

Art 382: Introduction to Interactive Media

mcdo@umbc.edu

Lecture 14, April 8

Schedule

- Take Roll
- Hand out grades
 - notebook postmortem
- Review: buttons
- Next step: chunks

Grades

- Assignment A grades pretty high
 - don't mind
- Notebook grades are good
- Assignment B grades OK

Assignment C

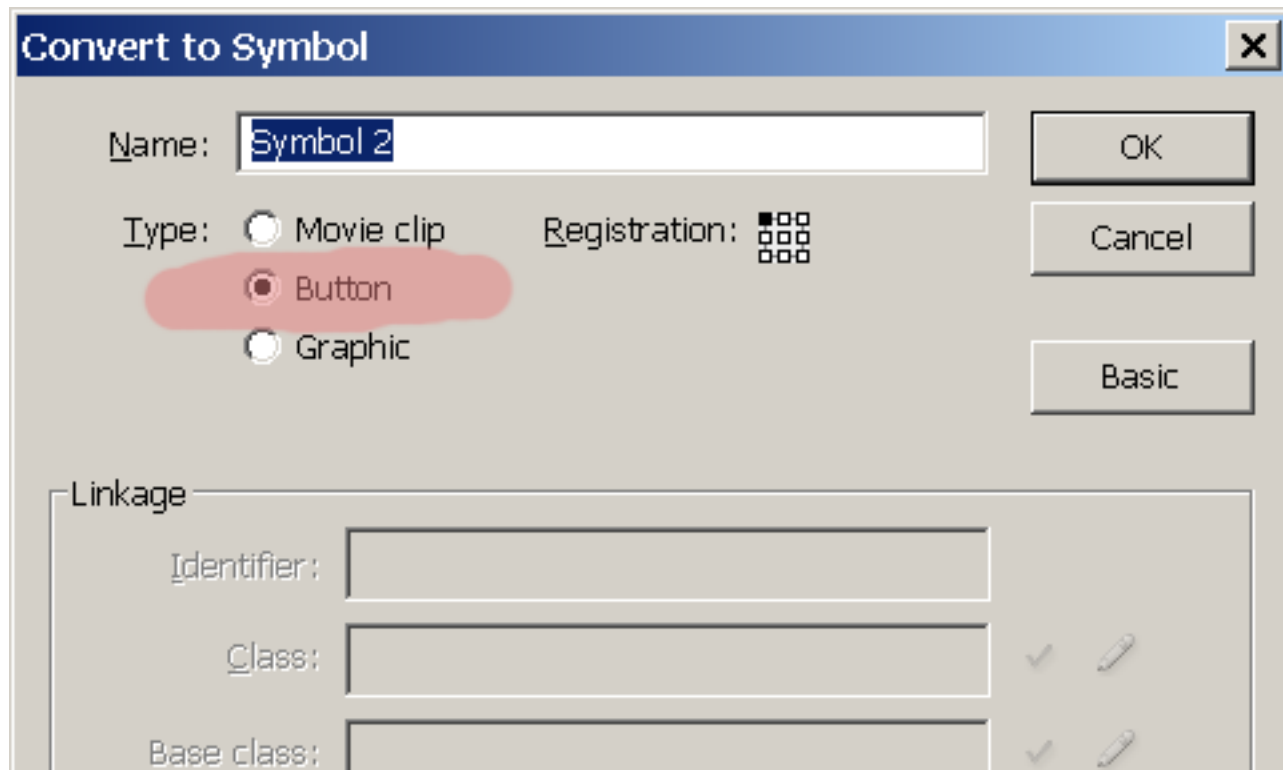
- Make an interactive killing device simulator
- Think of old monster movies
- Use buttons, animations
- Use chunks

Button-making review

- make a button symbol
- name the button
- add script to frame 1
 - addEventListener
 - a callback routine

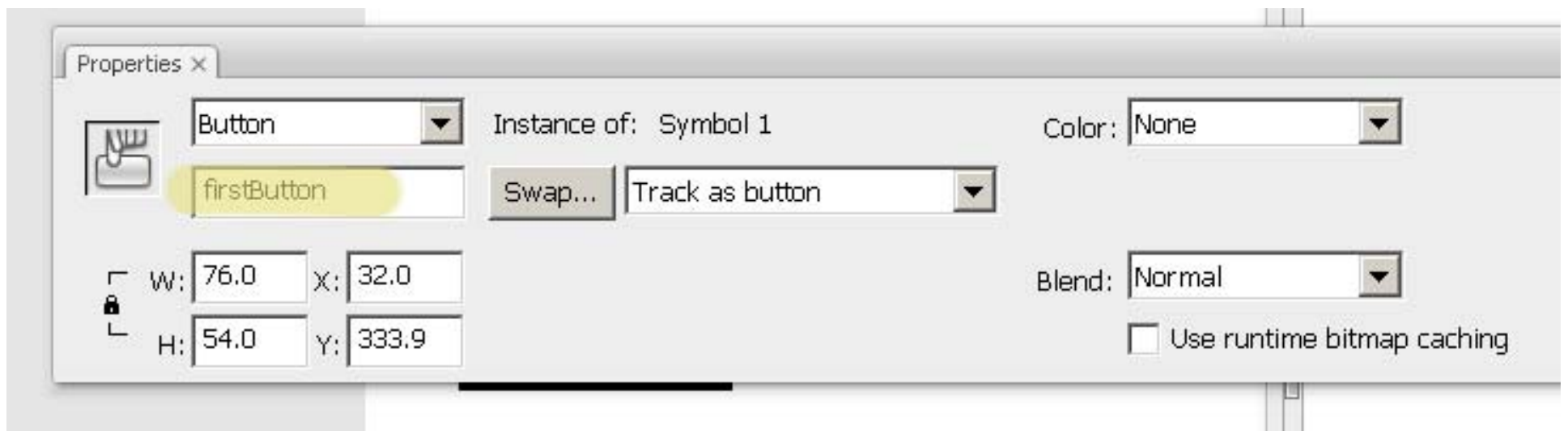
1: make a button

- same as making a symbol
 - but you make a "button" symbol



2: name the button

- look at the button's properties
 - ctrl-f3 for PC's, or use the menu
- set the button's "instance name"
 - you're naming that copy of the button symbol
 - call it "firstButton"



3: attach a function

- add this code to your script layer:

```
firstButton.addEventListener(  
    MouseEvent.CLICK, clickHandler);
```

```
function clickHandler(evt:Object):void {  
    gotoAndPlay(2);  
}
```

Great. Buttons. Now What?

- You know how to make a button, but not what to do with it.

Chunks

- Sections of the timeline that implement a state
- Chunks include
 - a set of buttons
 - an animation loop
 - an intro animation
 - an exit animation

Why does Neal like chunks?

- 1) Breaks the "interactivity" up into smaller parts
- 2) Reflects how most computer-augmented gizmos operate: modes
- 3) Easy as pie to set up
- 4) Adequate for most interface design jobs
- 5) Embodies good interface design practices
 - visual cues for mode transitions
 - separation of modes

What is a chunk-- in Flash?

- A set of frames
- With a "gotoAndPlay" statement at the end
- The only way out of the chunk is through a button's code
- The only way into a chunk is from the start, or by clicking buttons
 - so, once you're in, you loop forever

Chunk context

- Not a part of Flash!
- "Chunk": my own private jargon
- Absolutely not necessary to make interactivity in Flash

- My invention; imposes helpful structure
- Like training wheels

Let's make a chunk

- first, make four animations
- 3 animations of a red button
 - first: moving from offstage-left to center
 - entry animation
 - second: moving in a little twirl
 - loop animation
 - third: moving offscreen right
 - exit animation
- 1 with a green box

add the usual button code

```
firstButton.addEventListener(  
    MouseEvent.CLICK, clickHandler);
```

```
function clickHandler(evt:Object):void {  
    gotoAndPlay(XXXX);  
}
```

Set XXXX to be the first frame of your exit animation

tie off the chunk

Put a "gotoAndPlay" command at the end of the "twirl" animation

Have it jump to the beginning of the "twirl"

Once the player enters the chunk, it stays in it forever

Once you enter a mode, you stay, usually

enter the chunk

- make sure that flash ends up in the chunk
 - either make the intro animation Frame 1 or 2
 - or use "gotoAndPlay"

return to the chunk

- put a "gotoAndPlay" at the end of the green animation
- test!
 - red button enters
 - twirls until you click
 - exits, plays green, re-enters

One Chunk per Mode!

- Make another chunk!
 - 3 more animations: a blue button
 - name and add code for the blue button
 - tie off the blue chunk
 - have the red button jump to blue, blue to green

chunk components

- frames 1-10 : entry animation
- frames 10-20 : loop animation
- frames 20-30 : exit animation
- frame 20: gotoAndPlay(10)
 - makes the loop repeat
- frames 45-70 : a second chunk

