

## Lecture 20

**Topics:** *Chapter 10. Defining Classes*

**10.1** Quick review of objects

**10.2** Example program: target

**10.3** Defining new classes

## 10.1 Quick review of objects

In previous chapters we have developed techniques for *structuring the computation of the program*.

In the next few chapters we will review some techniques *structuring the data* that our programs use.

## 10.1 Quick review of objects

**Object** is an active data type that *knows stuff* and *can do stuff*.

An object consists of :

- A collection of related information (*knows stuff*)  
*instance variables*
- A set of operations to manipulate the information (*can do stuff*)  
*methods*

Collectively, *instance variables* and *methods* are called *attributes of an object*.

## 10.1 Quick review of objects

Objects *interact* by sending each other messages (*requests for an object to perform one of its operations*).

Every object is an *instance* of some class.

New objects are created by invoking a *constructor*.

**Example:**

```
c = Circle(Point(100,100),20)
```

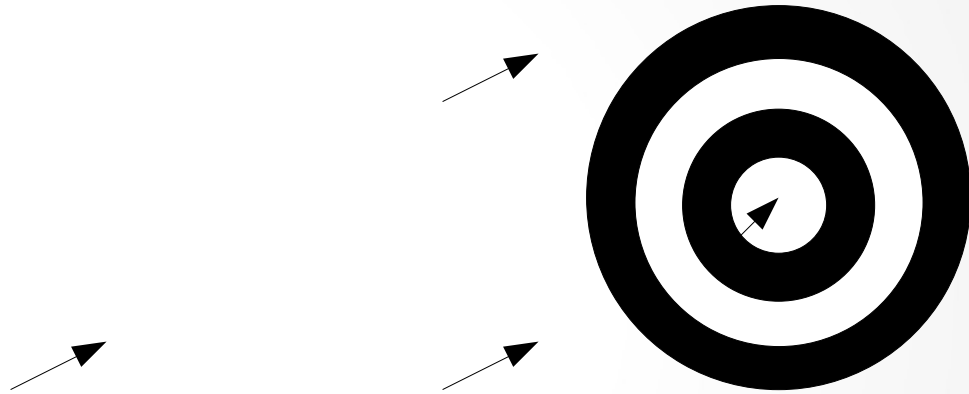
The parameters to the constructor are used to initialize some of the instance variables (*center, radius* for Circle).

Then we can manipulate the object (*draw, move, undraw*).

## 10.2 Example program: target

### Example:

Let's create a dart board and emulate dart throwing



## 10.2 Example program: target

### Example:

Let's create a dart board and emulate dart throwing

```
r = 200
for i in range(10):
    C = Circle(Point(400, 300), r-20*i)
    if i%2 == 1:
        C.setFill("white")
    else:
        C.setFill("black")
    C.draw(w)
```



## 10.2 Example program: target



### Example:

Let's create a dart board and emulate dart throwing

```
p = w.getMouse() # getting the mouse click
x = p.getX()    # store x-coordinate of the click
y = p.getY()    # store y-coordinate of the click

L = Line(Point(x-20,y+20),p) # create a dart
L.setArrow("last") # with the arrow at the end
L.draw(w) # draw it
```

## 10.2 Example program: target



### **Example:**

Let's create a dart board and emulate dart throwing

See programs [target.py](#) and [target\\_mod.py](#)



## 10.3 Defining new classes

Class definition has the following form:

```
class <class-name>:  
    <method-definitions>
```

## 10.3 Defining new classes

Class definition has the following form:

```
class <class-name>:  
    <method-definitions>
```

**Exercise:** Let's define a class for a die

- A die has  $n$  sides  
(set it up during the *initialization*),
  - Instance variable `sides`
- When we roll a die it gets some value
  - Method `roll`
- A die has a side up after we roll it
  - Instance variable `value`



## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
```

```
    def __init__(self, sides):  
        self.sides = sides  
        self.value = 1
```

```
    def roll(self):  
        self.value = randrange(1, self.sides+1)
```

## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
```

```
    def __init__(self, sides):  
        self.sides = sides  
        self.value = 1
```

```
    def roll(self):  
        self.value = randrange(1, self.sides+1)
```

## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
    def __init__(self, sides):
        self.sides = sides
        self.value = 1
    def roll(self):
        self.value = randrange(1, self.sides+1)
```

*class name*

*constructor*

## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
    def __init__(self, sides):
        self.sides = sides
        self.value = 1
    def roll(self):
        self.value = randrange(1, self.sides+1)
```

*class name*

*constructor*

*instance variables*

## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
    def __init__(self, sides):
        self.sides = sides
        self.value = 1
    def roll(self):
        self.value = randrange(1, self.sides+1)
```

*class name*

*constructor*

*instance variables*

*method of the class MSDie*

## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
```

```
    def __init__(self, sides):  
        self.sides = sides  
        self.value = 1
```

```
    def roll(self):  
        self.value = randrange(1, self.sides+1)
```

*first parameter of  
each method – a  
reference to the  
object it works on*



## 10.3 Defining new classes

**Exercise:** Let's define a class for a die



what other methods should we add?

- recall Point, Text, ... from graphics library

## 10.3 Defining new classes

**Exercise:** Let's define a class for a die

what other methods should we add?

`getValue` and `setValue` methods



## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
```

```
    def __init__(self, sides):  
        self.sides = sides  
        self.value = 1
```

```
    def roll(self):  
        self.value = randrange(1, self.sides+1)
```

```
    def getValue(self):  
        return self.value
```

```
    def setValue(self, value):  
        self.value = value
```

## 10.3 Defining new classes

**Exercise:** Let's define a class for a die

```
from random import randrange
```

```
class MSDie:
```

```
    def __init__(self, sides):  
        self.sides = sides  
        self.value = 1
```

```
    def roll(self):  
        self.value = randrange(1, self.sides+1)
```

```
    def getValue(self):  
        return self.value
```

```
    def setValue(self, value):  
        self.value = value
```

MSDie
self.sides self.value
__init__(self, sides) roll(self) getValue(self) setValue(self, value)

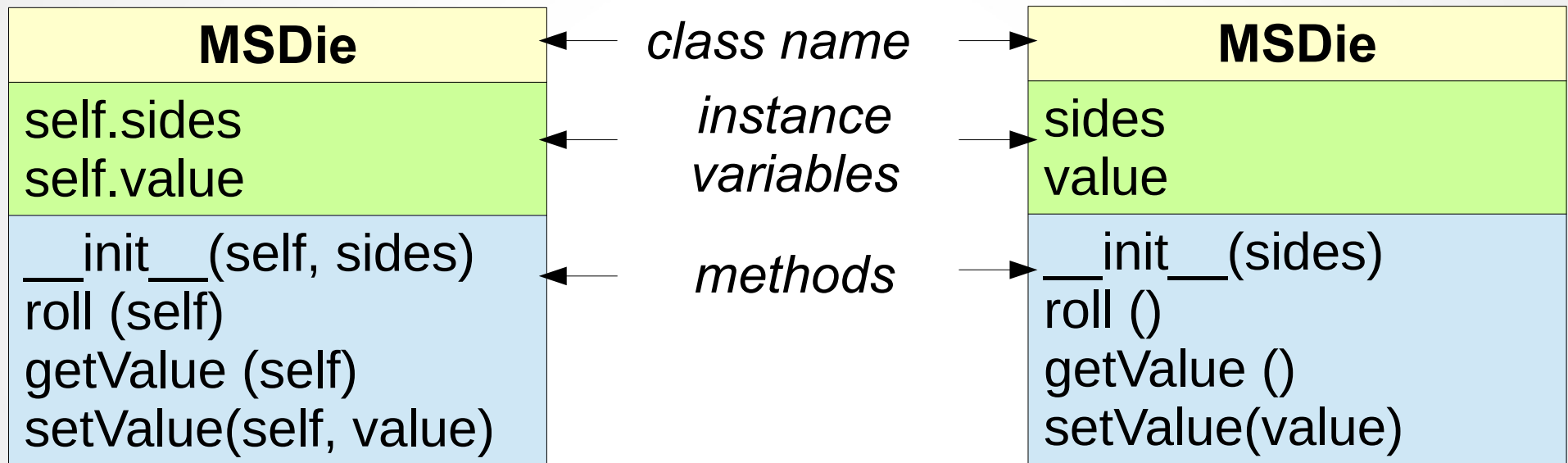
*class diagram*



## 10.3 Defining new classes



**Exercise:** Let's define a class for a die



*class diagram*

*class diagram*

## 10.3 Defining new classes



**Exercise:** Let's define a class for a die

See program [dice.py](#)