Intermediate Milestone 2 (Spring 2018)

Mohammad H. Salim

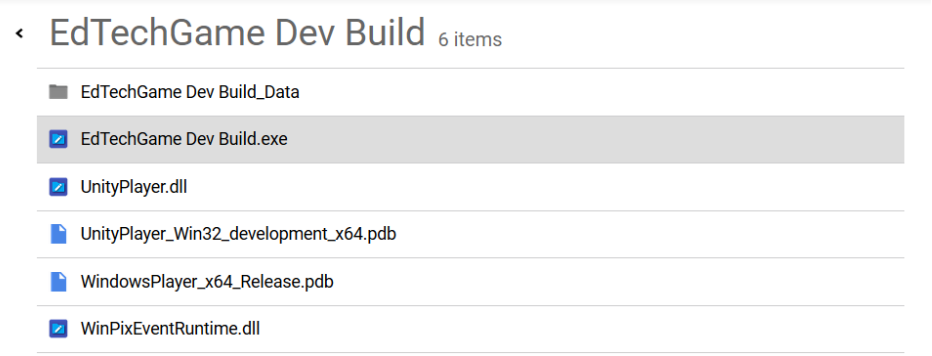
Msalim7@gatech.edu

Intro

I am developing a 2D semi-text based immersive educational game to teach beginner-level programming to adults utilizing balanced game design. If you are interested in the context, I will discuss the theory I'm testing followed by my initial proposal plan later in the paper.

Prototype

You can download the prototype (currently in the work). Any feedback would be great! You must unzip the download and run the EdTechGame Dev Build.exe file.



EdTechGame Dev Build 6 items

- EdTechGame Dev Build_Data
- EdTechGame Dev Build.exe
- UnityPlayer.dll
- UnityPlayer_Win32_development_x64.pdb
- WindowsPlayer_x64_Release.pdb
- WinPixEventRuntime.dll

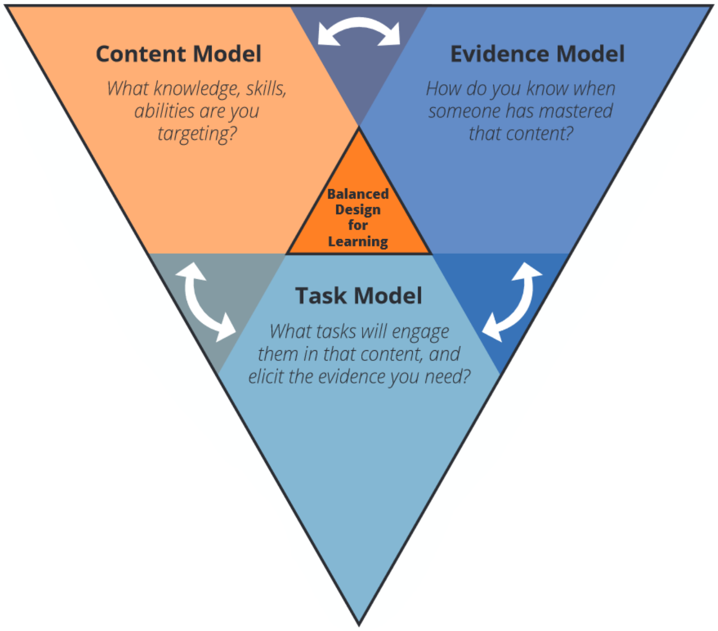
Here is a set of known bugs and considered enhancements:

Summary	Additional Notes
Hint system is useless for runtime or compiler errors.	
Lack of music.	Considered adding a control to play a set of royalty free music. Button for skipping songs and turning music on or off.
Code is not beautified.	The InputField isn't friendly with RichText styling (only styling support in Unity) as the player is typing. Additionally, it gets tricky to run the code since I must carefully parse out RichText styling that wasn't written by the player.
Code sanitization.	Most players will not write dangerous code on their own machine. Additionally, a lot of resources and files cannot be accessed via Unity/IronPython combination, so it would be difficult to accidently write dangerous code.

Untested on diverse set of machines	I’ve only had time to test on a few Windows machines.
Top-down canvas where player can walk around.	This would give the game a more game-like feel over the current tutorial-like feel. The computer screen canvas would still be used when the player uses an in-game computer or terminal.

Theory

Balanced Game Design[1] (BGD) a learning design approach that should be integrated with whichever game design approach you go want such that it better aligns learning goals with the game mechanics. This not only will help drive engagement, but also make the serious game more effective. BGD builds off the Evidence-Centered Design (ECD) which you can see in the diagram below.


Content Model
What knowledge, skills, abilities are you targeting?
Evidence Model
How do you know when someone has mastered that content?
Balanced Design for Learning
Task Model
What tasks will engage them in that content, and elicit the evidence you need?

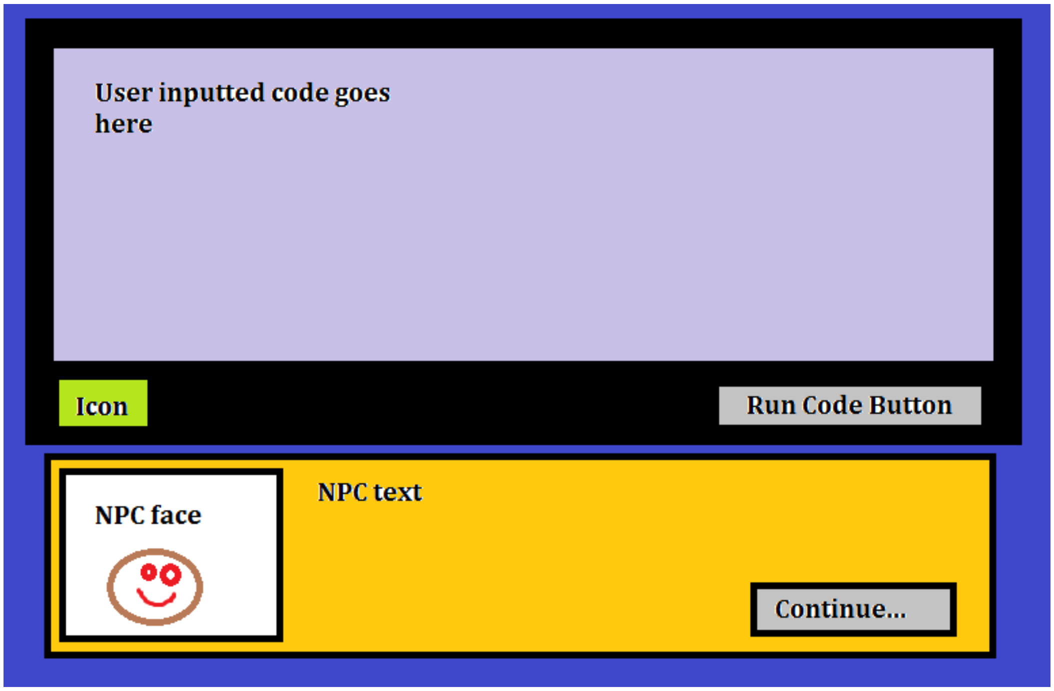
ECD focuses on the measurable success of a student’s learning. BGD has three models: content, task, and evidence. Here is a set of examples for my use case:

Model	Definition	Example
Content	What skills we teach the student.	Beginner-level programming for adults.
Task	Which goals does the student need to complete.	Playing a video game.
Evidence	Knowing if a student has mastered the content.	Completing the game or how well the student has completed the game.

The Content/Evidence/Task models seem intuitive enough to measure, so I will be using BGD as my primary guideline in development. Additionally, I want to have a focus on Immersion from the I-Feature[2] framework because I think that's the most important feature that isn't really covered by Balanced Game Design.

Tasks

This is my initial mock up of the game. It has two main components: code terminal and NPC dialogue box.



There are four scenes which will have a series of lessons. Each scene has a very simple but silly plot. The player is a student in a programming class and must complete assignments at the start. The player will be betrayed by a classmate pretending to be a friend. After the player makes up for a bad grade, the player will get in contact with an anonymous hacker. I don’ think I will be able to complete all four scenes by semesters end, but if I did, the remainder of the story would involve blackmail by the anonymous hacker, getting involved with the government to fight off black hat hacking Martians, and an ancient space mortal programming competition.

Scene	Plot	Lessons
S1	Your classmate, Broseph the Brogrammer, needs help on programming homework.	Console output, variables, and arithmetic.
S2	Professor Gevabad Graids hands out a difficult assignment. He expects a printed out copy of the assignment.	Strings and arrays.

	Due to your part time job, you won’t be able to hand in the assignment. Broseph insists to drop it off for you.	
S3	You receive a 0. When you ask Professor Gevabad Graids, he says that the TA never received your assignment. Broseph received a 100%. When you ask Broseph about what happened, Broseph says he doesn’t remember offering to drop off your assignment for you. TA Ova Wurkt offers an extra credit assignment as a chance to make up for the 0.	For loops.
S4	Ova is impressed by your work. You later get a message from @n0n_h@ck3r asking if you want a chance to get revenge on Broseph. If you can prove to @non_h@ck3r that you are a decent programmer, then he/she will help you get back at Broseph.	Methods and combining previous lessons.

This was my initial attempt at BGD document. I originally wanted to do a C# tutorial set, but Unity has an existing bug where it fails to compile code on a build (.exe) version of the game (whereas it works fine during the in-editor run). I had to switch to Python, so the BGD is slightly outdated. I plan to update it for the final proposal.

Content	Task	Model
Learn basic form of outputting to console.	Fix spelling error in “Hello Wrld” program. Then have it to say right after: “Goodbye World”.	**Correct**: Player learned to write two lines of text to the console. **Incorrect**: 1. ***No change within quotes.*** Player does not understand where the actual text is. 2. ***Only one line printed out.*** Player replaced original string with new string instead of printing one after the other. 3. ***Text has incorrect casing.*** Player did not understand the strict nature of programming (such as case-sensitive strings).
Variables	Four variables will appear on screen: an integer, a double, a string, and a Boolean. Print the four variables one line after the other.	**Correct**: Player learned how to initialize, change, and print four different primitive variable types. **Incorrect**: 1. ***Variables did not all print.*** Player does not

	Create four more variables (one of each type with a specific value), then print those. Change the value of the integer variable from the very top, then print it out.	understand that you can pass a variable to System.Console.Writeline. 2. ***Incorrect types for the four variables created.*** Player does not understand the strict typing requirement when declaring variables. 3. ***Incorrect value for integer.*** Player does not understand that you can re-initialize a variable after it's declaration and initial initialization.
Arithmetic	Summation for two variables will appear. Change this to multiplication and print. Subtract a number in-line (without creating a variable) during the multiplication and print. Do the subtraction first before multiplying using parenthesis and print.	**Correct**: Player learned how to do arithmetic while considering order of operations. **Incorrect**: 1. ***Initial output isn't multiplied.*** Player does not know the arithmetic character for multiplying or does not know where to change the arithmetic. 2. ***Second output has an unnecessary variable.*** Player does not know how to use value in-line. 3. ***Second variable is incorrect.*** Player does not know how to subtract or does not know how to handle two operations. 4. ***Third output is incorrect.*** Player does not understand how to use parenthesis to enforce precedence.
Strings	Two string variables will appear for a first name and last name. Create a full name by "adding" (concatting) the two strings and print.	**Correct**: Player learned how to concat strings and use built-in string methods. **Incorrect**: 1. ***First or second output is not concatted.*** Player did not use + operator

	Now add a space between the two strings (full name with correct format) and print. Use built in methods: Substring, Replace, and IndexOf. Print results.	correctly or there is a confusion with the usage of + being both concat and add. 2. ***Second output is missing space.*** Player does not know how to concat a string in line. 3. ***Third output is incorrect.*** Player does not know functions since this is the first exposure to it or does not understand that C# is zero-indexed.
Arrays	Given an integer array with initialized values, variable with length property, variable with index. Print these. Change a value at a certain index and print. Initialize an empty array. Print length. Print at a certain index. Change at a certain index and print.	**Correct**: Player has learned two ways to initialize arrays, index arrays, and check the length property. **Incorrect**: 1. ***Initial output is not printed.*** Player has forgotten or not correctly learned to write lines to console. 2. ***Array index value is not correct after change.*** Player is has not gotten familiar with the indexing syntax. 3. ***Array index value is correct but in the wrong index.*** Player has forgotten C# is zero-indexed. 4. ***Length output is incorrect.*** Player is familiar with methods but not properties. May have added parenthesis.
For Loops	Given a for loop, print the number at each iteration. Make a second loop that is incrementing. Make a third loop that is decrementing.	**Correct**: Player has learned to use for loops that go in incrementing or decrementing order. **Incorrect**: 1. ***All elements do not print.*** Player has not place print

		statement inside the braces of the loop. 2. ***Loop never ends.*** Player has put a logical expression that causes an infinite loop. 3. ***Incrementing loop does not increment.*** Player does not understand the ++ operator. 4. ***Decrementing loop does not decrement.*** Player does not understand – operator.
Methods	Given a method that is passed an integer array and returns the product the first two elements. Print the result. Change method to loop entire array and add each number. Then append the returned value to a string and print.	**Correct**: Player has learned to write a method and been tested on previous tasks. **Incorrect**: 1. ***First output is invalid.*** Player does not understand the return concept of a method. 2. ***Entire array in not looped.*** Failed to master for loops and arrays. 3. ***Summation is incorrect.*** Failed to master variables and arithmetic. 4. ***Printed string is incorrect.*** Failed to master strings.

In order to create a minimum viable product that could test BGD before semester's end, I would need to complete the following tasks:

Task ID	*Task*	*Estimated Hours*	*Type*
T1	List of problems	10	Design
T2	Story-problem integration	10	Design
T3	Hint/failsafe system	5	Design
T4	GUI for input text.	4	Programming
T5	GUI for output text	4	Programming
T6	GUI for character	6	Programming
T7	Parser for Language	10	Research & Programming

T8	Success/failure response	4	Programming
T9	Scene System	14	Research & Programming
T10	Complete Scenario 1	8	Programming
T11	Complete Scenario 2	6	Programming
T12	Complete Scenario 3	6	Programming
T13	Complete Scenario 4	6	Programming
T14	UI Improvements	10	Design & Programming
IM1	Intermediate Milestone 1	4	Presentation
IM2	Intermediate Milestone 2	4	Presentation
F1	Final Presentation	4	Presentation
F2	Final Paper	4	Paper
	119 Total Hours		

This is the schedule I have set. Although I underestimated the workload (probably due being relatively new with Unity), I still am on track!

Deliverable	Week of	Tasks	Estimated Total Hours
Weekly Status 1	02/26/2018	T1, T4	14
Weekly Status 2	03/05/2018	T2, T5	14
Weekly Status 3 **Intermediate Milestone 1**	03/12/2018	T7, IM1	14
Weekly Status 4	03/19/2018	T3, T6	11
Weekly Status 5	03/26/2018	T8, T9	18
Weekly Status 6 **Intermediate Milestone 2**	04/02/2018	T10, IM2	12
Weekly Status 7	04/09/2018	T11, T12	12
Weekly Status 8	04/16/2018	T13, T14	16
Final Project **Final Presentations** **Final Paper**	04/23/2018	F1, F2	8

References

1. Groff, Jennifer; Clarke-Midura, Jody; Own, V. Elizabeth; Rosenheck, Louisa; Beall, Michael. (2015). *Better Learning in Games*. http://education.mit.edu/wp-content/uploads/2015/07/BalancedDesignGuide2015.pdf
2. dos Santos A.D., Fraternali P. (2016). *A Comparison of Methodological Frameworks for Digital Learning Game Design*. In: De Gloria A., Veltkamp R. (eds) Games and Learning Alliance. GALA 2015. Lecture Notes in Computer Science, vol 9599. Springer, Cham