

## TCH090: Flash Game Development (Elective)

*formerly Online Game Design*

### Course Overview

This course introduces students to the design of online Flash games. They learn how to develop games in a variety of genres. They learn some basic programming concepts using ActionScript, the native scripting language of Flash, to develop cool games and publish them online. By the end of this course, students will have a collection of fully functioning multi-level online games.

**Course Length:** One semester

**Materials:**

*Software:* Adobe Flash CS4 (preferred) or Adobe Flash CS5

*System Requirements:* Microsoft Windows XP, Windows Vista, or Mac OS X operating system; 1 GHz or faster processor (for Mac OS X, a PowerPC G5 or multicore Intel processor); 1 GB of memory (RAM); at least 2 GB of available hard drive space

**Prerequisites:** None

### Course Outline

#### Course Overview

Students learn how to navigate the course and turn in assignments. They set up a course folder and download and install software and course resources.

- ◆ Start the Course
- ◆ Set Up Your Computer
- ◆ Set Up a Browser and Install 7-Zip
- ◆ Download Resources and Zip Assignments

#### Project 1: Zoo Escape

Students identify the purpose of branching in a game, set up the Flash workspace, and add and name keyframes. They add, name, and arrange layers; add and align images; and add and edit text. They add "play again" buttons, test the game, write an ActionScript stop() function, and convert movie clip symbols to button symbols. They write an event handler, check the syntax of code, create title screen frames, and publish the game as SWF and HTML files. Students take a quiz and complete a graded assignment.

- ◆ Set Up the Rooms
- ◆ Create a Room
- ◆ Add Text
- ◆ Create More Rooms
- ◆ Control the Timeline
- ◆ Create and Edit Buttons
- ◆ Code the First Doors
- ◆ Code the Other Buttons

- ♦ Make a Title Screen

### **Project 2: Pirate Ship Peril**

Students add and align a background image; add a title, text, and start button; and add frame labels and a stop() function. They check the syntax and test the game. They open the "ship's child" timeline and add code inside it, and code the ship to move when the mouse clicks and drags it. They write and change hit tests, add dynamic timer text, and write a countdown function. They add the end screen background and text, add and code "play again" buttons, and publish the game as SWF and HTML files. Students take a quiz and complete a graded assignment.

- ♦ Make the Title Screen
- ♦ Create Level 1
- ♦ Code the Ship
- ♦ Add a Goal Hit Test
- ♦ Add Maze Hit Tests
- ♦ Add a Timer
- ♦ Make Level 2
- ♦ Add End Screens

### **Project 3: Dragon Tamer**

Students add a background and maze, add a player and lives, and add collection items. They code the player to move with the arrow keys, make item and life variables, and code the goal so the player wins if he reaches the goal with all the items. Students add an enemy and add code that makes the enemy move with artificial intelligence. They code the player to lose a life and to scream when it hits an enemy. They add more enemies and code the game so the player loses when all the lives are lost. They add background music and a title screen, then publish the game as SWF and HTML files. Students take a quiz and complete a graded assignment.

- ♦ Add Game Items
- ♦ Code the Player
- ♦ Code the Food Items
- ♦ Add an Enemy
- ♦ Hurt the Player
- ♦ Add Music and a Title Screen

### **Project 4: Space Scavenger**

Students add a starry sky and code the game so it scrolls vertically. They create a parallax perspective by scrolling the near stars faster than the far stars. They add a spaceship and a laser and use an alpha effect to make the laser disappear quickly or fade. They add a scoreboard with static and dynamic text; create variables for the time, score, lives, and level; and code the scoreboard and timer. They code the player to lose a life when an enemy hits it, export explosions, and play an explosion when an enemy hits the player. They add a health pack and code it to restore a life to the player. They publish the game as SWF and HTML files. Students take a quiz and complete a graded assignment.

- ◆ Scroll the Stars
- ◆ Add the Spaceship
- ◆ Add Enemies
- ◆ Make a Scoreboard
- ◆ Make Title and End Screens
- ◆ Hurt the Player
- ◆ Destroy Enemies
- ◆ Restore Health

### **Project 5: Robot Rescue**

Students add a room background and add, rotate, and position doors on each wall. They add a player robot to the game, code the doors to move to the correct rooms when the player hits them, and add and code collection objects. They add generators and power buttons and code the game so the player can win and lose. They add a title frame, play button, and background music, and publish the game as SWF and HTML files. Students take a quiz and complete a graded assignment.

- ◆ Make the Start Room
- ◆ Add the Player
- ◆ Code the Doors
- ◆ Collect Objects
- ◆ Wake Up the Helper
- ◆ Add a Title and Sound