Informatik Klasse 13, Foliensatz 9
Zeichenfeld-Widget (Canvas)

Prof. G. Kemnitz

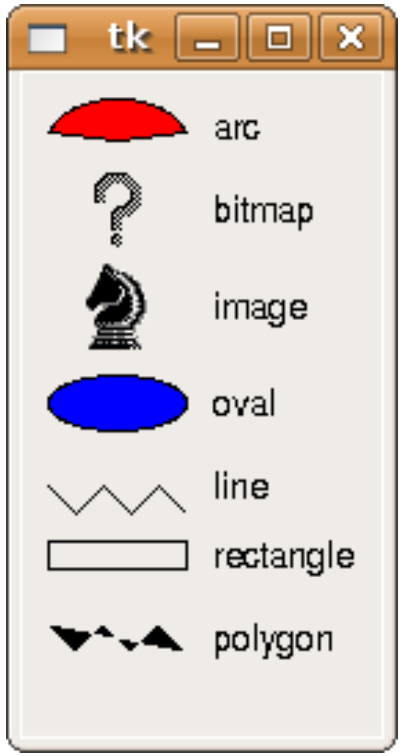
Institut für Informatik, Technische Universität Clausthal

3. Dezember 2009

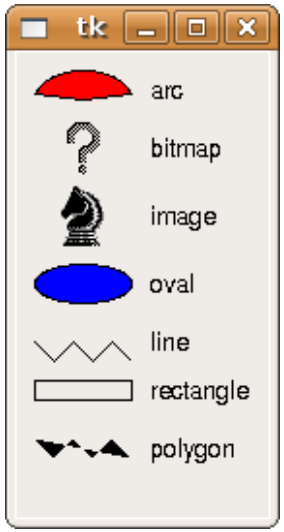
Canvas

Widget zur Anzeige und zur Bearbeitung von Graphik-Elementen:

- Bogen (arc), durch einen Bogen und eine Gerade begrenzte Flächen (chord), Tortenstück (pieslice)
- zweifarbige Pixelgraphik (bitmap)
- Bildobjekt (image)
- Linie (line)
- Kreis, elypse (oval)
- Polygon (polygon)
- Rechteck (rectange)
- Text (text)
- andere Widgets (windows).

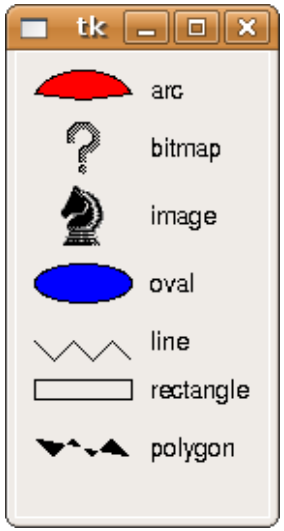


Experiment: Erzeugung einer Oberfläche mit Graphikelementen




```
from Tkinter import *
root=Tk()
c=Canvas(root, width=130, height=240)
# Erzeugung der Graphikelemente
c.create_arc(10,10,60,40, start=10, extent=160, fill="red")
c.create_bitmap(35, 50, bitmap="question")
photo=PhotoImage(file="../P8/ichess.gif")
c.create_image(35, 85, image=photo)
c.create_oval(10, 110, 60, 130, fill="blue")
c.create_line(10,150,20,160,30,150,40,160,50,150,60,160)
c.create_rectangle(10, 170, 60, 180)
c.create_polygon(10,200,20,210,30,200,40,210,50,200,60,210)
```

Erzeugung der Texte




```
# Erzeugung der Texte daneben
c.create_text(70, 20, text="arc", anchor=W)
c.create_text(70, 50, text="bitmap",anchor=W)
c.create_text(70, 85, text="image", anchor=W)
c.create_text(70, 120, text="oval", anchor=W)
c.create_text(70, 150, text="line", anchor=W)
c.create_text(70, 175, text="rectangle", anchor=W)
c.create_text(70, 205, text="polygon", anchor=W)
c.pack()
root.mainloop()
```

Konstruktionsparameter

- Koordinaten: x,y-Wertepaare; Ursprung obere linke Canvas-Ecke, x zählt nach rechts und y nach unten
  - ein Koordinatenpaar: Ankerpunkt (bitmap, image, text)
  - zwei Koordinatenpaare: umschließendes Recheck (arc, oval, rectangle)
  - variable Anzahl: Punkte eines Linienzugs oder Polygons
- Attribute, abhängig von der Art des Graphikelements
  - tag: Zuordnung von Gruppenbezeichnern (Tupel von Strings), über die später Bearbeitungsmethoden zusammengehörige (gleich zu bearbeitende) Elemente finden
  - anchor: Ausrichtung zum Ankerpunkt für bitmap, image, text; Werte: N, NE, E, SE, S, SW, W, NW (Himmelsrichtungen, ohne Angabe CENTER)
  - fill: Füllfarbe für Flächen (arc, oval, rectange, polygon)
  - ...

Methoden

Der Konstruktor liefert einen Objektzeiger, auf denen zahlreiche Methoden angewendet werden können:

- Koordinaten lesen und verändern
- Attribute lesen und verändern
- Ereignisse binden
- Stapelreihenfolge ändern (Graphikobjekte werden übereinandergestapelt, bei Verdeckung ist das oberste sichtbar.

Achtung, diese Methoden werden auf das Canvas-Objekt angewendet:

canvas.methode(Objektzeiger, weitere_Parameter)
→ *Rückgabewert*

Objektzeiger des Elements, auf das die Mouse zeigt: CURRENT

Experiment: Tortenstücke verschieben

```
from Tkinter import *
def callback(evobj):
 dx=10; dy=20
 evobj.widget.move(CURRENT, dx, dy)
root=Tk() c=Canvas(root, width=500, height=300)
c.bind("<Button-1>", callback)
xy=(20, 20, 300, 180)
x1=c.create_arc(xy, start=0, extent=270, fill="red")
x2=c.create_arc(xy, start=270, extent=60, fill="blue")
x3=c.create_arc(xy, start=330, extent=30, fill="green")
c.pack()
root.mainloop()
```

- evobj.widget: Canvas-Objekt
- CURRENT: ausgewähltes Graphikobjekt
- dx, dy: Verschiebung in x- bzw. y-Richtung

Andere Methoden zum Ausprobieren

- löschen

  ```
  evobj.widget.delete(CURRENT)
  ```

- skalieren

  ```
  xscale=1.3; yscale=1.3, xoffst=3, yoffstet=7
  evobj.widget.scale(CURRENT, xscale, yscale, xoffset, yoffset)
  ```

- In der Vordergrund bzw. Hintergrund verschieben

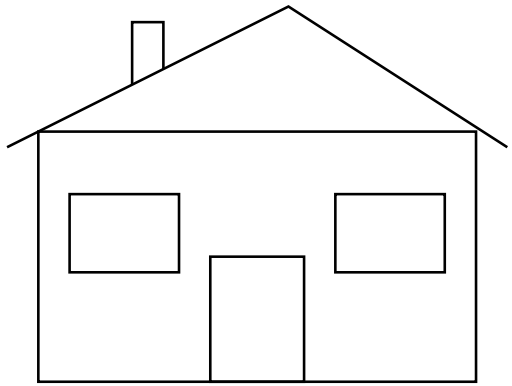
  ```
  evobj.widget.lift(CURRENT)
  evobj.widget.lower(CURRENT)
  ```

- Lesen und verändern der Koordinaten

  ```
  evobj.widget.coords(CURRENT) → aktuelle_Koordinaten
  evobj.widget.coords(CURRENT, neue_Koordinaten)
  evobj.widget.bbox(CURRENT) → Eckpunkte
  ```

Aufgabe 9.1: Vektorgraphik erzeugen

Scheiben Sie ein Programm, das in einem Canvas aus Rechtecken und Linien ein Haus mit einer Tür, zwei Fenstern, einem spitzen Dach und einem Schornstein erzeugt.



Aufgabe 9.2: Testprogramm für Bearbeitungsmethoden für Graphikobjekte

Schreiben Sie eine Oberfläche mit einem Zeichenfeld und zwei Texteingabefeldern. Das erste Eingabefeld soll zur Eingabe von Anweisungen zur Erzeugung neuer Graphikobjekte dienen, z.B.

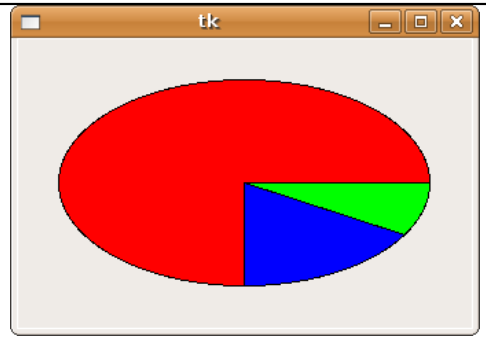
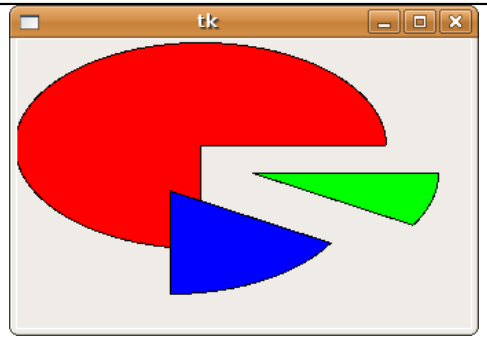
```
»c.create_arc(10, 15, 30, 45, start=270,
     extent=60, fill="blue")«
```

Dazu an diesen Feld ein Ereignis-Handler für die Betätigung der Enter-Taste zu binden, in dem der String mit exec(...) ausfühgeführt wird und dabei ein neues Graphik-Objekt erzeugt. Das andere Eingabefeld dient zur Eingabe der Methode, die im Ereignishandler für Mausklicks im Canvas mit dem ausgewählten Graphikobjekt mit exec(...) auszuführen ist, z.B.

```
»evobj.widget.scale(CURRENT, 0.4, 0,6, 0, 0)«
»print.evobj.widget.coords(CURRENT)«
```

Aufgabe 9.3: Graphik-Objekte mit der Maus verschieben

Schreiben Sie ein Programm mit einem Zeichenfeld und drei »Kuchenscheiben«, indem mit der linken Maus-Taste eine Kuchenscheibe auswählt und mit gedrückter Maus-Taste bewegt werden kann.

Oberfläche nach Programmstart	Oberfläche nach Verschieben der Kuchenstücke
	

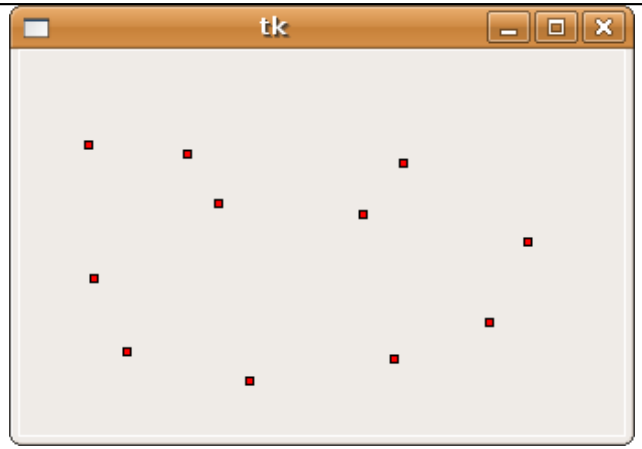
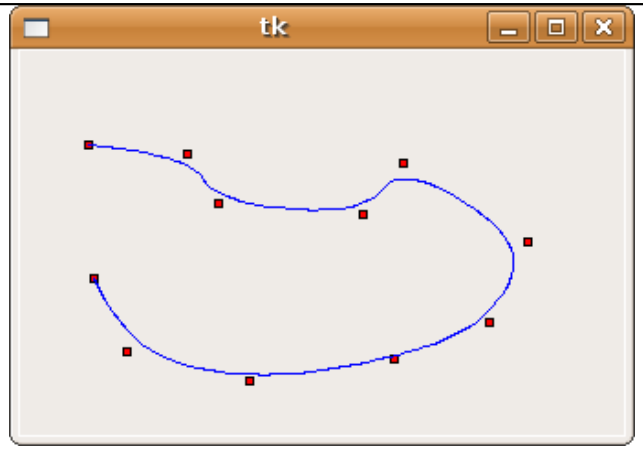
Hinweise: Die Lösung erfordert zwei Ereignis-Handler. Bei einem <Button-1>-Ereignis ist die Cursor-Position in einer globalen Variablen oder einem Attribut der Klasse zu speichern. Bei einem <B1-Motion>-Ereignis ist das ausgewählte Objekt »CURRENT« mit der Canvas-Methode

```
Canvas_Objekt.move(CURRENT, dx, dy)
```

(dx, dy – x- bzw. y-Verschiebung) um die Differenz zwischen der aktuellen und der gespeicherten Cursor-Position zu verschieben. Anschließend ist die gespeicherte Position gleich der aktuellen Mauszeiger-Position zu setzen.

Aufgabe 9.4: Linienzug mit der Maus eingeben

Schreiben Sie ein Programm, in dem Sie mit der linken Maus-Taste kleine Rechtecke auf der Zeichenfläche platzieren können, durch die bei der Betätigung der rechten Maus-Taste ein Spline gelegt wird.

nach 10 <Button-1> Ereignissen	nach dem nachfolgenden <Button-3>-Ereignis
	

Hinweise:
Schreiben Sie die Applikation als Klasse mit einer zu Beginn leeren Liste als Attribut. Der Ereignis-Handler für <Button-1>, der die Quadrate an die aktuelle Maus-Position zeichnet, soll zusätzlich die aktuellen x- und y-Mauszeiger-Koordinaten an die Liste anhängen. Beispieltestausgabe der Liste mit »print self.l«:

```
[126, 63]
[126, 63, 202, 98]
[126, 63, 202, 98, 213, 140]
[126, 63, 202, 98, 213, 140, 91, 130]
```

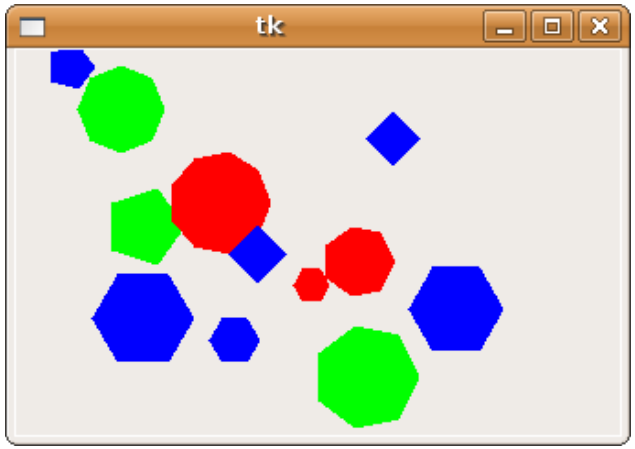
Der Ereignis-Handler für <Button-3> soll das Linien-Objekt mit den Geometrie-Daten aus der Liste und den Attributen »fill="blue"« (Linienfarbe) und "smooth=True" (Spline-Approximation) zeichnen

```
Canvas_Objekt.create_line(self.l, fill="blue",
  smooth="True")
```

und die Liste mit den Koordinaten wieder leeren.

Aufgabe 9.5: Zufallspolgone

Schreiben Sie ein Programm mit einem Zeichenfeld als Oberfläche, das bei einem Maus-Klick ein regelmäßiges Polygon mit dem Maus-Zeiger als Mittelpunkt, einer zufällig ausgewählten Eckenanzahl n im Bereich von 3 bis 10, einem zufälligen Radius r im Bereich von 10 bis 30 Pixel, einer zufälligen Verdrehung α im Bereich von Null bis 2π und einer zufälligen Füllfarbe aus der Menge "red", "green" und "blue" auf dem Zeichenfeld hinzufügt.



Hinweise:

- Die n Eckpunkte eines regelmäßigen Polygons liegen auf einem Kreisbogen mit dem Radius r um den Mittelpunkt (m_x, m_y). Berechnungsvorschrift:

$$\begin{aligned} x_i &= m_x + r \cdot \cos\left(i \cdot \frac{2\pi}{n} + \alpha\right) \\ y_i &= m_y + r \cdot \sin\left(i \cdot \frac{2\pi}{n} + \alpha\right) \end{aligned}$$

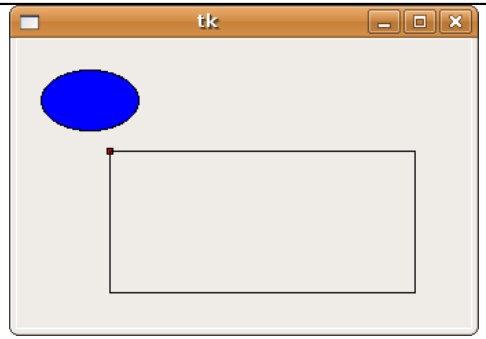
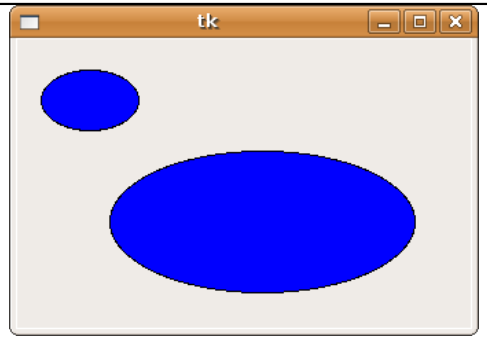
- Schreiben Sie eine Hilfsfunktion mit den Aufrufparametern n, m_x, m_y, r und α, die die Eckpunktkoordianten als Liste der Form

$$[x_0, y_0, x_1, y_1, \ldots x_{n-1}, y_{n-1}]$$

  zurückgibt.
- Die benötigten Zufallsfunktionen stehen im Modul »random«: »choise(<Tupel der Farbwerte>)« für die Farbauswahl und »randint(min, max)« für das Auswürfeln der Zahlenwerte.

Aufgabe 9.6: »Oval« zeichnen

Schreiben Sie ein Programm, mit dem man auf einem Zeichenfeld Graphik-Objekte vom Typ »Oval« erzeugen kann.

vor Losslassen der Maustaste	nach Loslassen der Maustaste
	

- Bei einem Maus-Klick an der Startposition ist ein Quadrat mit der Kantenlänge 5×5 Pixel, Füllfarbe rot, und ein Begrenzungsrechteck mit zwei übereinstimmenden Begrenzungspunkten zu zeichnen

- Der Ereignis-Handler für <B1-Motion> soll den zweiten Begrenzungspunkt des Begrenzungsrechtecks mit der Canvas-Methode:

  ```
  canvas.coords(gobj, neue_Koordinaten)
  ```

  (canvas – Canvas-Objekt; gobj – ID oder Tag des Graphikobjekts, dessen Koordinaten verändert werden) gleich der aktuellen Mauszeiger-Position setzen. Die Methode »coords« verlangt ganzahlige Koordinatenwerte und akzeptiert keine Listen oder Tupel.
- Bei Loslassen der Maustaste ist ein blaues »Oval« mit den Begrenzungspunkten des Rechtecks zu zeichnen. Das Quadrat und das Begrenzungsrechteck sind mit der Methode

  ```
  canvas.delete(gobj)
  ```

  zu löschen.