Writing Interactive Programs

Alice
Overview

- Control of flow
- Creating our own methods for interactive worlds
- Planning ... how is the storyboard different?
- Behavior ... a link between event and responding method
Control of flow

*Control of flow* -- how the sequence of actions in a program is controlled.
- What action happens first, what happens next, and then what happens...and so on.

In movie-style programs ( chapters 1-4) the sequence of actions is determined by the programmer
- Creating a storyboard design
- Writing program methods to carry out the designed sequence
Interactive Animations

In interactive programs, the sequence of actions is determined at runtime when the user provides input:
- clicks the mouse
- presses a key on the keyboard

Other sources of input are possible, depending on your computer system.
Interactive games

In a video game where the user is guiding a spaceship, the sequence of actions in the game depends on what direction the user guides the ship and how fast the user presses the controls.

Each time the program runs, user input may cause a different sequence of actions from previous executions of the program.

In essence, control of flow is now “in the hands of the user!”
Events

Each time the user provides some sort of input, we say an event is generated.

An event is “something that happens”
Event-handlers

An event may
- Trigger a response, or
- Move objects into positions that create some condition (e.g., a collision) that triggers a response.

A method is invoked to carry out the response. We call this kind of method an **event-handler**.

When an event is linked to an event-handler, a **behavior** is created.
How does this affect your program?

The goal is to write your own interactive programs.
The approach will be the same as before, but the difference is that we must now be concerned with **behaviors**:
- input from the user (**events**)  
- how objects respond to an event (**event-handlers**)
Example

Build an air show flight simulator. In an air show, the pilot uses biplane controls to perform acrobatic stunts.
Problem

The whole purpose of a flight simulator is to allow the user to control the flight path.

The problem is: how do we write our program code to provide a guidance system allowing the user to be the pilot?
Solution

Use keyboard input

- Up-arrow key to move the biplane forward
- Space to make the biplane do a barrel turn

(Note: other sets of keys could be used, we just arbitrarily picked a couple of keys on the keyboard.)

Write event-handler methods that respond to each key press
Since two keys are used, two events are possible – so two storyboards are needed:

Event: Space press
Response:
  Simultaneously:
  roll biplane a full revolution
  play biplane engine sound

Event: Up Arrow key press
Response:
  Simultaneously:
  move biplane forward
  play biplane engine sound

Each storyboard outlines an event-handler that responds to a particular event.
Demo

A demonstration of building the biplane acrobatic simulation
Biplane.flyForward method

No parameters

No variables

```
// simple horizontal move forward
```

Do together

- Biplane move forward 1.25 meters duration = 1 second style = abruptly
- Biplane play sound Biplane.biPlaneShort (0:02.324) duration = 1 second
Biplane.barrel method

```
// biplane makes a full rotation right

Do together
- Biplane roll right 1 revolution duration = 1 second style = abruptly
- Biplane play sound Biplane.biPlaneShort (0:02.324) duration = 1 second
```
Events Editor

Once the event-handler methods are written, each method must be linked to an event.
Assignment

Read Chapter 5 Section 1

- Control of flow
- Events
- Event-handlers
- Behavior