Instructor	Sam Stokes	Email	sstokes@microsoft.com
Classroom	SAC 2012	Class time	7 PM to 10 PM
Office	Call or email to set up apt.	Office Hours	
Phone	949 6275736 Skype: socalsamstokes	URL	http://blogs.msdn.com/devschool

Course description:

This course teaches students through lectures, discussions, demonstrations, and classroom labs. Students learn the knowledge, skills, and abilities necessary to create games in the C# programming language using the Microsoft XNA Framework and Silverlight.

This course is intended for people who aspire to careers as computer programmers and game developers.

Prerequisites:

- None

Textbook:

- Introduction to Programming through Game Development Using Microsoft XNA Game Studio 4.0
- Students can purchase the book, or use the free eBook provided by professor. Students will accessibility issues are ask to use the free eBook

Course Outcomes:

By the end of this course, the students should be able to:

1. Understand the use and application of “cloud” computing and the design of web based games
2. Understand the fundamentals of c# program construction including the implementation of blocks, conditional statements, loops, and switch constructions.
3. Understand the principles of variable creation and data storage in a c# program, including the use of identifiers, variable declaration, local and class scope variables, integer and floating point storage, widening, narrowing, and casting.
4. Make use of c# collections in programs, including arrays, lists, and dictionaries.
5. Understand the use of methods and the use of value, reference, and out parameters in method calls.
6. Create objects containing data fields, behaviors, and constructors.
7. Create data properties to provide managed access to data fields within an object.
8. Understand the fundamental difference between objects managed by value and reference.
9. Create class hierarchies which make use of method overriding to customize child class behaviors and constructor chaining to ensure proper creation of classes at each level in the hierarchy.
10. Use abstract classes to create template objects to be used as the base of a class hierarchy.
11. Create custom enumerated types with a range of values that reflect the problem domain.
12. Use interfaces to express a set of required behaviors and so allow the creation of software components.
13. Use delegates to implement method references and so bind methods to event generators.
14. Understand the c# pre-processor and the use of directives to manage the compilation process.
15. Understand the principles of test-driven development and the steps to be followed when creating software using this technique.
16. Understand the use of state machines when creating a program and how program states are expressed on a state diagram.
17. Understand the software engineering terms coupling, cohesion, and encapsulation. Know the reasons why high cohesion is a good design aim, along with low coupling and objects with high encapsulation.
18. Understand the reasons behind code refactoring and the need to reduce the duplication of statements, and to ensure that the identifiers of items in a program properly reflect the purpose of the item.

20. Understand the role of the XNA framework as a set of resources for game development.
21. Be able to construct the setup, update and draw behaviors to create a working game using XNA.
22. Make use of xna types in the creation of 2D sprite-based games, including Color, SpriteBatch, Texture2D, SpriteFont, Rectangle, Point, Vector, SoundEffect, Song, NetworkSession, GamepadState, and KeyboardState.
23. Use the XNA Content Manager to add textures and sound content to an XNA game, and the XNA content loading methods to retrieve the content when the game runs.
24. Use the gamepad and keyboard as input devices, both in a level- and edge- triggered manner.
25. Use the XNA Network application programmer interface to create networked games with a game lobby.
26. Create pseudo-3D displays by use of overdrawing and sprite scaling.
27. Create animated sprite game elements.
28. Use the C# random number generator in game play to introduce randomness to game play elements and behaviors.
29. Create XNA games for deployment to Xbox, Windows PC, and Windows Phone devices.
30. Use Microsoft Visual Studio to create and deploy XNA games.
31. Manage the deployment of games to the cloud, Xbox, Windows PC, and Zune devices.
32. Debug programs by the use of breakpoints, single stepping, and viewing the contents of variables.

Attendance:

Students are expected and encouraged to attend lectures and contribute to discussions. It is the student's responsibility to contact the instructor as early as possible if he/she cannot attend class. There will be no make-up opportunities, although all classes will have companion videos available on line.

Requirements:

To pass this course, you must meet the following requirements:

Grading:

% of grade	Task
20	Incorporate Twitter, Facebook, Bing Services, into your game
10	Create "web service" using SOAP, REST, JSON or other tools
30	Create 5 Windows Phone Applications, post them to AppHub and set up Ad Revenues using pubCenter. To be graded, each app must make an average of $5.00 and consume the web service provided by your Web Site
10	Web Site and Windows Phone Application must utilize Bing Maps. Student is required to build a web site application using the Bing Map SDK and submit the app for approval.
30	Sign up and submit for the Imagine Cup o As a team of 3 or 4, a Project Document, Video and Alpha project for the Software Design Category o As a team of 2 or 3, a Silverlight-based web game to the Imagine Cup.com site, with all materials required
	To get an automatic "A", succeed at any of the following for an automatic A, proof required, and meet all of the goals shown above o Pass through Round 2 of the Imagine Cup, o Make the most money on AppHub selling your phone app (must be greater than $10), o Make the most money using pubCenter (must exceed $10)

Grading Scale:

A's are awarded only if you meet one of the requirements specified AND do the rest of the work required.

100-95 = A- (Only if you are on a team that submits the round 2 project document, video, and single level game)

87-89 = B+

83-86 = B

80-82= B-

77-79= C+

73-76= C

70-72= C-

67-69= D+

63-66= D

below 60 = F

GENERAL POLICIES:

ACADEMIC POLICY

Honor Code. The following cases will be considered as violations: identical code, and extremely similar code. All code will be profiled using standard tools. Violations will be reported to the Office of Vice President of Academic Affairs.

ATTENDANCE POLICY

Excessive absences will result in lowered grades. Excessive absenteeism, whether excused or unexcused, may result in a student's course grade being reduced or in assignment of a grade of "F". Absences are accumulated beginning with the first day of class.

STUDENT ACADEMIC APPEALS PROCESS

Authority and responsibility for assigning grades to students rests with the faculty. However, in those instances where students believe that miscommunication, error, or unfairness of any kind may have adversely affected the instructor's assessment of their academic performance, the student has a right to appeal by the procedure listed in the Undergraduate catalog and by doing so within thirty days of receiving the grade or experiencing any other problematic academic event that prompted the complaint.

ADA STATEMENT

Students with disabilities, who believe they may need an academic adjustment in this class, are encouraged to contact me as soon as possible to better ensure receipt of timely adjustments.

DEFINITION OF CHEATING AND PLAGIARISM

CSUDH is dedicated to a high standard of academic integrity among its faculty and students. In becoming part of the California State University academic community, students are responsible for honesty and independent effort. Disciplinary action will be taken against any student who alone or with others engages in any act of academic fraud or deceit. (Read University Regulations in University Catalog)

COURSE OUTLINE

Week 1: Overview and introduction to game design tools

Week 2: Team Formation, project planning and Imagine Cup project plan submissions

Week 3: Repurpose existing game to meet needs of project document

Week 4: Classes and constructors in games, continue game design

Week 5: Variables, textures and art in games, continue game design

Week 6: Advertisement in games, how to set up a market account

Week 7: Project Management

Week 8: Playability and usability

Week 9: Using results from playability and usability surveys

Week 10: Generating test patterns, how to use testing to make sure changes don't break the build

Week 11: Version control, documentation

Week 12: Quality control, designing for the long run

Week 13: Team Foundation Server

Week 14: Showing off your game

Week 15: Pricing your game and improving proposals

Week 16: Demonstrations