JOURNAL OF OBJECT TECHNOLOGY

Online at http://www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2007**

Vol. 6, No. 8, September-October 2007

Introductory Game Programming Instruction with OOP - What is required, How is it addressed, and Which language wins?

Lakshmi Prayaga

Abstract:

This paper analyzes three popular programming languages (Adobe/Macromedia Flash, Java and Visual Basic.net), used to teach introductory 2D game programming courses. Presented in this paper is a discussion on the parameters required for the course, followed by a comparison on how each language addresses these parameters, and a suggestion on which language is the winner in this race.

This paper analyzes three popular programming languages (Adobe/Macromedia Flash, Java and Visual Basic.net), used to teach introductory 2D game programming courses. A discussion on the parameters required for the course is followed by comparing how each language addresses these parameters, and ends with a suggestion on which language is the winner in this race.

1 INTRODUCTION

Serious games or interest in using video games as an effective means to deliver instruction, and provide training for specific skills, has been increasing across the board, thus creating a high demand for game developers. Smith (2006) reports that several game industries including Vtech, Stottler Henke and Pokemon, are developing games for serious purposes which provide training for military personnel, medical professionals, and educators. A recent study by Crandall & Sidak (2006) reports that by 2009, the need for game programmers is estimated to be around 250,000. To keep up with this demand and to also cater to the .net genners' who are people born between 1970 and 1990 (Oblinger, 2004) interests in video games (Gee, 2003), many secondary and higher educational institutions have included gaming as a minor or major in their curriculum. The question then arises, what is an appropriate programming language for introductory game programming courses? Three of the more popular programming languages, particularly for 2D gaming, are Adobe/Macromedia Flash, Visual Basic.Net, and Java.

Cite this article as follows: Lakshmi Prayaga: "Introductory Game Programming Instruction with OOP - What is required, How is it addressed, and Which language wins?", in *Journal of Object Technology*, vol. 6, no. 8. September – October 2007, pp. 51 – 66
http://www.jot.fm/issues/issue_2007_09/column5

This paper analyzes the strengths and weaknesses of these three languages for teaching introductory 2D game programming. Specifically a discussion, is provided on the choice of language (Edginton & Leutenegger 2007), its relation to what is required in game programming (Prayaga 2005), how each language addresses these requirements, and which language is the winner. The three sample programs used for the purposes of this paper are provided as attachments with this document. The sample java program is modified from an available online java tutorial, the Visual Basic.net version is an adaptation from Prayaga 92005) and the Flash program is modified from Moronta (2003) to fit the scheme of this paper.

2 CHOICE OF LANGUAGE

We chose Flash, Visual Basic.Net and Java to specifically show that though all three languages are OOPLs, which is very important for game programming, each language may be best suited for a particular audience. The audience are categorized into three groups, students enrolled in the introductory, (students taking a first programming course), intermediate, (students who completed data structures) and advanced courses (students who are taking Operating Systems and Networks courses). Therefore a one size fits all approach is not the best approach. For purposes of discussion, this paper will use the example code from the attached programs written in all three languages.

3 COMMON FEATURES OF THE SOFTWARE

It should be noted that the output from all the three programs written in the different languages is almost identical except for color choices, size of game assets and sound files, see figures 1, 2 and 3. Each of the three programs has

- graphics, sound and animation
- a player and some enemies
- Collision detection and response - The player-enemy collision creates a shattering effect. The collision between enemies just bounces the enemies off against each other in all the three programs. See figures 1, 2 and 3 for a sample output of each program.

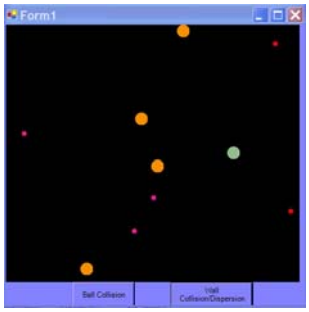

Figure 1 VB.Net runtime

Figure 2 Java runtime

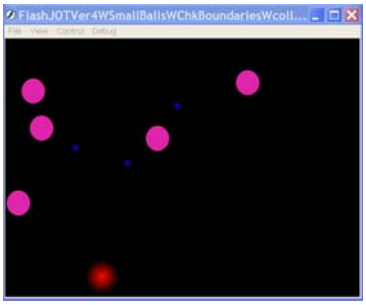

Figure 3 Flash runtime

The differences in these programs are in the design aspects, which make one language more suitable than another for a specific audience. The rest of the paper will discuss the three questions:

1. *What is required in Game Programming?*
2. *How does each language address these requirements?*
3. *Which language is the winner for introductory game programming?*

4 WHAT IS REQUIRED IN GAME PROGRAMMING:

Game programming is a phase of game development. This phase includes the choice of the programming language, based on the following features:

- Capability of the software to create game assets (graphics and sound files) or use existing game assets required for the game
- Support for features such as collision detection, AI, and math capabilities of the language to match the requirements specified in the design of the game
- Support to Build a GUI
- Animation
- Portability and extensibility

- Provide for a game loop

Section 5, provides a discussion on these requirements.

5 HOW DOES EACH LANGUAGE ADDRESSES THESE REQUIREMENTS:

5. 1 Create or use existing game assets:

Both Visual Basic.Net and Java have standard APIs and libraries (System.Drawing and java.awt) which can be imported and used for several tasks such as creating graphics objects and sound objects required for the game. Graphics are at least as important as the story line in the game development phase (Phelps, 2005). Flash has a major advantage in this aspect. It does not explicitly require the use of APIs or external libraries to accomplish the above mentioned routine tasks. These features are all built-in and can be used directly. Flash allows the student to create graphics including freehand graphics from the tool box and convert them to buttons or movie clips which can then be added to the library, for use in the game. All this can be done in the IDE (Integrated Development Environment) for Flash as shown in figures 4, 5, and 6. Figure 4 shows a circle object created by the ellipse tool from the tool box, Figure 5 describes the process of converting this graphics object to a move clip and adding it to the library. Similarly you can create fancy shapes (Figure 6) with tools from the tool box and convert them to movieclips. Flash also allows you to import any external irregular shaped image files to the library and convert them to movie clips too. So we see that there is a great deal of flexibility in using Flash for graphics objects. The corresponding code required to create the graphics object, circle, is the Cball class, provided in the example code blocks in java and VB.Net which is created with code, not with a GUI interface. Creating such user defined classes requires that the student must be comfortable with the java or VB.net graphics model at an abstract level which is not easy in a first game programming course without any previous background.

One other advantage in Flash, is that the movie clips are the concrete objects, that students can relate to. These movie clips can then be instantiated by simply dragging them to the stage or use methods such as “attachMovie” or “duplicateMovie” to attach them to the screen at run time depending on the need in the program. Flash also makes it easy to add properties and methods to these concrete objects in the main program. Hence the level of complexity with regards to abstraction is minimal which is important in an introductory game programming course. Students can also manipulate the objects and use them with a minimal learning curve which boosts the confidence levels in students (Keller, 1987). Flash thus allows the student to easily deal with the graphics objects without any previous programming experience.

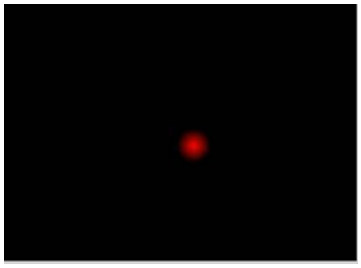

Figure4 (instance on stage)

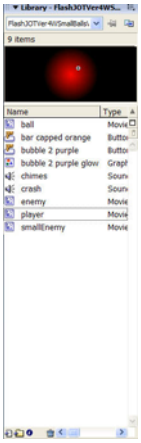

Figure 5 (movieclip in library)

Figure 6 (complex graphics)

5.2 Collision detection:

The Collision detection feature is extremely important in game programming since this is what allows the player to continue in his mission by either gaining points or losing points. In order to implement this feature in the sampleVB.net and Java programs, we included the functions *hit* and *handleCollision* for the java program and *collide* for the Vb.net program to detect collisions between any two objects. The java program used the *center to center* distance algorithm, while VB.Net used the *intersects with* method to detect collision between two objects.

Flash has a built-in function hitTest() which returns a Boolean value (true) when two objects collide. The hitTest function in Flash implements the bounding box collision algorithm vs. the pixel collision detection algorithm. The bounding box collision detection algorithm is not very precise, but is sufficient for the beginning game programming student. The hitTest function in Flash works both for geometric shapes as well as irregular shapes, since it draws an invisible rectangular box around the object and checks for collision between these invisible boxes at run time. Vb.Net and java can also

accomplish the same effect with the built-in *intersects with* method. However, the intersects with method is strictly limited to circle and rectangle objects, and does not extend to irregular shapes which is very confining to the student. The hitTest() function is also very simple to explain to introductory programming students. The only prerequisite knowledge required to explain this concept is the notion of "conditional statements" which is covered pretty early in the semester. Hence we see that conceptually collision detection algorithms at the level of bounding box collisions are simple in all three languages and can be covered in a first semester introductory course with the understanding that java and VB.net limit this feature to rectangle and circle objects.

5.3 AI and physics for games:

The programming language must also support facilities to build in some AI and physics to create realistic games. The three languages discussed here have the required math capabilities built into the language to generate a simplistic version of AI and physics for the games. Since AI is an advanced topic, it was not implemented in the examples provided for this paper.

5.4 GUI interface:

User input is important for any event driven programming, but more so in game programming (Pereez 2006). Java has some popular editors such as Eclipse, but using the IDE is not simple drag and drop as is the case with VB.Net or Flash. VB.Net. has Visual Studio.Net as an editor, which allows the drag and drop feature. Flash on the other hand, is an environment in itself to create Flash movies, games, graphics and the corresponding GUIs. But it is to be mentioned that all three languages offer a great deal of flexibility to

- design the GUI and
- get user input, process it and provide feedback

Java requires that you import javax swing class to design a GUI and an extra class (import java.awt.event.*) to detect mouse events, such as mouse click, mouse enter etc. In addition, the programmer is also required to write individual methods for detecting each of the mouse events. Examples include methods for mouse enter, mouse leave, mouse coordinates etc. VB.Net and Flash do not require importing external classes to design the GUI or for the purposes of capturing user interaction with the mouse. Both languages have these methods built-in to the language and can be used directly without having to specifically write the code to create the mouse object.

5.5 Animation:

Animation is a crucial requirement in game programming. This can be achieved through the use of timers or threads as is shown in the sample code for java (used threads) and vb.net (used a timer). Flash however does not support the use of threads, but the use of frames in Flash can be used effectively to achieve the required animation (Adobe blogs, 2006). The example code included in this paper shows how the frame rate in Flash works

as a timer to animate several objects on the screen at the same time. It is important here to identify the method that can be easily used to introduce all the elements of the game loop to an introductory game programming student who has not had many programming classes or a strong CS background. The use of frame rates vs. timers vs. the use of threads is an excellent example for this discussion. At this introductory level, the simplest approach is to explain this idea of animation through the use of frame rate followed by timers. The example program in Flash uses the frame rate (see actions for in frame1), to animate the balls on the screen according to some set interval for the frames, the default being 12 frames per second. The idea of timers used in the VB.net example (see Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) can also be explained very easily by using the timers tick event which contains the code to animate the objects . However in the example java program, the concept of threads was used to animate the objects, (see public class WPlayerV2 extends Applet implements Runnable) even though java supports timers as well, in which case the level of complexity is similar to that of VB.net. The concept of threads is not a simplistic concept and its purpose and use is generally not discussed in an introductory programming course. The concept of threads is usually discussed in an operating systems course and is required in games which require parallel processing and concurrency. Threads is not a topic that can be introduced in an introductory game programming course.

Flash also has another attractive feature – shape tweening, which is similar to morphing when one shape changes to another shape at runtime. This can be used very effectively in games to create special and stunning effects with minimal effort, which leave a lasting impression on the student's mind (Georgenes, 2005). Both VB.net and java do not provide this frame based shape tweening which makes Flash a winner in this aspect.

5.6 User Experience:

User experience is the result of a motivated action in a certain context. In a beginning game programming class, the student has very high expectations of what can be achieved by the end of the semester. These expectations are a result of the student having played several games before enrolling in a game programming course. The motivation levels are therefore very high for the student to implement several aspects of those games in the course on game development. The choice of the right programming language plays a major role in helping the student achieve these high expectations. Flash offers several features such as

- a simple graphics model that allows the student to create freehand graphics directly in the IDE
- a built-in collision detection method that covers both geometric and free hand graphics and
- a frame based environment for animations which mimics a formal timer object

which makes it easy to create a user experience that is both motivating and enjoyable. Java and VB.Net do not provide a simple graphics model, or a built-in collision detection

method to deal with irregular shape objects. Based on these parameters, it is evident that of all the three languages being evaluated, Flash offers a user friendly environment which enables the student to achieve high levels of satisfaction in the first game programming course.

5.7 Provide a game loop:

Unlike computer applications such as a word document or a software program which respond to user input or remain idle, a game program is one in which something happens all the time regardless of user input. Examples include enemies moving on the screen, the timer ticking, or collision incidents. A game loop is what enables this continuous activity. Some of the phases of a simplified game loop are presented below (Dunlop, R., (2000), Igda, wikepedia)

```
while( user doesn't exit )
  check for user input
  provide response
  move enemies
  resolve collisions
  draw graphics
  play sounds
  animations
end while
```

Table 1 describes each of the phases in the game loop and compares how each of the three example programs address that phase. From table 1, we see that the game loop is executed almost identically in all three languages. But from the point of view of getting an introductory programming student with the required student preparedness to implement the entire game loop, Flash scores the highest rank (28), compared to Vb.net (19 or 23 (for timer)) and Java (17 or 19(for timer)) for reasons explained above.

5.8 Support for Portability and Extensibility of games:

The last aspect that will be addressed with respect to what is required in a game programming language is the extent to which that language supports portability and extensibility of games created in that language. Both Java and Flash have a major advantage in this aspect, since java files can be embedded into applets which can run on any browser, and the same is true for Flash files which can be converted to .swf files which can also run on any browser since most browsers have the required plug ins built in to run these files. The files created in the .net frame work however are not as portable. In order to run a .net file on a client machine, the client machine should either have the .net framework installed on it or the program can be packaged in such a way that it checks the client machine for the .net framework and if it does not exist, the executable file installs the required .net framework to enable the running of the program.

6 WHICH LANGUAGE IS THE WINNER TO TEACH INTRODUCTORY GAME PROGRAMMING?

The question that this paper attempts to answer is, if the three languages discussed in this paper, can accommodate the requirements of a game, then, what is the basis for choosing a particular language to teach game programming at an introductory level? The answer can be based on three parameters:

1. *What matters to the introductory student*
2. *What is the preparatory knowledge required for the student to achieve what matters to him.*
3. *What kind of user experience did this language provide to the student to further motivate him to continue game development as a career.*

6.1 What matters to the introductory game programming student?

What matters to the introductory game programming student is the implementation of the game loop in a clear and simple manner. This includes creating the game assets, check for user input, render and update graphics. From the discussion above, all three programming languages were shown to be capable of delivering these features but each language had its own unique way to respond to the requirements, some more simplistic than the others.

6.2 What is the preparatory knowledge required for the student to achieve what matters to him

Student preparatory knowledge was also minimal to use Flash for creating game assets such as graphics, animations, collision detections and responses. With Flash, students could accomplish all these aspects of game development in the first semester. However as shown in table 1, it would take at least two to three semesters to accomplish the same tasks with Java or VB.net.

6.3 What kind of user experience did this language provide to the student to further motivate him to continue game development as a career.

User experience depends on question 1 which is what matters to the introductory game programming student. As mentioned above Flash provides an environment to accommodate all the requirements of a 2D game program in the first semester, which makes the student very satisfied. Edgington and Leutenegger (2007) also confirm this observation. These authors have observed that user experience with Flash is positive. Flash was used as an introductory game programming language for the game development track in the Computer Science department at Denver. Students in this course are reported to have a very exciting experience in creating the game assets. Students also find that Action Script is relevant because they see its applications in the real world with its use in web site development, advertisements etc. This shows that the medium of instruction is important to provide user satisfaction and motivate students. Game development is one such medium that the current generation of students can relate to and

identify with. These authors reported that the game development track has resulted in student satisfaction, increased student enrollment and in student retention.

7 CONCLUSION

Java and VB.Net are good programming languages, but have a higher learning curve and "require significant scaffolding" (Edgington and Leutenegger 2007) before anything interesting can be done. Action script which is the programming language used for flash games is an easy language for students to use, which encourages and motivates them to continue with courses in game development and also consider majoring in computer science. It is for this reason that several colleges and universities are using Flash as an introductory game programming language (Chaffey College, Michigan State University). For these reasons and the fact that many practitioners and educators such as Argent etal (2006) and Leutenegger (2006), are suggesting the use of Flash as an introductory game programming language, we conclude that Flash is a winner for this niche.

Game loop phase	Vb.net	Java	Flash
Check for user input and provide response	User prompted to choose *wall* or *ball* collision, execute appropriate block of code based on input. Conceptual level of difficulty (5, requires no external classes). Student preparedness, 1st semester Lines of code (button click function 3 to 4 lines)	User prompted to choose *sound on* or *sound off*, execute code based on input, execute appropriate block of code based on input Conceptual level of difficulty (4, used java.awt class) Student preparedness, 1st semester Lines of code (mouseDown function, 3 to 4 lines)	User prompted to *start* the program, execute appropriate block of code based on input Conceptual level of difficulty (5, requires no external classes) Student preparedness, 1st semester Lines of code (button on release function, 3 to 4 lines)
Move enemies	Player ball and enemy balls start to move on the screen and continue to do so. Conceptual level of difficulty (2, due to design and use of a user defined class, objectShape, which is an	Same as in Vb.Net Conceptual level of difficulty (2, due to design and use of a user defined class, cBall which is an abstract	Same as in VB.net Conceptual level of difficulty (4, reference is made to the concrete object enemy or player ball, and reference to

	abstract object for the student; and reference to width and height of picture box and the ball object) Student preparedness, 2nd or 3rd semester Lines of code (function moveAround, 8 to 10 lines)	object for the student; and reference to width and height of applet window and the ball object) Student preparedness, 2nd or 3rd semester Lines of code (function move, 8 to 10 lines)	width and height of the stage) Student preparedness, 1st semester Lines of code (function checkBoundaries in the eneymyball or playerball object, 8 to 10 lines)
Resolve collisions and play sounds	When collision occurs against the wall, the balls bounce back. When collision occurs between the player ball and an enemy ball, the latter further shatters to smaller pieces. When collision occurs between enemy balls they bounce off of each other. In all cases a sound clip is also played. Conceptual level of difficulty (3, due to the need for creating a new function – collide and use of inherent method `IntersectsWith`, to check for collision between two objects) Student preparedness, 2nd semester	Same as in VB.Net Conceptual level of difficulty (2, due to the need for creating two new functions – 1. hit and 2. handle collisions to check for collision between two objects. This includes explanation of finding center to center distance between the two circular objects.) Student preparedness, 2nd semester	Same as in Vb.net Conceptual level of difficulty (5, since the only function required is a built-in function: hitTest() which is based on the principle of a conditional statement. But it should be noted that the hitTest() only provides a bounding box collision, hence it is not precise enough for detecting collisions between objects other than rectangles. Of course, Flash allows the programmer to write code for more precise collision detection. Student preparedness, 1st semester

	Lines of code, for Collide function: 10 to 15, and 4 lines for the use of intersectswith function.	Lines of code, for hit function: 4, and 10 to 12 lines for the use of handle collision function.	Lines of code for the hitTest, 1 line and 6 lines of code to handle collisions
Draw graphics	The balls (player and enemies) get repainted at regular intervals Conceptual level of difficulty (3, due to the fact that the student has to create a class in this case, the ObjectShape, with properties of size, color, shape and velocity and methods to set and get the values for these properties. The next phase involves the instantiation of the object, with the properties and finally use the paint method to draw the abstract object as a concrete object on the screen. Student preparedness, 2nd or 3rd semester Lines of code for the ObjectShape class 220.	Same as in VB.net Conceptual level of difficulty (3, due to the fact that the student has to create a class in this case, the CBall, with properties of size, color, and velocity and methods to set and get the values for these properties. The next phase involves the instantiation of the object, with the properties and finally use the paint method to draw the abstract object as a concrete object on the screen. Student preparedness, 2nd or 3rd semester Lines of code for the Cball class is 130.	Same as in VB.net Conceptual level of difficulty (5, since there is no necessity for an external class, the circle object can be drawn by the ellipse tool from the tool box, and concerted to a movieclip. The movieclip can be dragged onto the stage which creates an instance to which properties of size and color can be added. There is no need of a paint method to draw the graphics on the screen. Student preparedness, 1st semester Lines of code required to draw an object on the screen is 0 with the drag and drop, and 4 to 5 lines of code to use the attachMovie() or duplicateMovie() method and assign properties of size, shape, color, velocity and the x, y coordinates for this object.
Animation	Support timers and threads Conceptual level of difficulty (1 for threads	Support timers and threads Conceptual level of difficulty (1 for threads	Does not support threads Conceptual level of difficulty (4 for use

	and 4 for timers) Student preparedness, 2nd semester for timer, 3 rd or 4th for threads Lines of code for threads or timer: 15 to 20 to create the timer/thread, define the methods	and 4 for timers) Student preparedness, 2nd semester for timer, 3 rd or 4th for threads Lines of code for threads or timer: 15 to 20 to create the timer/thread, define the methods	timer, 5 for frames) Student preparedness, 1st semester Lines of code for timer, 3 to 4
Portability and extensibility	Must have .Net framework installed on the client machine, not available by default, to run a VB.Net executable file. Student preparedness, 1st semester Conceptual level of difficulty (to create an executable, 5) Lines of code 0	Use of Applets to run files on any browser, as long as it has the java runtime software installed (most new machines have it by default) hence very portable and extensible Student preparedness, 1st semester Conceptual level of difficulty (to create an executable, 5) Lines of code 0	.swf file runs on any browser that has the Flash plug-in installed on it, again many new machines have this installed on it. Student preparedness, 1st semester Conceptual level of difficulty (to create an executable, 5) Lines of code 0

Table 1 (Comparisons of Game Loop Phases and Game Requirements)

The source files and code for this article can be found at
http://www.uwf.edu/lprayaga/gaming/jot1.html

Key: conceptual Level of difficulty: 1- most difficult and 5 - easiest.

REFERENCES:

Adobe-blogs(2006), retrieved on Nov 26th 2006 from: http://blogs.adobe.com/kiwi/2006/05/as3_programming_101_for_cc_cod_1.html

Adobe, (2006), retrieved on Nov 20th 2006 from: http://www.adobe.com/products/director/resources/integration/flash/when_director.html

Argent, L, Depper, B.; Fajardo, R.; Gjertson S.; Leutenegger, S., T.; Lopez, M., A.; and Rutenbeck , J.(2006), "Building a Game Development Program," *Computer*, vol. 39, no. 6, pp. 52-60.

Chaffey College, retrieved on Nov 26th from: http://www.chaffey.edu/bat/cis/chaffcis/game.html

Crandall, R., & Sidak, J., (2006), Video Games Serious Business For America's Economy, retrieved on Dec 20th from: www.theesa.com/files/VideoGames-Final.pdf

Dunlop, R., (2000) Writing the Game Loop, retrieved on April 1st, 2007 from: http://www.mvps.org/directx/articles/writing_the_game_loop.htm

Edgington, J., Leutenegger, S., (2007), A games First Approach to Teaching Introductory Programming, *SIGCSE'07*, March 7-10, 2007, Covington, Kentucky, USA.

Gee., J. P., What Video Games Have to Teach US, Palgrave Macmillan; 1st edition (May 16, 2003)

Georgenes, C. (2005),Animation: Flash Shape Tweening Techniques, retrieved Dec 20th, from: http://www.adobe.com/devnet/flash/articles/tween_macrochat.html

Igda, Programming & Technology/Game Programming Patterns for Beginners/Game Loop, retrieved on April 1st 2007 from: http://www.igda.org/wiki/index.php/Programming_&_Technology/Game_Programming_Patterns_for_Beginners/Game_Loop

Java Tutorials, retrieved on Nov 26th from: http://javaboutique.internet.com/tutorials/Java_by_Example/section6_10.html

Keller, J.M. (1987a, Oct.). "Strategies for stimulating the motivation to learn. *Performance and Instruction*, 26(8), 1-7. (EJ 362 632)

Leutenegger, S. T. 2006. A CS1 to CS2 bridge class using 2D game programming. *J. Comput. Small Coll.* 21, 5 (May. 2006), 76-83

Moronta, Lewis., 2003. Game Development With Actionscript, Course Technology Ptr Michigan State University, retrieved on Nov 26th from: http://dmat.msu.edu/degrees/gamespecialization.html

Michigan State University, Specialization in Game Design and Development, retrieved on Nov 20th 2006 from: http://dmat.msu.edu/degrees/gamespecialization.html

Mobile Games, (2006), Mobile Game User Experience, retrieved on Dec 26th from: http://flashlitemobilegames.blogspot.com/2006/04/mobile-game-user-experience.html

Oblinger, D. (2004). The Next Generation of Educational Engagement. *Journal of Interactive Media in Education*, 2004 (8). Special Issue on the Educational Semantic Web. ISSN:1365-893X , retrieved on April 16th 2006 from: www-jime.open.ac.uk/2004/8

Phelps, A., (2005), Got Game? retrieved on Nov 20th 2006 from: http://gotgame.corante.com/archives/2005/06/13/graphics_dont_matter_and_other_assertions.php

Prayaga, L: "Game programming – The "Why", "What" and "how" with Graphics Objects", in Journal of Object Technology, vol. 4, no. 9, November-December 2005, pp. 39-58 http://www.jot.fm/issues/issue_2005_11/column5

Preez, H., (2006), Discovering the API Using Visual Basic 6 and Visual Basic .NET, retrieved on Nov 20th 2006 from: http://www.codeguru.com/vb/gen/vb_general/ideincludingvisualstudionet/article.php/c11981/

Smith, P., (2006) Serious Games Initiative, retrieved on April 26th 2007 from: http://www.seriousgames.org/index2.html

Serious Games: http://en.wikipedia.org/wiki/Serious_game

Wikipedia, Game Programming, retrieved on Nov 20th 2006 from: http://en.wikipedia.org/wiki/Game_programming

About the author

Lakshmi Prayaga is a faculty member in the Computer Science department at the University of West Florida (UWF), Pensacola. She has a Masters in Software Engineering from the University of West Florida, a Masters in Business Management from the Alabama A & M University. She has recently defended her dissertation on the "Game Programming to Teach Computer Science Concepts". Lakshmi teaches the Internet Programming courses, Visual Programming and Game Development courses. Lakshmi is also a software consultant to DCF in Pensacola, where she works on web and database applications.

Lakshmi's research interests lie in the role of computers and technology in education and Web Applications. Recently, (January 2007) Lakshmi was the PI for a partnership 1.5 million dollar grant that was awarded by the Florida Department of Education, to the Escambia County School District and UWF to develop games to teach Mathematics to middle school students, with the focus of game design being, the relevance of Mathematics in Career choices. Lakshmi is also developing a gaming curriculum at the University of West Florida.