TKInter

- Es gibt APIs = Application Programming Interface
- für GUIs = Graphical Use Interface
- in Python gibt es tkinter (relativ einfach)
- alternativ aber komplizierter sind PyGtk, PyQT, …
- unser Ziel: Visualisierung unserer geometrischen Objekte.

Fenster

Fenster

1. tkinter wird importiert

Fenster

1. tkinter wird importiert

2. mit Hife des Tk()-Konstruktors wird ein Wurzelobjekt erzeugt.

- Diese Wurzelobjekt ist ein Fenster.
- Zu diesem Objekt können weitere Bestandteile zugefügt werden.

2. mit Hife des Tk()-Konstruktors wird ein Wurzelobjekt erzeugt.

- Diese Wurzelobjekt ist ein Fenster.
- Zu diesem Objekt können weitere Bestandteile zugefügt werden.

3. `lab` repräsentiert ein Label-Widget.

- Das Label wird dem Wurzelelement hinzugefügt.
- Widget = rechteckige Fläche auf dem Bildschirm mit einer Funktionalität
- Label-Widget kann nur Text anzeigen.

3. `lab` repräsentiert ein Label-Widget.

- Das Label wird dem Wurzelelement hinzugefügt.
- Widget = rechteckige Fläche auf dem Bildschirm mit einer Funktionalität
- Label-Widget kann nur Text anzeigen.

4. Anordnung des Widgets im Fenster

- Vorgefertigte

4. Anordnung des Widgets im Fenster

- Vorgefertigte

Canvas

1. Canvas dem Fenster

- Canvas ist ein Widget
- Canvas = Leinwand
- Auf Canvas können geometrische Figuren gemalt werden.
- Der Konstruktor `tk.Canvas()` übernimmt als Parameter Breite und Höhe in Pixeln

Canvas

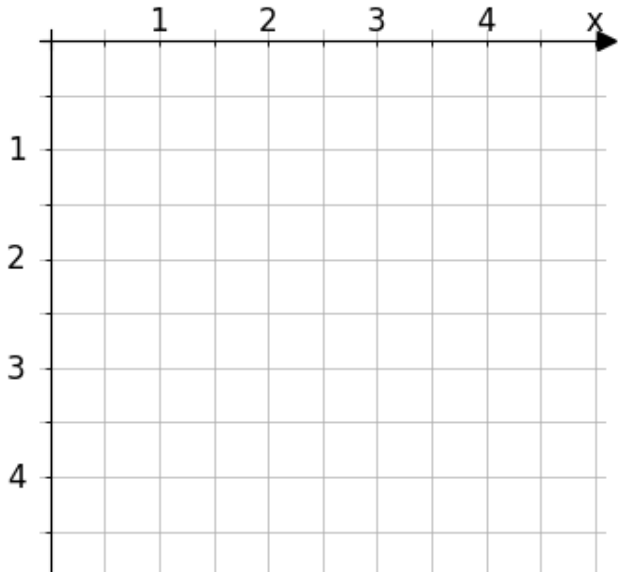
1. Canvas dem Fenster

- Canvas ist ein Widget
- Canvas = Leinwand
- Auf Canvas können geometrische Figuren gemalt werden.
- Der Konstruktor `tk.Canvas()` übernimmt als Parameter Breite und Höhe in Pixeln

2. Auf dem Canvas malen

2. Auf dem Canvas malen

Graphikkoordinaten



Canvas Methoden

- Linie von (x1, y1) nach (x2, y2)
  `canvas.create_line(x1, y1, x2, y2, **options)`
- Rechteck von obere linke Ecke (x1, y1) nach untere rechte Ecke (x2, y2)
  `canvas.create_rectangle(x1, y1, x2, y2, **options)`
- Oval innerhalb des Rechtecks gebildet durch obere linke Ecke (x1, y1) und untere rechte Ecke (x2, y2)
  `canvas.create_oval(x1, y1, x2, y2, **options)`

=> Alle `create`-Methoden liefern den Index des erzeugten Objekts
= Eindeutige Referenz auf das Objekt. - Damit kann das Objekt noch nachträglich geändert werden.

Canvas Methoden

- Linie von (x1, y1) nach (x2, y2)
  `canvas.create_line(x1, y1, x2, y2, **options)`
- Rechteck von obere linke Ecke (x1, y1) nach untere rechte Ecke (x2, y2)
  `canvas.create_rectangle(x1, y1, x2, y2, **options)`
- Oval innerhalb des Rechtecks gebildet durch obere linke Ecke (x1, y1) und untere rechte Ecke (x2, y2)
  `canvas.create_oval(x1, y1, x2, y2, **options)`

=> Alle `create`-Methoden liefern den Index des erzeugten Objekts
= Eindeutige Referenz auf das Objekt. - Damit kann das Objekt noch nachträglich geändert werden.

- `canvas.delete(i)` löscht Objekt mit dem Index i
- `canvas.move(i, deltax, deltay)` bewegt Objekt i um deltax und deltay
- `canvas.update()` erneuerte die Darstellung des Bildschirms
- Referenz unter www.tkdocs.com