Day 1 – Flash Listeners

Question of the Day:

- What is a symbol in Flash?
- How do you create a symbol in Flash?
- How does Flash know the difference between copies of the same symbol?

Objectives:

- Answer Question of the Day.
- Review **Flash Symbols.**
- Complete review assignment.

Assignment:

- Create a Flash program in which at least 2 movie clips are controlled via ActionScript.
- Show Mr. Fornstrom when finished (**5 pts**).

Answer to Question of the Day:

- A **Symbol** is a reusable image, animation, or button in Flash.
- Create a symbol: 1. create an object 2. turn into symbol: **Modify / Convert to Symbol** 3. Specify a name 4. Provide an **Instance Name** for all instance copies on the stage.
- An **Instance** is a copy of any symbol from your Flash file's library which resides on the stage. **Instance Names** are used by ActionScript to identify each instance. This is what distinguishes copies of the same symbol.

Day 2 & Day 3 – Flash Listeners

Question of the Day:

- What are **Listeners** used for in Flash ActionScript?
- List advantages of usings Listeners in ActionScript.
- List the steps for using Listeners in ActionScript.

Objectives:

- Answer Question of the Day.
- Demo **Listeners** in ActionScript.
- **Demo / Assignment** (10 points): due at the end of class on Day 3 → show Mr. Fornstrom → Use Listeners to animate Pac-Man
    - Copy the following folder to your H:\ drive **\\tjhsnt01\Shared\MrFornstrom\ClassResources\FlashResources\ActionScript Listeners**
    - Open the file "**pac-man-sprites.fla**"
    - Save the file as "**pac-man-yourName**"
    - Press <ctrl> <enter> to run the program. There should be 4 animations playing, and 1 static Pac-Man in the middle of the screen.
    - Provide static Pac-Man with instance name "**pac**"
    - Double-click on pac. You are now in the timeline for the **pac** instance. Name the 1st layer "**actions**", add a 2nd layer and name "**graphics**"
    - Select **actions layer**, put in code: **stop( )**
    - Select **graphics layer** frame 1, right-click and choose "**Convert to Keyframe**". Delete static pac in frame 2, drag moveright into frame 2.
    - In frame 2, right-click and choose "**Convert to Keyframe**". Delete moveright pac in frame 3, drag moveleft pac into frame 3.
    - In frame 3, right-click and choose "**Convert to Keyframe**". Delete moveleft pac in frame 4, drag moveup pac into frame 4.
    - In frame 4, right-click and choose "**Convert to Keyframe**". Delete moveup pac in frame 5, drag movedown pac into frame 5.
    - Open "**PacAnimation.txt**", select and copy all of the text.
    - Return to the main timeline of your Flash file, select the **actions frame** and paste code.
    - Press <ctrl> <enter> to run program; use the arrow keys on the keyboard to move Pac-Man.

Answer to Question of the Day:

- What are **Listeners** used for in Flash ActionScript?
  → Listeners receive and respond to events. Listeners allow developers to enable objects to become aware of (or “listen for”) events that occur in the environment.
- List advantages of usings Listeners in ActionScript.
  1. Makes **code management easier** by keeping all event-handling code in one or two loctions in the movie, as opposed to attaching code to many objects in the movie.
  2. Provides **more flexibility** as to which events Flash responds to. Events are determined by the developer rather than Flash.

List the steps for using Listeners in ActionScript.

1**. Set up the Listener as an object in the declarations section of the code.** (At the top of code, not attached to an event).

```
myListener = new Object();
```

2. Use the **addListener( )** method to **register the listener** with the event source. **removeListener( )** can be used to “unregister” an object with and event source. The Listener must be attached to 1 of 5 possible event sources: (**Key, Mouse, Selection, Stage, Text Field**)

```
Key.addListener(myListener);
// Note syntax:  EventSource.addListener(ListenerName);
```

3. Define the Listener function(s).

```
myListener.onKeyDown = function( ) {
        trace(“A key is being held down”);
}   // end myListener.onKeyDown

myListener.onKeyUp = function( ) {
        trace(“A key has just been released”);
}   // end myListener. onKeyUp
```

PacAnimation.txt

```
stop( )
// pac-man movement variables:
xmove = 0;
ymove = 0;

// Step #1 - Create a new listener object
myListener = new Object();

// Step #2 - Register the listener
Key.addListener(myListener);

// Step #3 Define the Listener functions
myListener.onKeyDown = function( ) {
        // determine which key was pressed
        if (Key.getCode() == 38) { // up key pressed
                _root.pac.gotoAndStop(4);
                ymove = -5;
        } else if (Key.getCode() == 39) { // right key pressed
                _root.pac.gotoAndStop(2);
                xmove = 5;
        } else if (Key.getCode() == 37) { // left key pressed
                _root.pac.gotoAndStop(3);
                xmove = -5;
        } else if (Key.getCode() == 40) { // down key pressed
                _root.pac.gotoAndStop(5);
                ymove = 5;
        } // end Key.getCode() pressed

}   // end myListener.onKeyDown

myListener.onKeyUp = function( ) {
        xmove = 0;
        ymove = 0;
        _root.pac.gotoAndStop(1);
}   // end myListener. onKeyUp

// make pac-man move when he enters the frame.
pac.onEnterFrame = function( ) {
        // move pac-man
        this._x += xmove;
        this._y += ymove;
} // end pac.onEnterFrame
```

Day 4 & Day 5 Flash Listeners

- **Assignment**: due at the end of class on Day 5 → show Mr. Fornstrom (10 points)
  - Create **animated gif** files of a figure that moves in at least 3 directions and at least 1 file in which the figure is stopped. You will need at least 4 files. The animation's can be any character and move in any way that you want.
  - Create a new Flash file named: **AnimationDemo**.
  - Import the **animated gif** files into **AnimationDemo**. Use descriptive names when importing animations to the **Library.**
  - Use **Listeners** to program movement for the animations.