J. Matthew Griffis
Major Studio 1
2012-11-28

How to Play - a Major Studio project

(the breakdown)

The following is the documentation and analysis of my project for the Instruction Sets from Strangers *theme, a video game I created called* How to Play*, which is available for play here. I will summarize the project and suggest how it succeeds, critically analyze the project and suggest how it fails, describe in minimal technical detail some of the challenges I faced and overcame in developing this project, and provide an annotated bibliography of the many sources I drew upon in creating* How to Play*.*

In Summary (the Good)

I took a class in high school about game design theory and read a book[1] in which the author examined various ways of communicating the game's instructions to the player. While I took issue with the author's criticism of some of my favorite games for their instruction methodology (more out of partisanship than actual disagreement), I found the topic fascinating. I was reminded of it when I played a game recently called *Nebulous Hero*, in which the game's tutorial appears at the end of the game instead of the beginning. With no instructions to go on, I played *Nebulous Hero* based on its aesthetic, which reminded me of other games I'd played and thereby implied I should play this game like those games. However when I got to the end of *Nebulous Hero*, I

[1] *Game Design: Theory and Practice* by Richard Rouse III

earned one page of the tutorial, and learned to my surprise that I'd been playing the game incorrectly. When I replayed it based on the new information, I earned another page of the tutorial and discovered that another of my assumptions was wrong. By the time I had earned the full set of instructions and had finished the game as it was "meant" to be played, I'd had a radically different experience than my first play-through and the game's appearance suggested.

I loved this conceit, which not only played with the ambiguity of an unexplained system but actively exploited it to highlight and subvert some of the tropes of video gaming. The game also richly capitalized on the satisfaction of moving through a system with increasing certainty about its inner workings. So when in class we read Tausk's *Influencing Machine* in preparation for the *Instruction Sets from Strangers* project and discussed the assumptions inherent in instruction - including that the instructor is trustworthy - I decided that I wanted to create a video game about instructions: one that was inspired by Richard Rouse III's critique, functioned similarly to *Nebulous Hero*, and featured a narrator of dubious reliability.

Fortunately I already had a prototype. For my CC Lab class I had created a simple game in which the player navigated a character around a two-dimensional world, jumping from platform to platform. It would be easy to expand this basic mechanic to be the first mode of the game - an unseen and somewhat condescending narrator would provide the player with instructions about how such a game works, and then at the "end" the player would discover the instructions were misleading and that the game should be played entirely differently. By playing the game over again with this new information, the player could reach the true ending.

I did not initially have a clear idea of what the new way to play the game should be - some vague concepts of different game mechanics but no certainty about what might be fun (or, for that matter, technically feasible). Conversations with classmates raised several important points for consideration: a) it is crucial that the game should reward replay with the new information; otherwise it's just a one-shot "gotcha!" moment at the end of the first mode; b) the more different ways of playing the game I could cram in, the more interesting the game would be; c) it would be interesting to use unconventional cues, such as a pained sound effect to note an incorrect action (which would otherwise go unremarked), thus rewarding the player's careful attention. On the whole people received the idea positively and their feedback was helpful in honing my vision.

I had a great deal of fun writing out and coding in the instructions that would be delivered textually during the first mode of the game. I drew upon the sarcastic, condescending and somewhat inhuman tone of the antagonist from the video game *Portal* for inspiration, and I spent a lot of time positioning the text on screen so that the progression of the text would have some visual dynamism, thereby making up some for its otherwise static presentation.

By the time I had finished the game's first mode, I had a fairly clear idea of how I wanted the second set of game mechanics (the alternative way of playing the game) to function. It needed to work well with the platform layouts I had created for the first mode of game play, layouts which increasingly had taken on visual significance to correspond with the instructions delivered on each screen. As had been the case from the very beginning in CC Lab, there were technical challenges to overcome in order to make

everything function the way I wanted (i.e. “How on Earth do I code this?”), but with persistence I was always able to find a way.

Originally it was my intention to conclude the project with the second game play mode, time having run short. However, with the onslaught of Hurricane Sandy and the pushback of the deadline, I decided to move the goal posts, so to speak, and add another level of depth. I devised a third way to play the game, the nature of which would not be explained to the players. It would be up to them to figure out what they had to do - drawing upon their experiences in the first two modes - and by doing it to reach the game’s true ending. Furthermore I created a special reward for players who made it to the end and were especially observant. This was especially satisfying to make because some of my favorite games are those with content that many players will never see - hidden rewards for only the most thoughtful and persistent explorers.

Having executed all the game play mechanics and modes, I crafted a title screen and an ending screen, into which I put an extra degree of visual polish and interactivity (in the form of how players move past these screens). I posted the game on the website Open Processing and asked a classmate if he would be willing to aggregate and summarize people’s responses to the game. He having agreed to do so, I posted in the class blog about the game and asked my classmates to play it and then send whatever reactions they wished to share to my aggregator, who would give me an anonymous report of the results.

I am extremely happy with the way *How to Play* turned out. I was able to execute on the vision I had from the beginning of the project, essentially exactly as I had envisioned it. I overcame a tremendous number of complex coding challenges, thereby

demonstrating to myself that I could achieve whatever I wanted in code. I created an experience with legitimate depth, complexity and polish, and filled it with humor and surprise. I was even able to add in the kind of secret content that characterizes the games I love the most. *How to Play* is my triumph.

Self-Critique (the Bad)

How to Play fails in several key areas. These are: fun; challenge; clarity; aesthetic. While there is a great deal of overlap between these four (and in some ways they cannot be separated), I will look at them on an individual basis. It is worth noting that as of right now, more than a week after posting the game and requesting people's feedback, I have - with one exception - heard nothing. The game has received 151 views so far on Open Processing (of which only a small fraction can likely be attributed to me) and no one has left a comment or sent me - or my aggregator, as far as I know - an email. It is certainly possible a great deal of feedback has found its way into his inbox and he simply hasn't told me yet, but I'm inclined to think people are not engaging with the experience as I'd hoped. Furthermore, no one has yet reached the end of the game. This is discouraging and plays into the assessment of failures, which I now turn to.

First and foremost, *How to Play* fails as a game. It is more functional as an artistic and technical interactive experiment, insofar as a defining criterion of games is that they should be fun. It is not especially fun. I may be biased, of course, having played the game so many times I cannot evaluate it with any degree of objectivity, but I do know that I designed the various mechanics, level layouts and other aspects of game play to work towards my larger narrative, mechanical and metaphysical goals for the

project, rather than incorporating them because they were inherently fun. “Fun” is a sticky term, of course, being hard to define, but suffice to say that ensuring the player was having a good time was never a priority for me, and there is little reason for people to return to *How to Play* once they have reached the end - or just gotten sick of the experience. The game play mechanics do have the potential to provide fun given the right context, which segues nicely into the next area of failure, challenge.

How to Play fails to provide the correct degree of challenge. In some ways it provides no challenge and in other ways it provides far too much. The complete failure so far of anyone to reach the game’s ending is a strong indicator of the latter. Indeed the one gift of feedback I received post-publishing expressed confusion about whether the game has an end at all, and if so how to reach it. While the game shepherds players through most of the experience in mode 1, they are left mostly to their own devices in mode 2, and entirely so when attempting to reach the ending. The game’s challenge comes from figuring out what to do to progress to the game’s next stage, and it is likely I made the clues too cryptic.

On the other hand, once players realize what they have to do, completing the game’s mechanics poses almost no challenge, since there is not really a way to fail or much in the way of opposition from the game system. One notable exception is a particular set of platforms that proved troublesome for players to navigate (as I discovered early on in CC Lab), to the point where they were giving up rather than completing the level; nevertheless I did not adjust the layout to make it easier to complete, and it’s possible this hangup is at least partially responsible for the absence of game completion. Challenge and fun are closely intertwined in games - creating the

appropriate degree of challenge requires providing obstacles complementary to the game mechanics, and fun occurs when it is neither too easy nor too hard to use the game mechanics to overcome those obstacles. Obstacles are sorely lacking in this game. It is the absence of challenge that hurts the game *qua* game the most; that being said, I should lay some blame at the altar of clarity (or rather, obscurity).

How to Play fails to provide sufficient clarity. Mystifying players is all well and good, and making a puzzling experience that only the best and brightest will solve is a great goal, but if no one sees the end, why did I bother putting so much work into it? It is important that some people succeed, even if it is equally important that not all of them do - otherwise it's less likely that I created clever clues and more likely that I created no clues, or at least inadequate ones. In retrospect this was easily the most ambitious goal to take on, since crafting a good mystery is no easy task, and finding the right balance between telling too much and telling too little is every bit as hard as creating the right degree of challenge. I'm very proud of the game's ending, which makes its possible irrelevance all the more frustrating. This failure bothers me the most.

How to Play fails to deliver a compelling aesthetic. I'm OK with how the game looks, but my experience building the title screen showed me how much more could be done. The title screen is very striking, with its appearance of a dimly lit room, the use of interesting fonts and variable font size, and the splash of color in the pull handle for the bulb. The game itself looks bland by comparison - its colors are few and muted, there's little variation in the appearance of the text, and there's simply not much to look at besides the player-character with its simple blinking animation. I'd hoped to add shuffling feet to the character but ran out of time (and that would have been its own

coding challenge). The game and its starting and ending screens don't really have any visual relation to each other, and the justification for the light bulb motif is vague at best (the first line of text from the narrator says "there you are," as if the lights had been turned on). That's the visual side of things, crying out for more graphics, more color, more vibrancy, more life.

On the aural side of things there's nothing - no music, no sound effects, no sound at all. While providing voice-over for the narration would probably not be a realistic goal within the constraints of the project, it would have been nice to add some effects. I incorporated a music track to my final project for Bootcamp so there's no reason that I couldn't have done so here. Unfortunately I put sound on the backest of the back burners, and the result is a silent experience, which really detracts from its tangibility. Even something as simple as a sound to accompany the character's jumping would instantly make the game more dynamic by providing another type of connection between the player's actions and the game. Games don't need sound to be enjoyable (as anyone who has ever muted a game can attest, they may actually become more so when silent!) and neither do they need sophisticated visuals, but there's no denying that the presence of a rich aesthetic does a lot of the legwork in drawing players in and compelling them to keep playing.

Finally, while I did not include it in the list above, *How to Play* fails on the research front. I include both traditional research (reading and playing the work of previous practitioners) and play-testing (crucial to game design) under that label. It's arguable that my experiences reading the game design theory book back in high school and playing *Nebulous Hero* constitute a sort of research, even though I did not do them

with this project in mind. Certainly I did some research into various coding solutions when trying to solve the many technical challenges that sprung up. But I did not during the course of this project seek out previous work in the area of the project's goals in order to shape my own vision. Whether this is a true failure is debatable - I knew more or less exactly what I wanted to make from the get-go, and I made it. Still there's no denying it could have been a richer execution had I sought more influences to draw upon.

More to the point is the issue of play-testing, or rather the lack of it, which is definitely a failure on my part. Play-testing is essential during the game design process because the designer cannot be objective about his own work; it is only by putting the product - whatever its current level of completion - in front of an audience that is new to it that the designer may see what is working in the game and what is not, and then use that feedback to shape continuing development. I fell into the common trap of "it's not done yet!", feeling that time was of the essence and that it was more important to rush ahead with development and get the game done, rather than to take the time to let people play it and see whether it was having the desired effect. I did let a housemate play the game at one point, and while she made it through mode 2, she played it in a different way than I had intended, which robbed the experience of much of its pleasure, a phenomenon that should have set off immediate warning bells in my head about clarity. Nevertheless, I barreled onward.

Truth be told, I didn't really care about play-testing, for the same reason that I didn't really worry about whether the game was fun - this was a project I made for myself to see if I could do it and to express my own interests. Other people's reactions

didn't have much bearing on that. That being said, now that the game is done I want people to enjoy the finished product and be impressed by my skill, a reaction that would be much more likely if I had addressed the failures described above. There's no question that play-testing would have been invaluable in that pursuit. In this light, *How to Play* is as much a failure as it is a success - it succeeds in expressing my vision but fails to communicate it to other people.

Down to the Nitty-Gritty (the Ugly)

I had intended to communicate something of the nuts and bolts of the project here, but several things occur to me:

a) I've already gone on a bit long with the two sections above;

b) I thoroughly (some would say too thoroughly) commented the game's code and explained many of the concepts I would describe here, so anyone who is interested can check the code via the link at the top of this document;

c) No one is interested - why would they care about my coding methodology for generating level layouts? But seriously, if they do care, the link is above;

d) I completed the game, and it speaks for itself (however competently). I don't want to spoil the game's secrets here or elsewhere (although, again, anyone who wants to cheat can look at the code).

This being the case, I believe I've covered the project's origins, motivations, evolution, and execution above, and the technical details are available in the code and its commentary. I don't think there's anything left to explain in this section, but I invite anyone who wishes to learn more to email me at grifj153@newschool.edu.

Bibliography (the Bibliography)

The following is a list of all the sources I can think of whose work I drew upon in creating *How to Play*. It does not necessarily represent the complete list - and I apologize to anyone I left out - since I failed to keep proper records of this nature over the course of the project (that should probably have its own failure section). Still, I noted all my technical help in the code commentary and described the spiritual help above, so here's a reprise of what I can remember:

Game Design: Theory and Practice (by Richard Rouse III) - I will never forgive Richard Rouse III for criticizing *The Legend of Zelda* for how it instructs players...but he's right. The book is about game design theory in general but includes a section on how to provide instructions.

Nebulous Hero (video game; download here) - born out of a challenge to create a game where the tutorial comes up after the credits, *Nebulous Hero* cheerfully and brilliantly subverts expectations to deliver a brief but unforgettable experience.

Dorkshop: Intro to Game Programming - Ramiro Corbetta's Bootcamp Dorkshop taught me how to use Booleans to generate smooth movement in video games, a trick I've been using ever since.

Dorkshop: Intro to Object-Oriented Programming - Jane Friedhoff's Bootcamp Dorkshop taught me the basics of OOP, which the additional coverage in CC Lab only built upon.

Francisco Zamorano - my CC Lab instructor made various contributions: his in-class example was the basis for my player-character, he assisted me in incorporating that character into my existing game framework, and he provided guidance in solving various technical challenges along the way.

Jennifer Presto - one of my Bootcamp fellows, Jennifer helped me to understand better through one of her code projects how to use arrays, and how one might add or subtract objects from an existing array. While I did not end up using this method for my own project, the additional understanding was extremely helpful.

Mauricio Sanchez - another of my Bootcamp fellows, Mauricio provided me with a code example of his own that helped me to understand ArrayList, a technique that proved essential for creating the levels in *How to Play*.

Portal (video game) - *Portal* features its own set of instructions to the player, the completion of which leads to a different outcome than expected...but it was the sarcastic, condescending, detached tone of antagonist GlaDOS that influenced my writing for *How to Play*. Alas that I couldn't replicate the voice. Or the cake.

Every Day the Same Dream (video game; play here) - this exceedingly depressing game is a good example of replaying a scenario multiple times with new information each time, leading to different results (sort of).

Processing.org - the Processing website offered lots of help, from the definitions of functions in the Reference section to the user-submitted solutions to various problems in the forums.