Making a Game
CS106AP Lecture 22
Roadmap

Day 1!

Programming Basics

The Console

Images

Data structures

Midterm

Graphics

Graphics 1.0
Graphics 2.0
Event-driven programming

Object-Oriented Programming

Everyday Python

Life after CS106AP!
Today’s questions

How can we write a game?
Today’s topics

1. Review
2. Creating a graphical, event-driven game
3. What’s next?
Review
Event-Driven Programs
The event listener model

Your code

def main():
   ...
   ...

def your_mouse_listener():
   ...

**Definition**

**mouse listener function**

A function that occurs immediately when a user triggers a particular **mouse event**

**clicking, moving, dragging**
The event listener model

Your code

```python
def main():
    ...

    ...

    def your_mouse_listener():
        ...
```

The function happens immediately, no matter where you are in your program!
Creating a mouse listener

1. Write a mouse listener function (handler)

   ```python
def mouse_listener_handler(event):
    ...
```

2. Use the corresponding campy `onmouseevent()` function to set up your mouse listener

   ```python
   onmouseclicked(mouse_listener_handler)
   ```
Abstraction
Why do we use classes?

- For ourselves
  - Grouping related data and the functions that act on it
  - Modular code development (isolation of particular tasks)

- For others
  - We hide the implementation details of our code so others don’t need to worry about them.
  - They can just use the class, like we do for SimpleImage.
Clients and Interfaces

- Classes—or really any code we write (modules, libraries, etc.)—can be thought of from two perspectives.
- The point at which the client and implementation meet and communicate is known as the interface, which serves as both a barrier and a communication channel.
Information Hiding

- One of the central principles of modern software design: **Hide as much complexity as possible** from the layers that depend on it.
  - This principle is called **information hiding**.
- When you **use** a function, it is more important to know **what the function does** than to understand exactly how it works.
Thinking about Objects

I need a bunch of GRects...

class GRect:
    def __init__(self,...):
    def move(self,...):
    def rotate(self,...):

rect = GRect(width,height)
rect.move(dx, dy)
rect.filled = True

campy
GRect
GOval
GLine
GLabel
...

Abstraction boundary (interface)
Abstraction protects the data stored in an object

- Getters and setters are the interface to the data
- Clients don’t have to worry about constraints on the data
How can we write a game?
A Story...
Bouncing Balls
y velocity stays the same
Bouncing Balls

x velocity
reverses
Bouncing Balls
Bouncing Balls

$x$ velocity stays the same
Bouncing Balls

y velocity reverses
Is Ball in Zone?

- No
- Yes
What’s next?
Tuples

- How can we bundle small amounts of information together

- Can a function output multiple pieces of information?

- More generally: Python idioms and life after CS106AP
Life after CS106AP!

Day 1!

Roadmap

Programming Basics

The Console

Images

Data structures

Midterm

Graphics

Object-Oriented Programming

Everyday Python

Tuples

List Comp.

Jupyter

Internet

Life after Exam

Life after CS106AP!