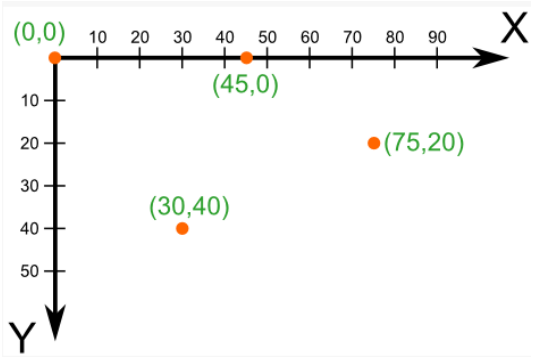
Урок 2. Рисование на холсте

Чтобы рисовать картинки в Python, нужно создать чистую область для рисования, которую называют холстом. Для указания, в каком месте холста рисовать, используются координаты X и Y.



0. Предварительная подготовка.

Начинаем новый проект.
Сначала импортируем библиотеку tkinter, на ней создаем холст размером 200*200 белого цвета.

canvas01.py - C:/Users/Adm-klass/Desktop/canvas01.py (3.7.4)

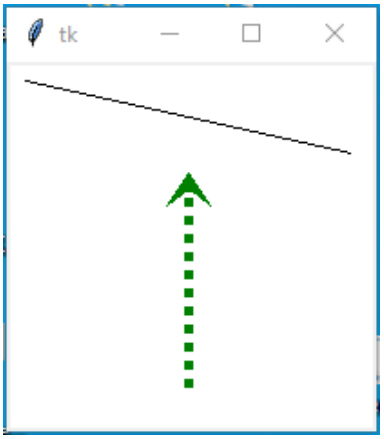
File Edit Format Run Options Window Help

```
from tkinter import *
root = Tk()

c = Canvas(root, width=200, height=200, bg='white')
c.pack()
```

1. Рисуем линии методом `create_line`

В методе `create_line` сначала указываются координаты начала (x_1, y_1), затем – конца (x_2, y_2).

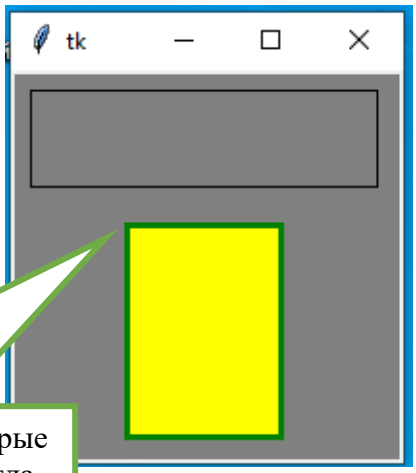
```
c.create_line(10, 10, 190, 50)

c.create_line(100, 180, 100, 60, fill='green',
                width=5, arrow=LAST, dash=(10,2),
                activefill='lightgreen',
                arrowshape="10 20 10")
root.mainloop()
```



Остальные свойства являются необязательными. Так `activefill` определяет цвет отрезка при наведении на него курсора мыши.

2. Рисуем прямоугольники методом `create_rectangle`

```
c.create_rectangle(10, 10, 190, 60)

c.create_rectangle(60, 80, 140, 190,
                   fill='yellow',
                   outline='green',
                   width=3,
                   activedash=(5, 4))

root.mainloop()
```



Первые координаты – верхний левый угол, вторые – правый нижний. В приведенном примере, когда на второй прямоугольник попадает курсор мыши, его рамка становится пунктирной, что определяется свойством `activedash`.

3. Рисуем многоугольники методом `create_polygon`

```
c.create_polygon(100, 10, 20, 90, 180, 90)

c.create_polygon(40, 110, 160, 110,
                 190, 180, 10, 180,
                 fill='orange', outline='black')


root.mainloop()
```

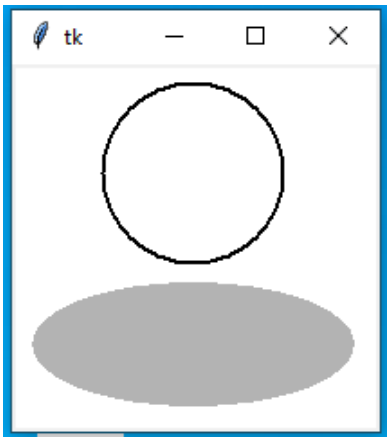
Для удобства координаты точек можно заключать в скобки:

```
c.create_polygon((40, 110), (160, 110),
                 (190, 180), (10, 180),
                 fill='orange', outline='black')
```

4. Создаем эллипсы методом `create_oval`.

При этом задаются координаты гипотетического прямоугольника, описывающего эллипс. Если нужно получить круг, то соответственно описываемый прямоугольник должен быть квадратом.

```
c.create_oval(50, 10, 150, 110, width=2)
c.create_oval(10, 120, 190, 190,
              fill='grey70', outline='white')
root.mainloop()
```

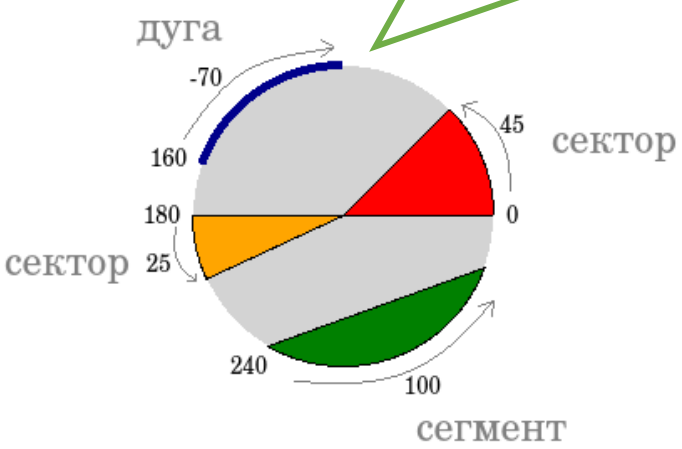


5. Более сложные объекты с использованием метода `create_arc`.

В зависимости от значения опции `style` можно получить сектор (по умолчанию), сегмент (`CHORD`) или дугу (`ARC`). Также как в случае `create_oval` координаты задают прямоугольник, в который вписана окружность (или эллипс), из которой "вырезают" сектор, сегмент или дугу. Опции `start` присваивается градус начала фигуры, `extent` определяет угол поворота.

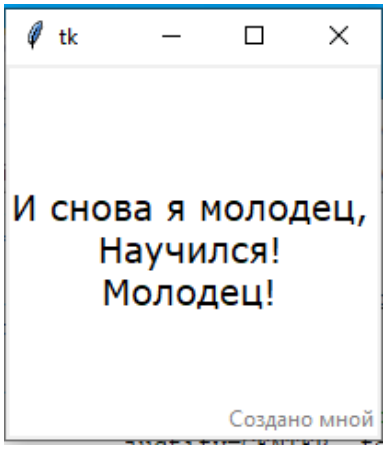
```
c.create_oval(10, 10, 190, 190,
              fill='lightgrey',
              outline='white')
c.create_arc(10, 10, 190, 190,
             start=0, extent=45,
             fill='red')
c.create_arc(10, 10, 190, 190,
             start=180, extent=25,
             fill='orange')
c.create_arc(10, 10, 190, 190,
             start=240, extent=100,
             style=CHORD, fill='green')
c.create_arc(10, 10, 190, 190,
             start=160, extent=-70,
             style=ARC, outline='darkblue',
             width=5)

root.mainloop()
```

В данном примере светло-серый круг используется исключительно для наглядности.



6. Размещаем текст с помощью метода `create_text`:

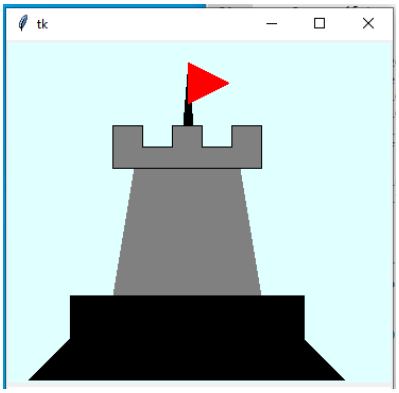
```
c.create_text(100, 100,
              text="И снова я молодец,\nНаучился!\nМолодец!",
              justify=CENTER, font="Verdana 14")
c.create_text(200, 200, text="Создано мной",
              anchor=SE, fill="grey")
root.mainloop()
```



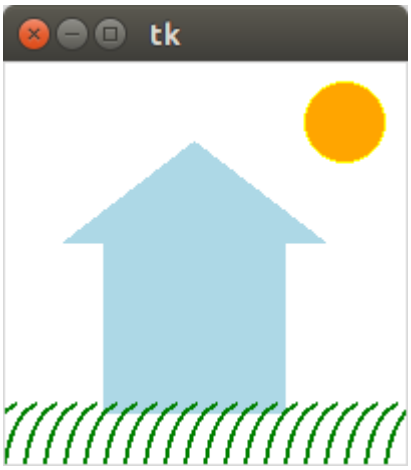
По умолчанию в заданной координате располагается центр текстовой надписи. Чтобы изменить это умолчание и, например, разместить по указанной координате левую границу текста, используется anchor (якорь) со значением `W` (от англ. west – запад). Другие значения: `N`, `NE`, `E`, `SE`, `S`, `SW`, `W`, `NW`. Если букв, задающих сторону привязки, две, то вторая определяет вертикальную привязку (вверх или вниз «уйдет» текст от заданной координаты). Свойство `justify` определяет лишь выравнивание текста относительно себя самого.

Практическая работа

1. Создайте башню как у меня или замок, который придумайте сами



2. Создайте на холсте подобное изображение:



Для создания травы используется цикл.