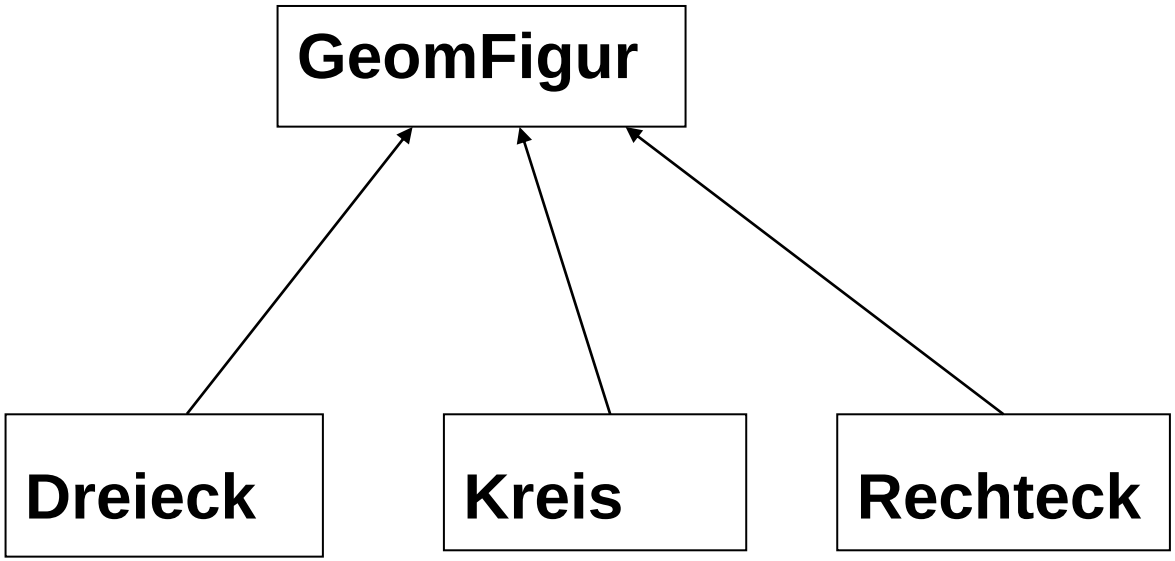


Abbildung 7.52: Mögliche Einteilung von Lebewesen (Ausschnitt)

Vererbung

Basisklasse

abgeleitete Klasse



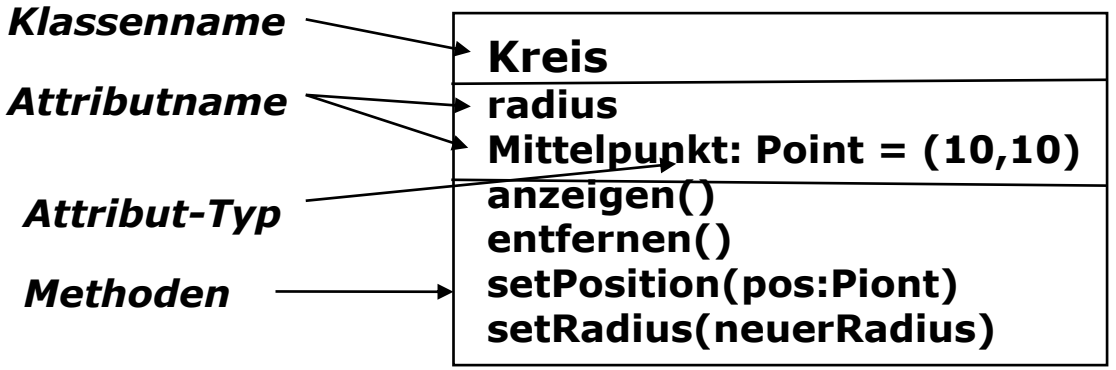
Attribute & Methoden

Attribute

die Struktur der Objekte: die in ihnen enthaltenen Informationen bzw. Daten -> “Variablen”

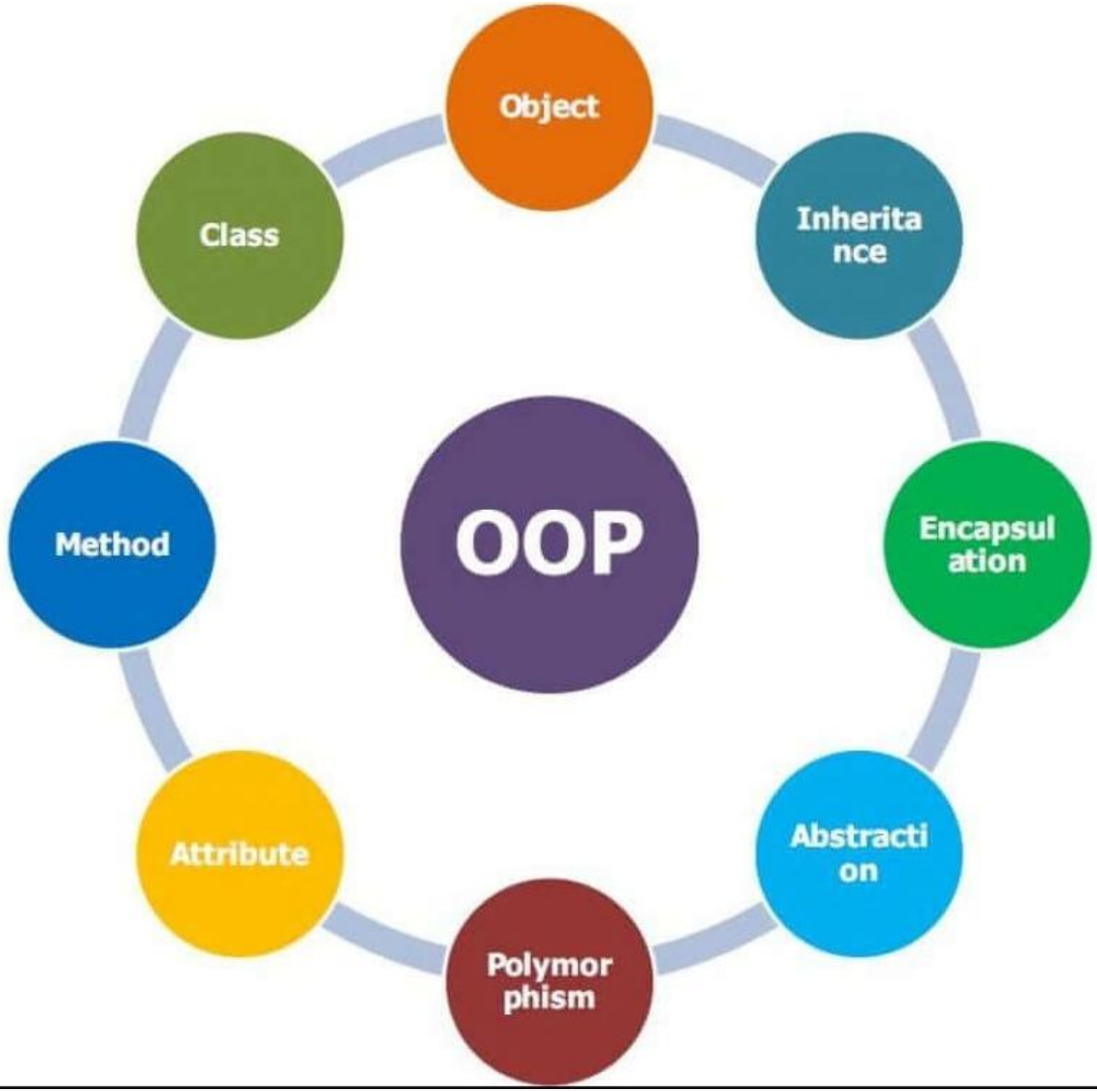
Methode (Operation)

Implementierung einer Operation -> “Funktionen”



OOP - Video

Objektc-oriented Progarmming in 7 minutes


Object
Inherita nce
Encapsul ation
Abstracti on
Polymor phism
Attribute
Method
Class
OOP

Stefan Mischook

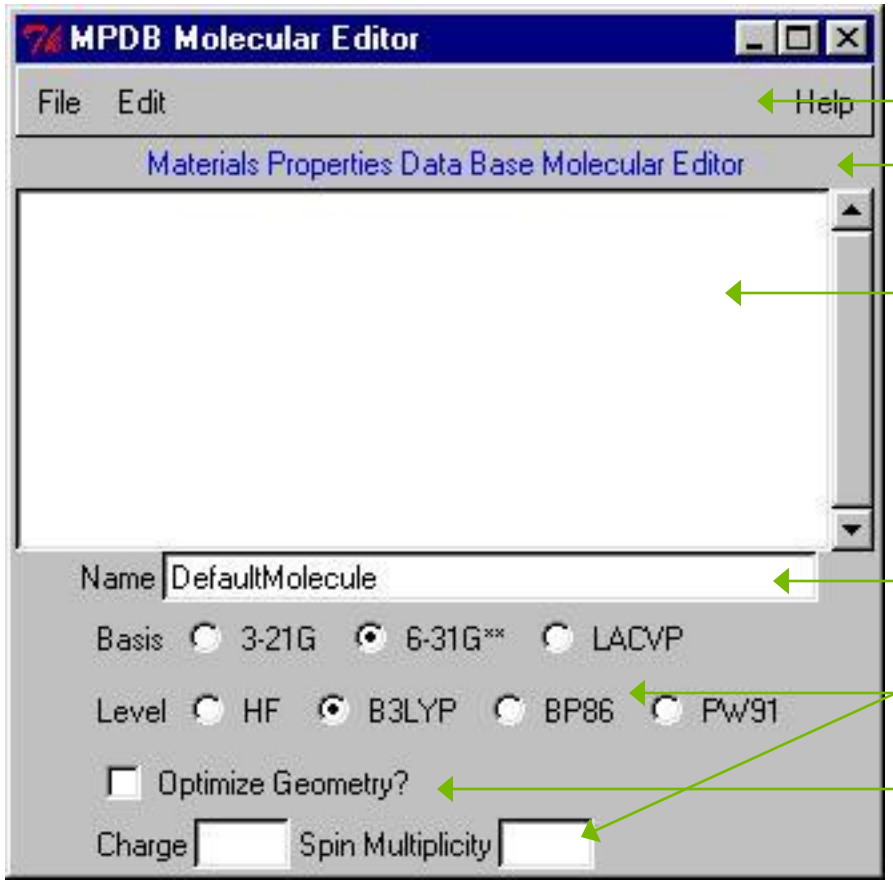
Why was OOP invented?

Why OOP inheritance sucks

Stefan Mischook

Why OOP inheritance sucks

Beispiele für GUI-Elemente



Menu bar

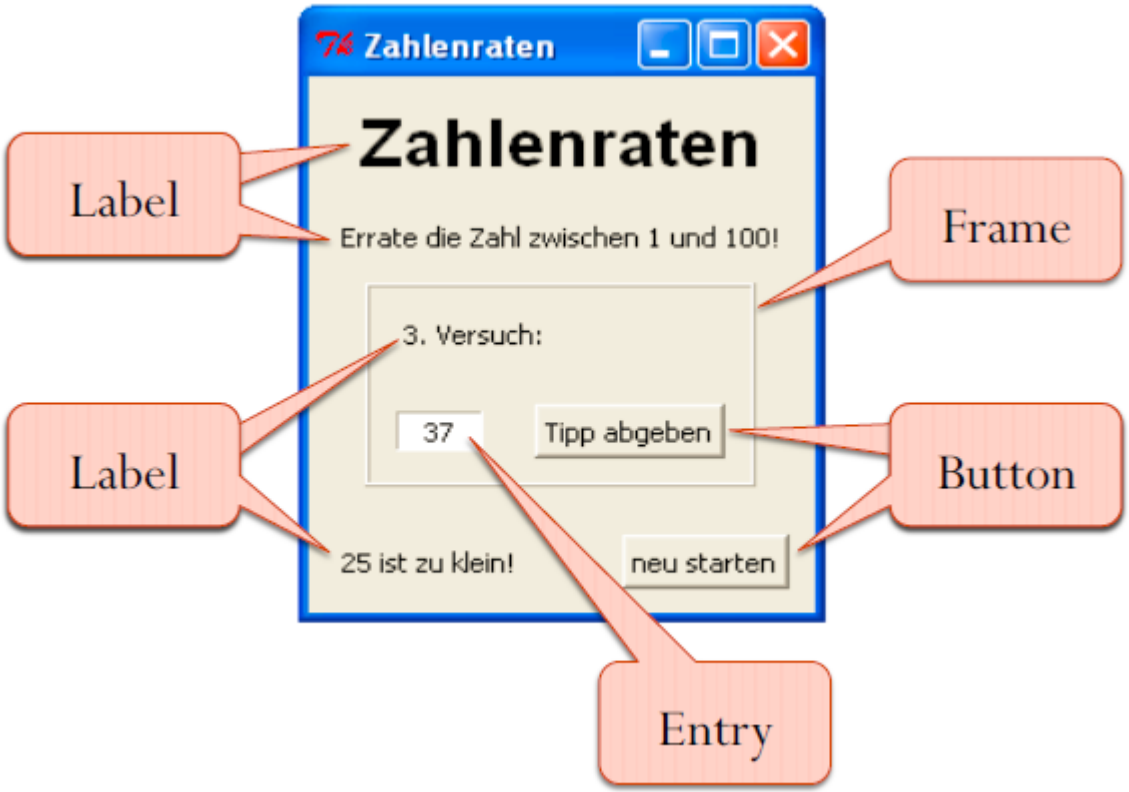
Label

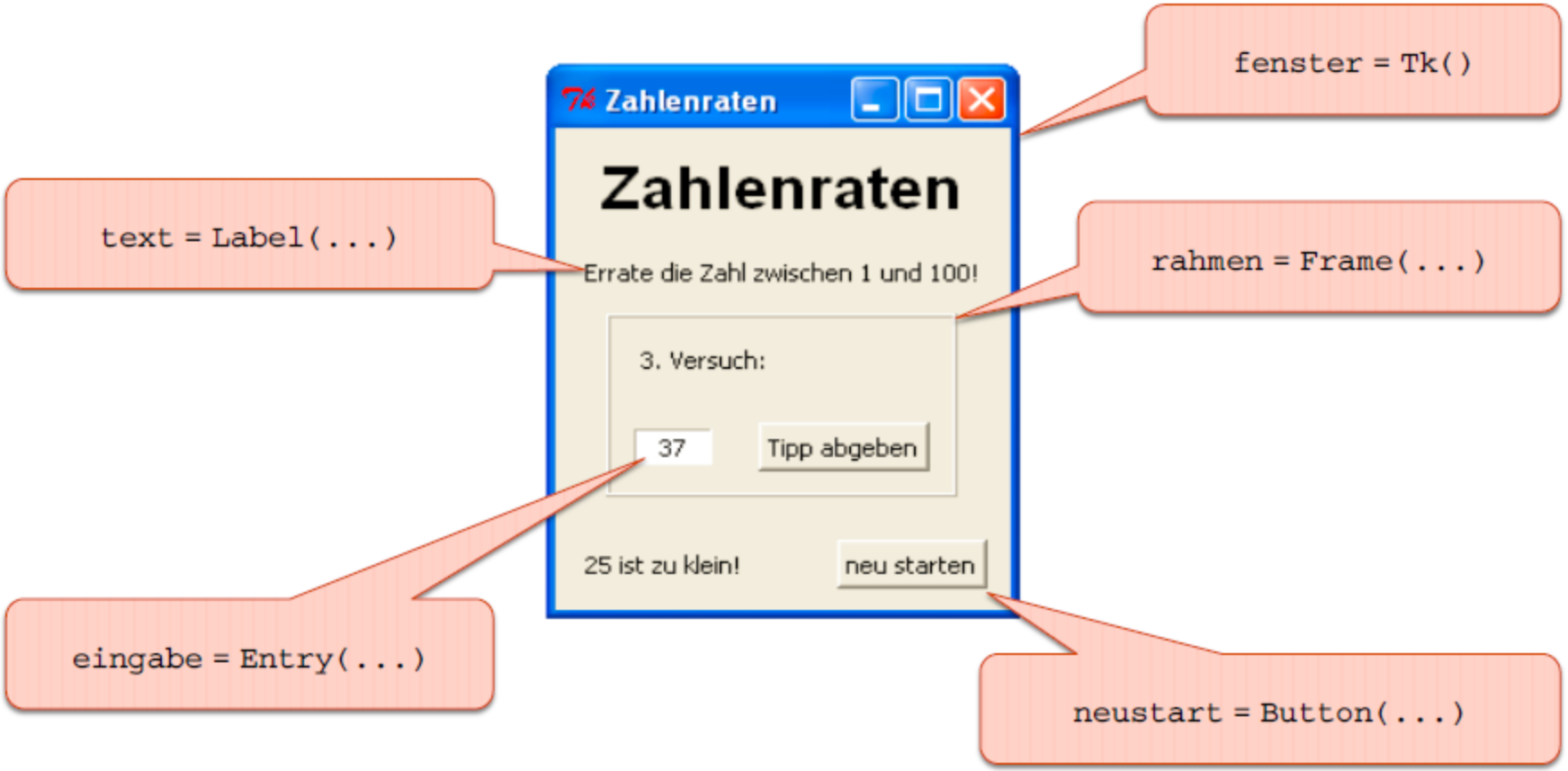
Text area (for geometry input)

Text entry

Radio buttons

Checkbox


Zahlenraten
Zahlenraten
Errate die Zahl zwischen 1 und 100!
3. Versuch:
37
Tipp abgeben
25 ist zu klein!
neu starten
Label
Label
Frame
Button
Entry


fenster = Tk()
Zahlenraten
Zahlenraten
text = Label(...)
Errate die Zahl zwischen 1 und 100!
rahmen = Frame(...)
3. Versuch:
37
Tipp abgeben
25 ist zu klein!
neu starten
eingabe = Entry(...)
neustart = Button(...)

Frame

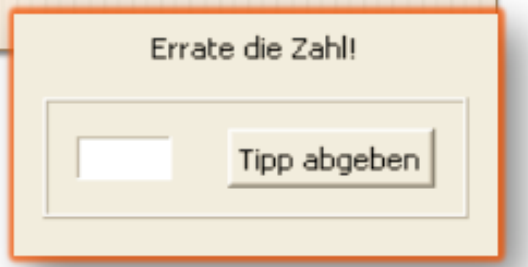
Frames dienen zum Gruppieren von Widgets.

```
rahmen = Frame(master[, option1=wert1[, ...]])
```

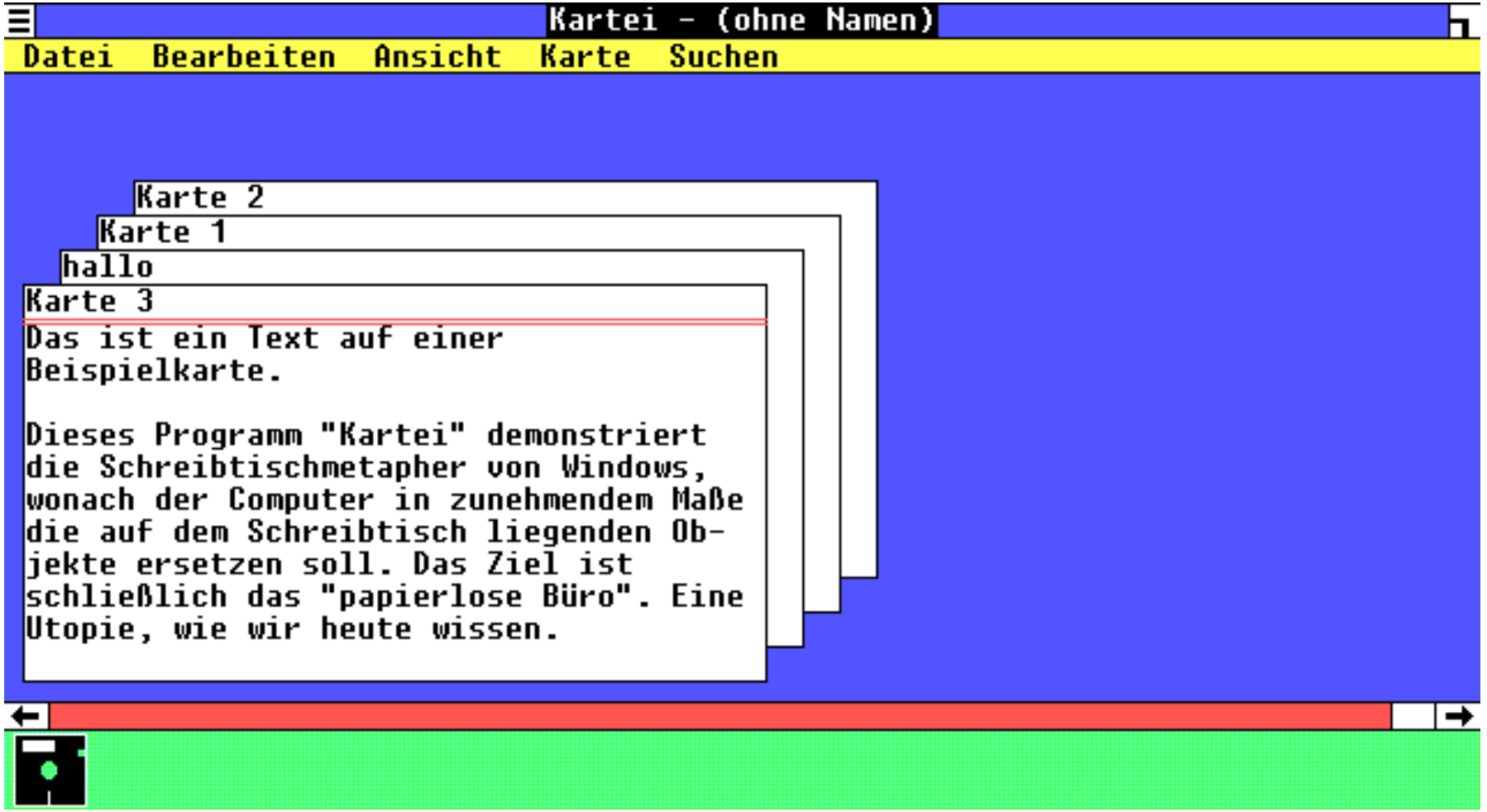
```
anleitung = Label(fenster, text='Errate die Zahl!')
rahmen = Frame(fenster, relief=RIDGE, bd=2)
eingabe = Entry(rahmen, width=5, justify=CENTER)
button = Button(rahmen, text='Tipp abgeben')
```

Der Name des Frames wird als master der untergeordneten Widgets angegeben.

In der Voreinstellung sind Frames unsichtbar.



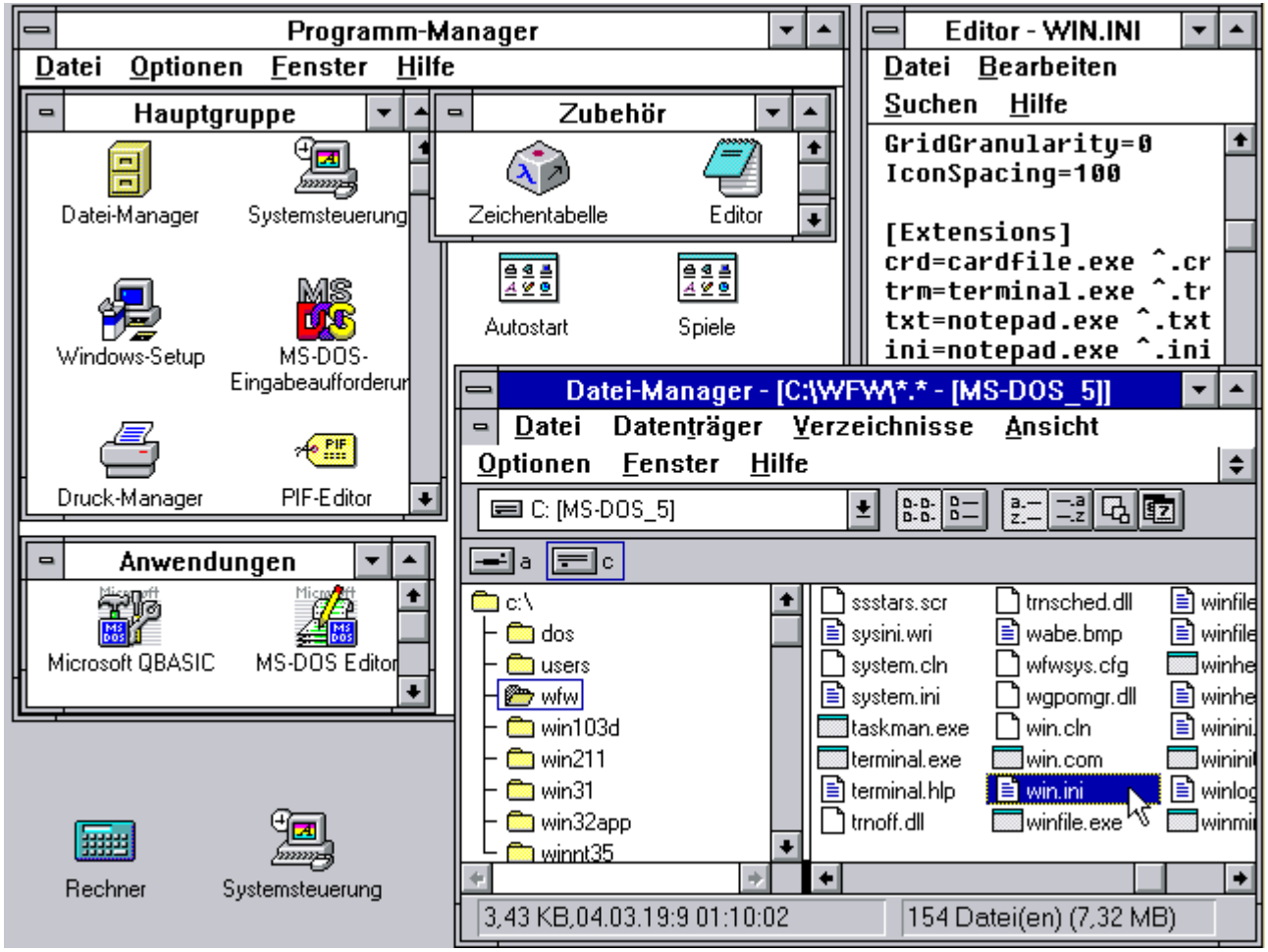
Beispiele von GUI‘s
-Windows 1.x



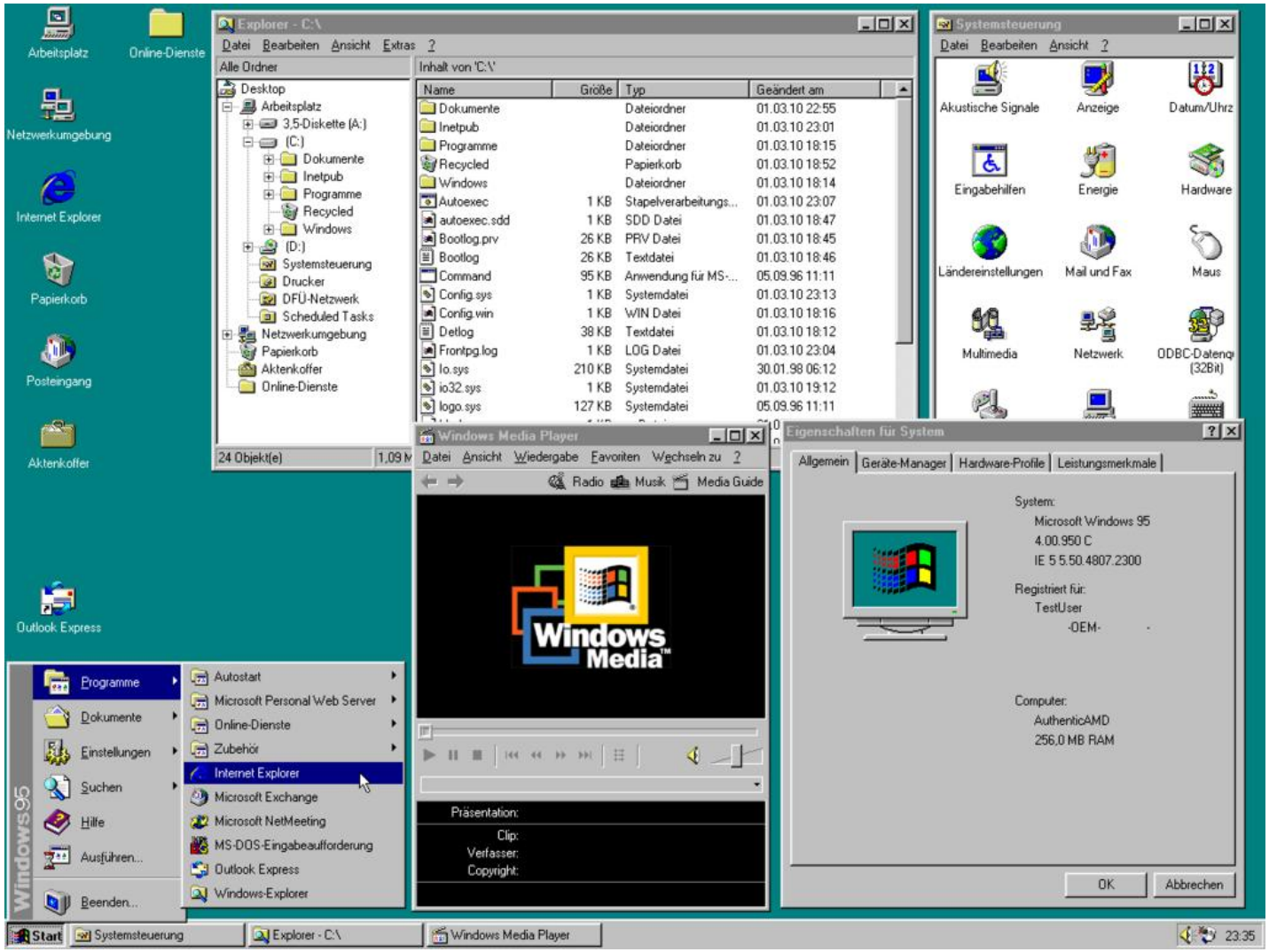
Beispiele von GUI‘s
-TOS1.0 (1985) auf Atari 1040ST

Beispiele von GUI‘s
-AmigaOS 1.3 (1987) auf Amiga 500

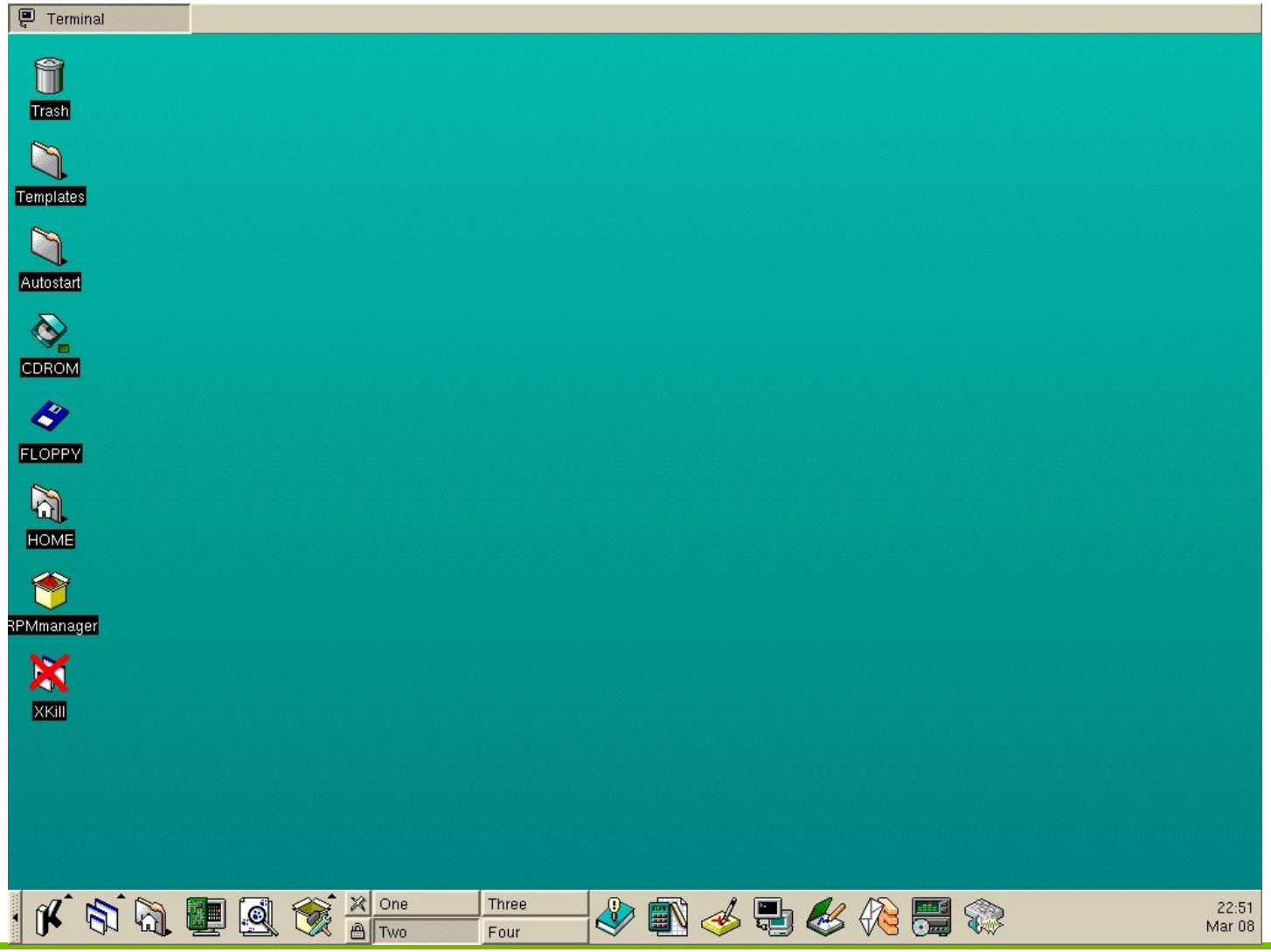
Beispiele von GUI's
-Windows 3.1 (1992)



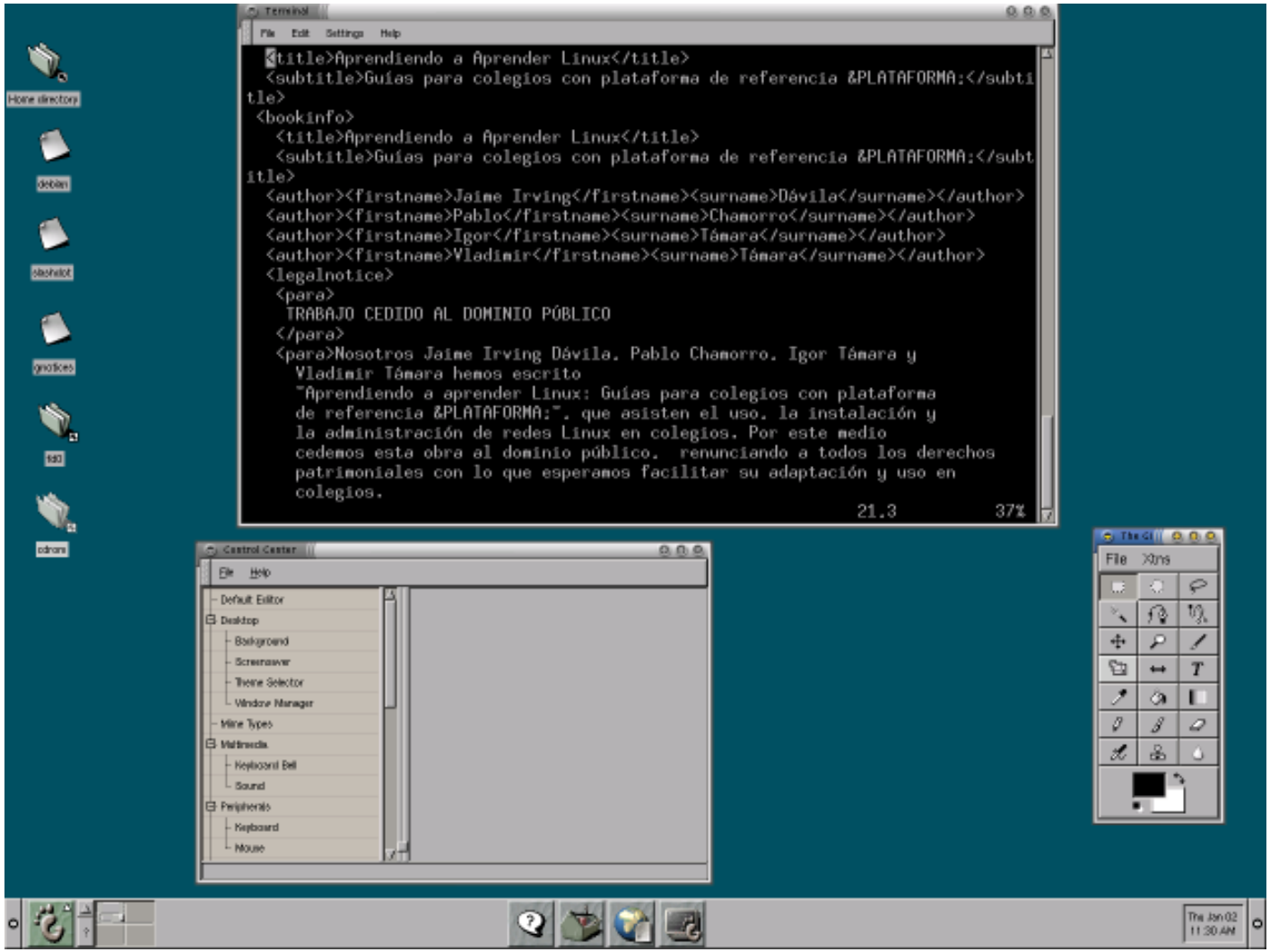
Beispiele von GUI's
-Windows 95 (1995)



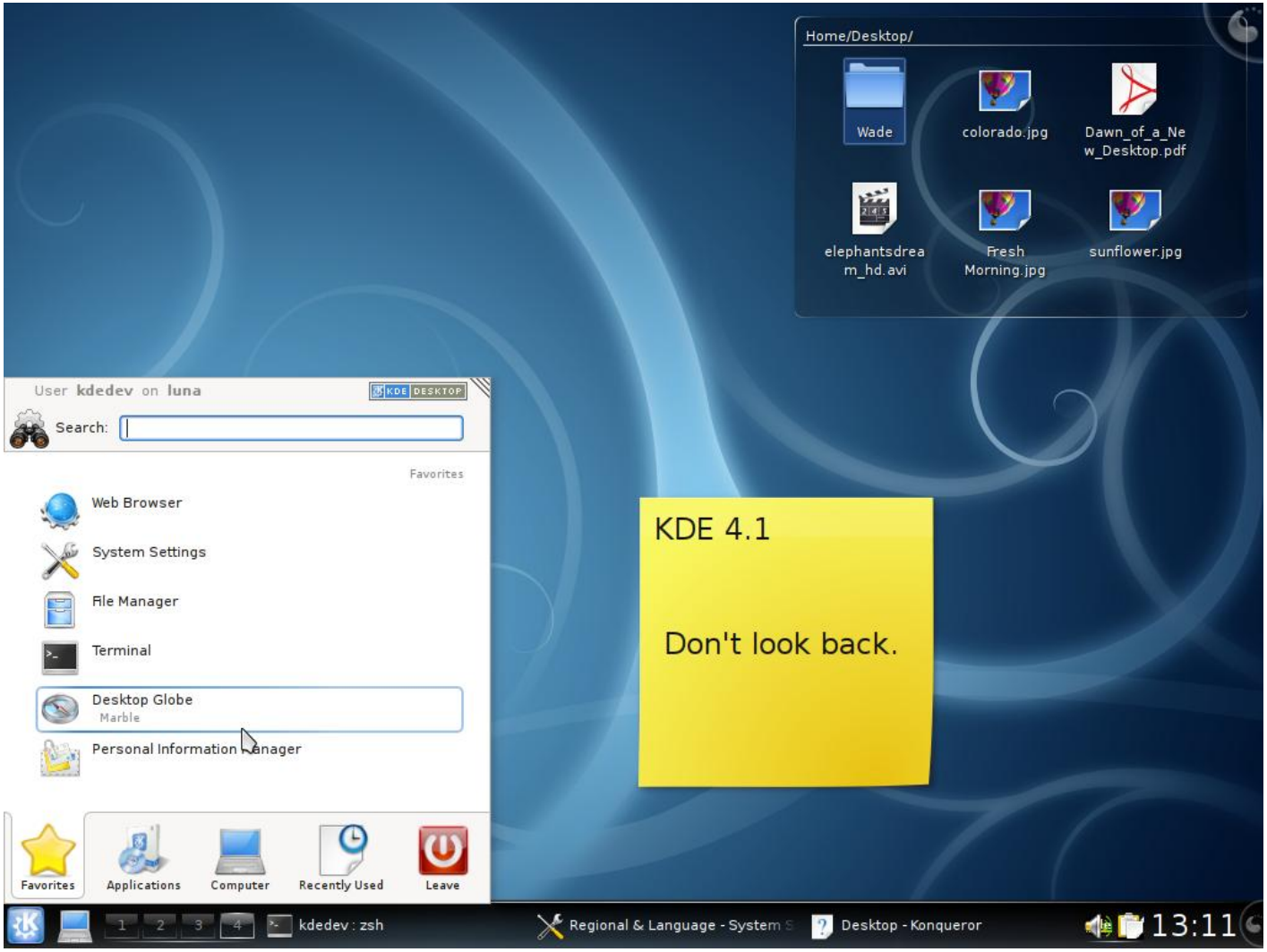
Beispiele von GUI‘s
-KD 1.0 (1998)



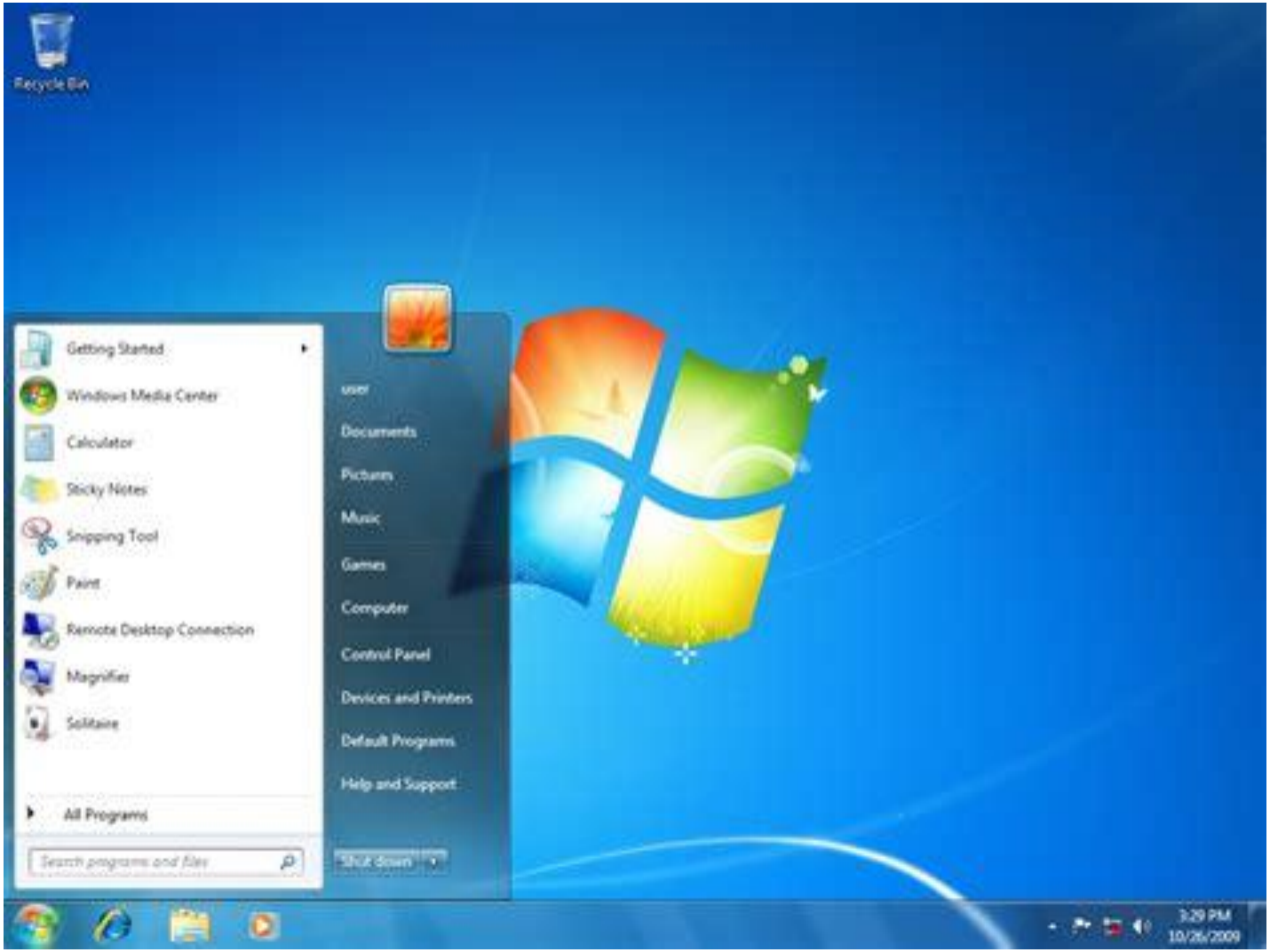
Beispiele von GUI's
-Gnome (1999)



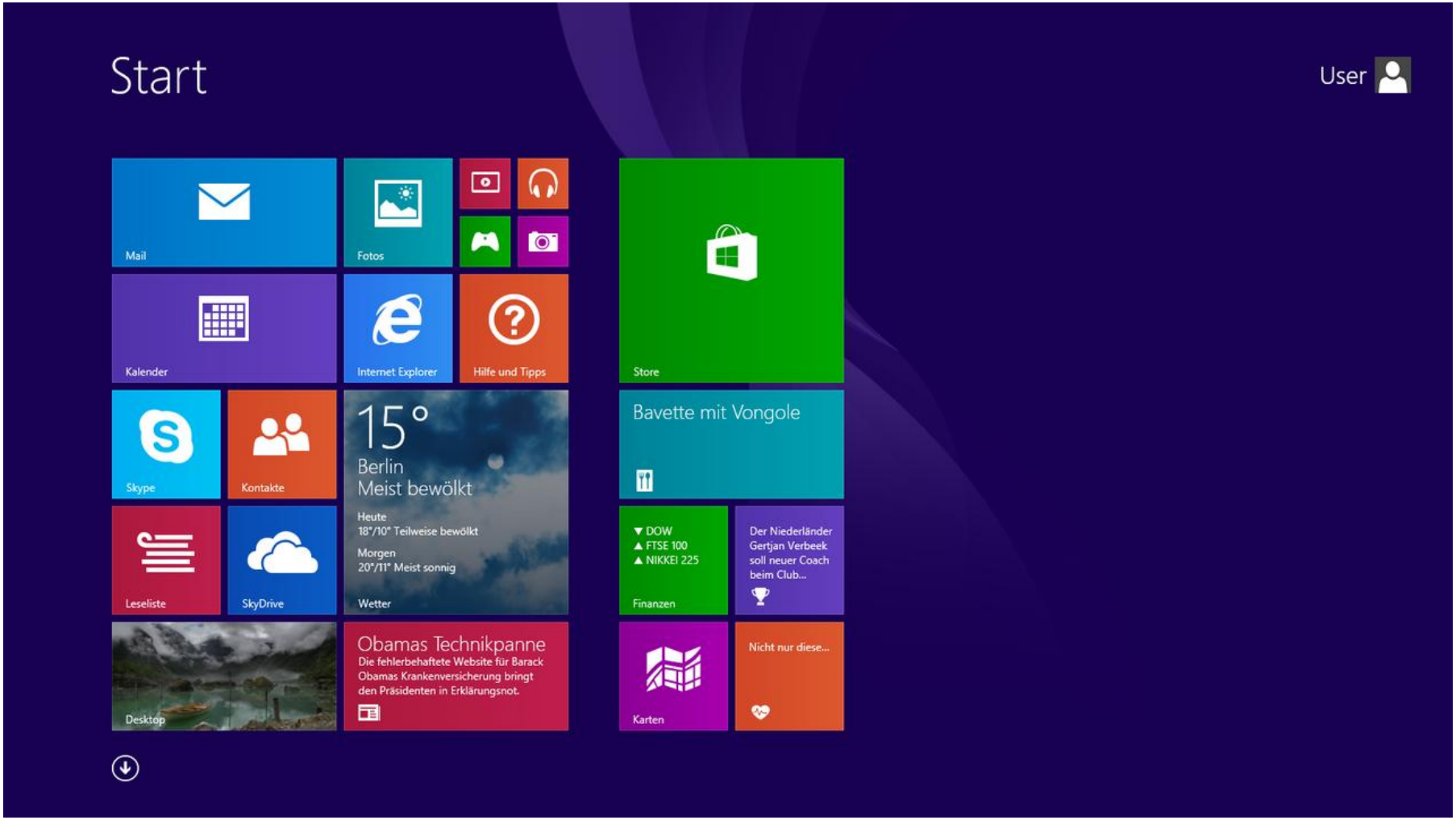
Beispiele von GUI's
-KDE 4.x (2008)



Beispiele von GUI's -Windows 7 (2009)



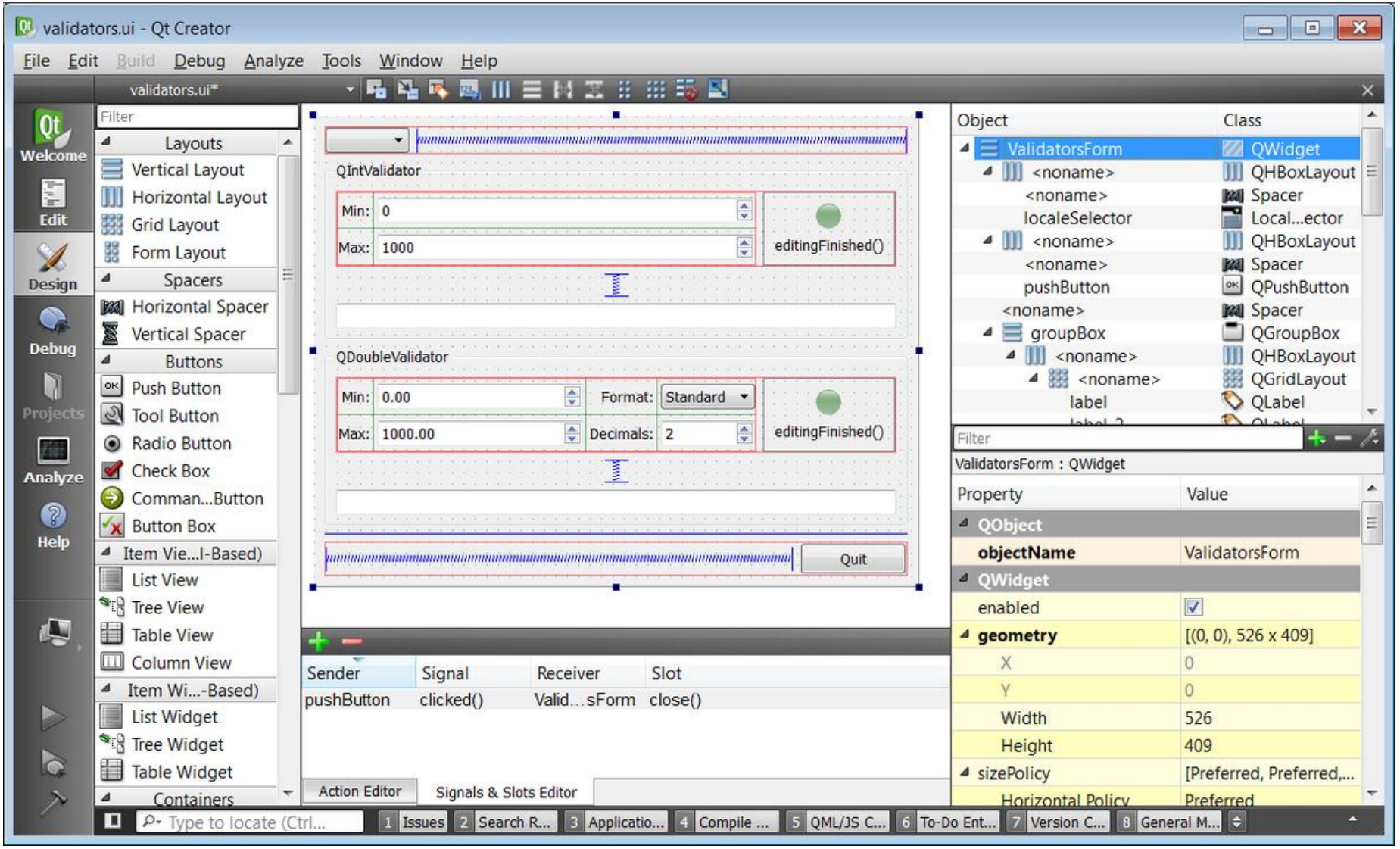
Beispiele von GUI‘s -Windows 8.1



GUI Toolkits

- Ansammlung an Werkzeugen die das erstellen von GUIs vereinfachen
- Programmbibliothek für das Erstellen von Desktopanwendungen
- Stellt einen Satz an Eingabe, Ausgabe und Steuerelemente zur Verfügung
- Mittels Drag&Drop können Elemente platziert werden

GUI Toolkits - Bsp QT Creator



Programming Language Rankings August 2020

- https://www.youtube.com/watch?v=KMz-HUFqvVk

The Healthy Software Developer

- https://www.youtube.com/channel/UCfe_znKY1ukrqlGActlFmaQ

Hochschule für Technik
und Wirtschaft Berlin

University of Applied Sciences

Danke für eure Aufmerksamkeit ☺

jungerdm@htw-berlin.de