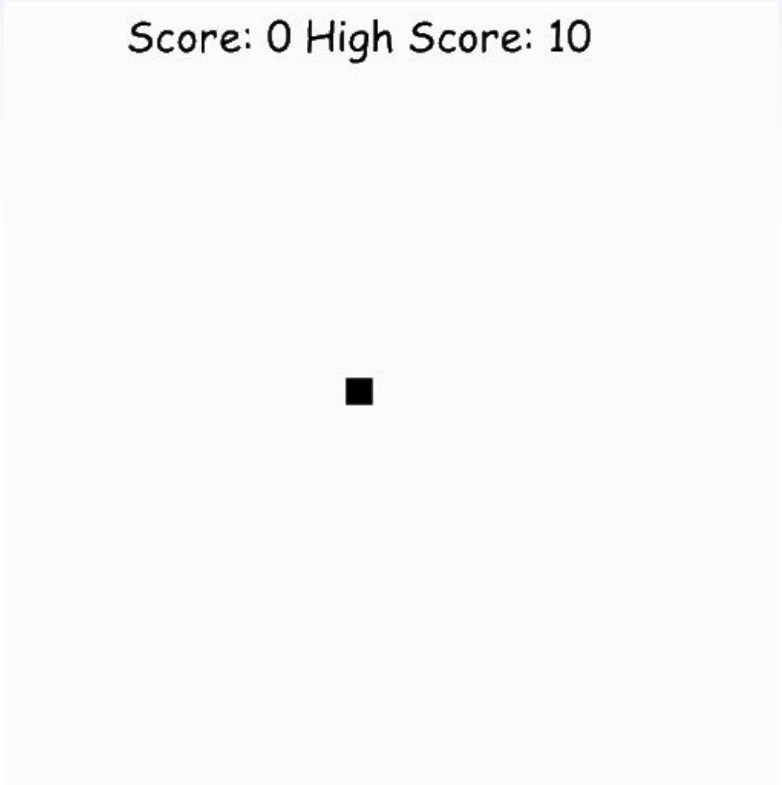
Welcome to
Python in Projects: Snake

Today's Lesson

- Snake!



Setup

```
import turtle
import time
import random


delay = 0.1
score = 0
high_score = 0
segments = [ ]
```

Creating Screen

```
wn = turtle.Screen()
wn.setup(width=600, height=600)
wn.tracer(0)
foodLst = [ ]
```

Creating Head

```
head = turtle.Turtle()
head.speed(0)
head.shape("square")
head.color("black")
head.penup()
head.goto(0,0)
head.direction = "stop"
```

Creating First Food

```
food = turtle.Turtle()
food.speed(0)
food.shape("circle")
food.color("orange")
food.penup()
food.goto(0, 100)
foodLst.append(food)
```

Text!

```
pen = turtle.Turtle()
pen.speed(0)
pen.shape("square")
pen.color("black")
pen.penup()
pen.hideturtle()
pen.goto(0, 260)
pen.write("Score: 0 High Score: 0", align="center",
font=("Comic Sans MS", 24,"normal"))
```

Moving Up and Down

```
def go_up():
~if head.direction != "down":
~~head.direction = "up"
def go_down():
~if head.direction != "up":
~~head.direction = "down"
```

Moving Left and Right

```
def go_left():
~if head.direction != "right":
~~head.direction = "left"
def go_right():
~if head.direction != "left":
~~head.direction = "right"
```

Moving

```
def move():
~if head.direction == "up":
~~y = head.ycor()
~~head.sety(y+20)
~if head.direction == "down":
~~y = head.ycor()
~~head.sety(y-20)
```

Moving More

```
~if head.direction == "left":
~~x = head.xcor()
~~head.setx(x - 20)
~if head.direction == "right":
~~x = head.xcor()
~~head.setx(x + 20)
```

Setting Keyboard Commands

```
wn.listen()
wn.onkey(go_up, "Up")
wn.onkey(go_down, "Down")
wn.onkey(go_right, "Right")
wn.onkey(go_left, "Left")
```

Main Loop

```
while True:
~wn.update()
```

If Borders

~if head.xcor() > 290 or head.xcor() < - 290 or head.ycor()> 290 or head.ycor() < -290:

~~time.sleep(0.5)

~~head.goto(0,0)

~~head.direction = "stop"

Resetting

~~score = 0

~~delay = 0.1

~~pen.clear()

~~pen.write("Score: {} High Score: {}".format(score, high_score), align="center", font=("Comic Sans MS", 24, "normal"))

More Resetting

~~for segment in segments:

~~~segment.goto(1000,1000)

~~del segments[:]
~~~

Generating More Food

~for food in foodLst:

~~if head.distance(food) < 20:

~~~colorlst = ["red", "orange", "yellow", "green", "lime green", "blue", "purple", "pink"]

~~~colorIndex = random.randint(0, 7)

~~~color = colorlst[colorIndex]

~~~x = random.randint(-290, 290)

~~~y = random.randint(-290, 290)

~~~food.goto(x, y)

New Segment When Eating

```
~~~new_segment = turtle.Turtle()
~~~new_segment.speed(0)
~~~new_segment.shape("square")
~~~new_segment.color(color)
~~~new_segment.penup()
~~~segments.append(new_segment)
```

Updating When Eating

~~~score += 10

~~~delay -= 0.001

~~~if score > high_score:

~~~~high_score = score

~~~pen.clear()

~~~pen.write("Score: {} High Score: {}".format(score, high_score),align="center", font=("Comic Sans MS", 24,"normal"))

Segments Change

```
~for index in range(len(segments)-1, 0, -1):
~~x = segments[index-1].xcor()
~~y = segments[index-1].ycor()
~~segments[index].goto(x,y)
```

More Segments Updating

```
~if len(segments) > 0:
~~x = head.xcor()
~~y = head.ycor()
~~segments[0].goto(x,y)


~move()
```

Reset If Hits Self

```
~for segment in segments:
~~if segment.distance(head) < 20:
~~~time.sleep(1)
~~~head.goto(0,0)
~~~head.direction = "stop"
```

Continue To Reset

```
~~~score = 0
~~~pen.clear()
~~~pen.write("Score: {} High Score: {}".format(score, high_score),
align="center",
~~~font=("Comic Sans MS", 24, "normal"))
~~~for segment in segments:
~~~~segment.goto(1000, 1000)
```

Continue To Reset

```
~~~del segments[ : ]
~~~score = 0
~~~delay -= 0.001
~~~pen.clear()
~~~pen.write("Score: {} High Score: {}".format(score, high_score),
align="center", font=("Comic Sans", 24, "normal"))
```

Ending

~time.sleep(delay)

wn.mainloop()