RISE OF THE VIDEOGAME ZINESTERS

HOW FREAKS, NORMALS, AMATEURS, ARTISTS, DREAMERS, DROPOUTS, QUEERS, HOUSEWIVES, AND PEOPLE LIKE YOU ARE TAKING BACK AN ART FORM

ANNA ANTHROPY

SEVEN STORIES PRESS
NEW YORK

Chapter Five

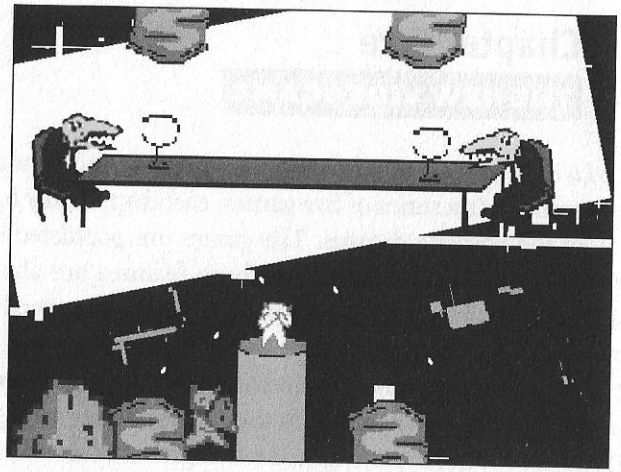
THE NEW VIDEOGAME

The *La La Land* series was created by Matt Aldridge under the alias "The Anemic." It's a series of five games, each supposedly based on one of the author's dreams. The games are populated with recurring scrawl-doodle characters whose features are abstract enough to be sinister. The protagonist, Biggt (pronounced like "bigot"), has a smile that's Glasgow-wide and a pair of eyes that don't line up. The music is often seemingly incongruous: the climax of *La La Land 5* happens with the anti-evolution song "I'm No Kin to the Monkey" playing in the background. And the arbitrary goals the games give the player are almost always revealed to be against the protagonist's best interest.

The first one I stumbled upon was *La La Land 2*. In this game, Biggt is confronted by a towering snaggletooth fish head, baby pink and jittering, which tells him: "it's ok to steal from the rich cos they havelots [sic] to spare!" A counter—which in most games would be nailed to a fixed point on the screen, but in this game jiggers and jumps from point to point—reads "$0.00."

Biggt travels left to a screen where crow-like "nobles" with fish heads sit at opposite corners of a long table. A figure that resembles Biggt is on a pedestal, face in his hands. As he cries, some of the tears pass through pipes and fill the nobles' wineglasses. At the far left is a hay-colored, shapeless pile that

represents gold. Biggt scoops up an armful, making the pile smaller, and carries it back to the initial fish head, adding money to the counter. The fish head says, "Oh thanx biggt i want MORE."



The player runs back and forth between the fish head and the nobles, redistributing wealth. When the player has stolen enough money, the fish head rises up to reveal a gold medallion hanging from one of its bones. "Look at the shini JEWELS I bought thanks. a lot." Returning to the nobles, the player finds them dead at their table. "without money how were, the nobles to surivve?" The game then closes.

Personal and abstract, The Anemic's *La La Land* games were the first games of their kind that I encountered, and they would never have been made if not for Game Maker.

I'm not sure when *La La Land 2* was created, but in 2003, when I was seeking out the other games in the series, I discovered a game called *Seiklus* by an admirer of the *La La Land* games. The author, cly5m, had made worlds for Tim Sweeney's *ZZT* that were as dreamlike as *La La Land*, if a bit larger in scope. (One, *Kudzu*, involves making a collect call to heaven at a melting payphone and visiting a hall filled with imprisoned invisible people.) American by birth, cly5m had visited the country of Estonia on a religious mission, then returned to America and discovered Game Maker. *Seiklus* is the Estonian word for "adventure."

Seiklus begins with a boy and girl watching the stars on a cliff. A meteor hits and causes the boy to fall into a valley below. The game ends when he has found his way back to the girl. To return, the player has to collect colored tokens, which are found in a giant hollow tree, on a quiet, snow-covered landscape, in a dark land full of bones, and inside the stomach of an enormous creature. All of these scenes were hand-drawn by cly5m himself; the thick line art style is as unmistakably his as the collection of pastoral-alien dream scenes.

To the West

From my current perspective, I can say that what I always wanted to do was make videogames, but as a child, making videogames was something mystical that I never thought I'd be capable of it. So I funneled my creativity into other channels. I was better at writing than drawing, so I was a creative writing major in college.

After playing *Seiklus*, I figured that if cly5m could make the transition from making add-ons for *ZZT* to making self-contained games in Game Maker, then I could do the same.

I had taught myself to program a little, particularly in a language called Blitz Basic, in which I had recreated a few games that I knew. I had made a board game–like version of *PAC-MAN* where the player and the ghosts took turns moving rather than

moving at the same time. Making objects move and react to each other all at once was something that was beyond me, and objects' placement on the screen had to be hard-coded, making it difficult to experiment with actual design. Game Maker was refreshingly similar to *ZZT* to me: every individual object had its own instructions to tell it what to do and how to react to the other objects. I could draw every object, give it instructions, put it wherever I wanted to on the screen, and let the program play out and resolve interactions based on how I'd told the objects to behave.

My first Game Maker game was called *Jaywalker*. In *Jaywalker*, the player controls a disembodied head named Marjorie with a HotHead Paisan–style crown of orange hair and a stud in her nose (I think I made the game around the time I first got my nose pierced.) Marjorie is fiercely about the rights of pedestrians, and is committed to destroying as many cars as possible. The player moves her back and forth across a busy intersection, causing cars to swerve around her and crash into each other. Cars will crash into cars that have already been wrecked, creating huge pile-ups and causing Marjorie to stick out her tongue triumphantly. If she's ever hit by a car, the game is over.

I dropped out of college some time after creating *Jaywalker*. When I quit, I was in my fourth year of school with no end in sight. I didn't see myself as a writer, I hated school, and I thought (and think) that "higher education" is bullshit.

I spent another year in New York, where I was born. I made some more games during that time. Terrified I'd be stuck in the same spot for the rest of my life, I decided I needed a reason to move somewhere far away and a way to earn rent, a trade that I could learn while I was there. The thought of earning my living by making games was exciting: why not go to a

games college in another part of the country, perfect my craft, and find a way to pay my rent by making videogames?

Specialized videogame schools had been around for a while at this time. I had known of DigiPen, in Seattle, since I was a teen, and had recently heard of Full Sail, a movie school in Florida, but both of those seemed to me like game programs tacked on to visual arts schools. I chose the Guildhall at Southern Methodist University because it was the only school I discovered that had a level design focus. (A Game Maker game I made after quitting creative writing school, *Invader*, which was about an alien from the game *Space Invaders* crash-landing on a weird planet and having to find her way off, was part of my portfolio.)

The Guildhall was in the middle of Plano, Texas. Plano, Texas, is brown and not much else. They have a Frito-Lay factory, parking lots, and a videogame school. At the time, I kept a strict vegan diet and didn't drive. There was nothing to eat and nowhere to go.

But the latter didn't matter; when you were at the Guildhall you had no life outside the Guildhall. I remember the first day of orientation, sitting in a lecture hall with my future classmates and the spouses they'd brought with them to this wasted brown land. One of the other level design students had his wife and their year-old child with him. "Give her a kiss and say good-bye," the director of the school told him in front of the assembly. "You're not going to see her for two years."

I was in Plano, Texas, for six months.

You're at school from nine to five. You stay after and do your work with the teams they've assigned you to. Late at night you drag yourself home and do your actual homework. Maybe you get a few hours of sleep. The idea behind the school is that you're always in what the Big Games Industry

calls "crunch time": unpaid overtime. Your masters want the game done by Christmas, so you don't leave the office until it's done. This is why people in the industry aren't healthy; this is why they burn out and quit games within a few years. This is why you miss the second year of your daughter's life. This is their scheme: you put up with crunch time all the time while you're in school, so when you work for a big publisher—or, rather, a studio contracted by a big publisher—you won't complain about being told you can't see your daughter until the game's done. The Guildhall boasts an over 90 percent employment rate, and it's true: they will get you a job in the games industry. That's because they will make you into exactly the kind of worker the games industry wants. It's that kind of school.

And it works; that's the horrifying thing. My classmates were all self-identified gamers and game fans and were willing to put up with anything in order to live their dream of making videogames. That's the carrot the industry dangles, and it's what we take away from the industry when we create a form to which anyone can contribute. As long as the industry is allowed to continue acting as the gatekeeper to game creation, people will continue to accept the ways in which the industry tramples the lives and well-being of the creative people who make games, rather than challenging the insane level of control that publishers ask over developers' lives.

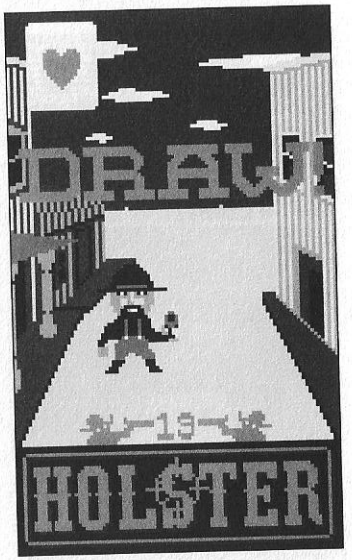
Needless to say, I was not at the school long. I butted heads with lecturers too many times, I asked too many questions, and my *Oblivion* mods were too experimental (I didn't include messages that popped up and told the player she'd achieved her goal because I didn't think the player was too stupid to realize that.) Eventually I was pulled into the director's office and asked to leave. "If your unit is marching to Fort Worth, you don't ask

'what if we go to Austin instead.'" That's a paraphrase of what the school's director told me.[54] I've had a while to think about this, and I've since decided that if you know there's an army waiting to ambush you at Fort Worth, it's fatally irresponsible not to try to dissuade your comrades from marching there blind.

I said that I was in Texas for six months. Each semester at the Guildhall was just over two months long; they canned me the first day of the third semester. I spent the last month or so getting my stuff moved back to New York and making a game called *Calamity Annie*.

Calamity Strikes

Stranded in Plano, Texas, with everyone I knew occupied with school all of the day and most of the night, working on *Calamity Annie* is what kept me from going stir-crazy until I flew back to New York.



The protagonist and namesake of *Calamity Annie* is a brash young dyke who rolls into the Old West (from the New West, naturally) with her breasts bound and a pistol in her holster. She tears across the countryside, dropping hombres in one-on-one shootouts. Along the way she meets a lady named Valentine with a different past but the same breed of loneliness. If the player's sharp enough, Annie and her Valentine ride off into the sunset together at the end.

The whole game is played with the mouse, by pointing to aim and clicking to shoot. A duel consists of "holstering" your cursor at the bottom of the screen, waiting for the call of "DRAW!" and quickly aiming and firing before your opponent can do the same. Interactions with Annie's love interest use the same controls. The first time you meet her, Valentine will ask for a light for her cigarette. Aim at the cigarette, click, and PEW! Flirtation has begun.

It's a hard game—the quick-draw contests get quicker and quicker, and eventually the player has just milliseconds to react. But even when the player gets GAME OVER and Annie spills some blood, Valentine patches her up, and their relationship goes on. What's between them outlasts any individual firefight. Every play of the game isn't self-contained but is—in a small way—fitted into the ongoing tale of love and connection, which the gunfights are just a means to developing.

Calamity Annie is a lot of things to me. It's about being an angry young woman in a hostile land (that just so happens to be Texas), trying to prove your worth. I certainly was trying to do that after they kicked me out of the Guildhall. I was making a game, and I was doing it my way: I built every piece of the game myself. I drew all the pictures; I wrote all the music; I scripted all the events in Game Maker. If the Guildhall taught me anything, it was that videogames needed

to be saved from the industry, that a creative form deserves better than an assembly line production process.

But the game is also about love, and how passion finds love. I had started seeing a loud-ass submissive named Daphny Drucilla Delight David shortly before starting at the Guildhall, and the workload had kept me away from her except for one week of vacation between each semester. The time we got to spend with each other while I was at school was mostly limited to me calling her and crying from the stress of having to split my energy between standing up to teachers, finishing my work, and doing good work, when all I wanted to do was make games.

I'm writing this, three years later, from the apartment where I live with her in California. The theme of *Calamity Annie* is that being driven will drive you to love, and that passionate people are attracted to passion. *Annie* was one of the first in a series of games that led to my being able to pay for a home and food by making games, which is what I went to Texas to seek. But it didn't come from the Guildhall. It came from me, from my stubbornness, and the stubborn friends who helped me.

Later games have made me more money—I asked players who wanted to support *Calamity Annie* for a donation of at least a dollar, in exchange for which they got a bunch of secret bad guys to shoot (I made a few hundred total)—but I still think

of *Annie* as my most important game. It's everything I realized I wanted games to be. It's personal (Annie has my name, after all) and it has a clear, unmuddied mechanical idea that translates perfectly to a storytelling idea. I want games to tell stories, and *Calamity Annie* tells my story, or at least the story of my stay in Texas.

Authors

My friend Lamar Williams is working on a documentary about videogame creators called *You Meet the Nicest People Making Videogames*. In one of the project's trailers, he points out that the idea of digital games as the product of teams is a myth: many of our most important games, and the ones that have been the most widely duplicated, are almost entirely the projects of impassioned individuals.[55] The examples that appear in the trailer are: *Another World*, a game about finding friendship during a struggle for survival in an alien world, created by Eric Chahi (who even painted the cover art for the game himself!); *Karateka*, a kung fu game with strong visual storytelling by *Prince of Persia* creator Jordan Mechner; *Defender*, a seminal arcade game by Eugene Jarvis and Larry DeMar; and *Berzerk*, almost as seminal an arcade game by Alan McNeil. He goes on to show contemporary games by individual authors, some of them made in Game Maker.

There's nothing unnatural about a digital game by an individual creator (or a pair of creators). It is, in fact, much harder to keep the idea behind a game coherent when the designer is managing a team of many people who are each working on one aspect of the game separately. That's part of the reason why contemporary big-budget games have so much clutter and so few strong ideas. The games are all over the place because the creators were all over the place. It's hard to have a

strong singular vision when the process of creation is spread so thin.

Digital games contain video, audio, animation, design, and rules. You can parcel out these roles, but the closer they remain to each other, the more cohesive the work you create. If I'm the designer and I'm also drawing the spaceship that appears in the game, I know exactly how I intend to use that spaceship in terms of play, what its place is in the larger story, and what its appearance should express. I have a vision, in other words.

Books are written by single authors or by author-editor teams. Visual art is typically made by an individual artist. It makes sense for creators to be close to their work and to own their work completely, and that's something that the big teams that big-budget games demand can't have. When an individual or pair is solely responsible for a work you can watch an individual style develop: you can trace themes, both mechanical and otherwise, across a creator's work. (The Anemic's *La La Land* games, though each game is very different from the rest, have a strong singular style that persists throughout the series.)

And being able (or learning to) identify the individual style, and growth, of individual authors leads to better criticism and a critical understanding of games. Not to mention, like I said, more personal games, more relevant games, more games with something to say. I want a world where everyone is capable of sitting down at a computer and making a game by herself. This is not to say that all games need to be made that way, but as a paradigm, I think the individual author has more to offer us than the team, especially at a time when videogames are so seemingly creatively bankrupt.

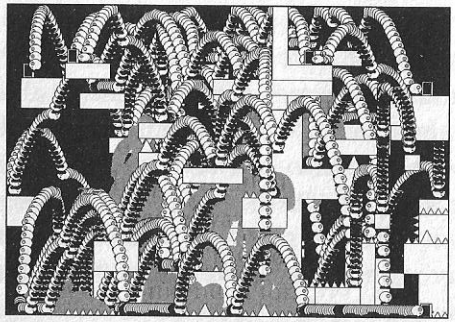
Death from Overwork

I first became aware of Jesse Venbrux, a game designer from the Netherlands, in 2008 because of his game *Execution*. When the player starts this game, she sees, through the sight of a gun, a prisoner tied to a stake. If the player shoots the prisoner, the prisoner dies, and the player is told, "You lose." If the player starts the game a second time to try to avoid losing, she again sees the same prisoner, still riddled with bullet holes.

Execution (a direct inspiration for *Calamity Annie*) points out something about causality in games: in particular, it points out how, as players, we expect to be allowed to undo and redo our choices at whim. The game takes just a few minutes to make this point: it's the kind of game that could never be a commercial product, but nevertheless has something essential to tell us.

Execution would probably never have been made without Game Maker. Jesse Venbrux might never have dug into game creation were it not for Game Maker. Though he's more recently gotten involved in Flash game creation and released games for the iPhone and its permutations, his website lists Game Maker–created games going as far back as 2004.[56] In Venbrux's body of work we can clearly see ongoing themes, both mechanical and contextual. His 2007 *Frozzd*, which was created for a Game Maker theme competition (entrants were asked to design games with a "winter" theme), involves navigating free-floating, irregularly shaped planets that all have their own gravity. That is, the player can walk all the way around the surface of one and arrive where she started, before jumping off and flying to another one. His 2010 game *Maru* features the same kind of irregular planet navigation, as does his most recent game (as of this writing), *They Need To Be Fed* for the iPhone.

His 2008 game *Deaths* remembers every time any player playing the game anywhere in the world meets death. Players will see the corpses of the last fifty players to have played the game, indicating particularly difficult areas and hidden traps. There is one screen in which players must continually die in order to build a bridge of corpses for future players to reach a high plateau (this is where the earlier-mentioned *We the Giants* gets its seed). His 2008 and 2009 diptych, *You Made It* and *You Probably Won't Make It*, also remember the player's death, painting patches of blood wherever she encounters a deadly spike. In the former game, this is especially important because the blood will paint over the visual information from the previous screen, showing what the terrain actually looks like. (In this game, nothing the player sees is ever erased from the screen. It is only overwritten by new information.)



The theme of protagonist death as narrative progress is an important one in Venbrux's work. His *Karoshi* series of games (*Karoshi* can be translated from Japanese as "death from overwork") require the player to guide the protagonist, a suit-and-tied salary man, to his death in a series of increasingly convoluted screens. Sometimes, in a similar vein to *Execution*,

the levels of *Karoshi* force the player to make use of something she thought was external to the game in order to proceed. For example, there's a stage in *Karoshi 2.0* called "What's On the Menu Today?" In this stage, the player must quit to the game menu—the screen where the player selects what stage she wants to play. The protagonist follows and falls to his death. Another stage requires the player to put a CD in her computer so that a boom box on the screen can play loudly enough to shunt a safe onto the protagonist.

Sure enough, Venbrux's iPhone game *They Need To Be Fed* requires the player to feed the protagonist to a monster at the end of every stage in order to complete the stage. (His 2008 game *Pazzon* requires the player to lead characters to the dinner tables of other game inhabitants.) It also features some of Venbrux's recurring "characters," such as the guided missile launchers that appear in his 2009 *Focus*, which in turn features the protagonist of *You Made It*.

We can notice and trace details and themes like these from game to game because each of these games is almost entirely the work of a sole, identifiable author. It makes us better, more literate critics of games to be able to see and discuss the progress of Jesse Venbrux as a designer. And it makes games more richly personal if we can play them in the context of the ongoing work and growth of a knowable author. This is a more useful paradigm than viewing games as the work of nebulous teams of hundreds.

The Author in the Games marketplace

At the time that Jesse Venbrux started making games, there was no place for him to market his work. Six years later, he's released a game for sale through the Apple Store. While this book isn't about making games for money—it's about making

games for the sake of having more games by more people, and there will always be more hobbyist authors than commercial authors—I think it's worth discussing the changing videogames marketplace in the context of the changing videogames paradigm. Just as digital games are starting to be thought of as the work of an author rather than a corporation, there is a growing place for solo game authors in the marketplace, and people can make a living doing this (though it's not easy).

I've been making my living in recent years by selling Flash games to "sponsors." Flash is an infrastructure for embedding movies and games in websites, and sponsors will pay for the privilege of including good games in their websites because games attract players who look at ads, and then explore the rest of the site to play more games and look at more ads. The more traffic a website gets, the more valuable that site is to advertisers, so traffic from popular games is important to site owners. Flash is much more complicated to use than something like Game Maker (although that's changing in a small way with tools like Stencyl, Flixel, and FlashPunk), and requires the kind of dedication that someone who just wants to make games in her spare time isn't likely to be able to spare. It's an avenue, though, for solitary authors to make the games they want and to find money for them.

The first Flash game I sold is about a flying pig. I sold it to *Newgrounds*—the "Everything, By Everyone" portal I mentioned earlier—for enough money to help me move from America's East Coast to the West. Later I sold a game called *REDDER* for twice as much, and while working on this book I sold a game called *Lesbian Spider-Queens of Mars* to *Adult Swim* for enough to keep me out of trouble for a while. I like this model because it means that although I get paid for my work, the game remains free to players. That's important, because

I want as few barriers between my work and its audience as possible.

Like Jesse Venbrux, I've also released an iPhone game, although mine is totally free. It's also based on a game I made in Game Maker—a friend of mine with greater technical knowledge, Bennett Foddy, programmed the iPhone version. The game's called *Chicanery*,[57] and it's about the interactions that go on between players outside of the digital components of the game (the stuff that's on the screen). Each player's goal is to keep a finger on the screen as long as possible, and to hinder the other players' abilities to do that. Usually the players do this by punching, shoving, or tickling each other. A guy I met who worked at WayForward Technologies later showed me the dent that playing the game had made in his iPhone.

When they're not free like *Chicanery*, iPhone games are usually sold for a dollar or two. And Apple has total control over its marketplace and what games can be sold within it, and is willing to wipe out games without a second word, as they did in 2010 with thousands of games they thought were too "sexual."[58] But the fact that authors are selling games there, and sometimes making a profit, shows that the digital games marketplace is making room for a new paradigm. Note that the Apple Store sells games exclusively through digital distribution—there's no publishing or manufacturing cost whatsoever.

Similar markets are Android phones, Steam for the computer, and the Xbox Live Marketplace. These are all digital distribution stores tied to specific technologies, and each is run by a corporation that exercises total control over what content is available on which device. It's a situation that leaves creators at the mercy of corporations, but it's a sign, at least,

that avenues are appearing for solitary authors to make money by creating games. And, hopefully, there will soon be more avenues, and decentralized ones.

Some authors take it upon themselves to sell and distribute their own games. My friend Edmund McMillen sold a CD with a bunch of his games and drawings on it—his attempt to create a game-as-zine. One of the things I used to do was to release a game that was free to play from start to finish—again, free is important—but to offer a secret password to anyone who donated at least a dollar. The password unlocked additional characters that would show up in the game as neat surprises. I got a few people who donated exactly one dollar, but most people gave five or ten: the invitation to donate was all they needed.

But that's enough about money. What I'm interested in is game creation as a goal in and of itself.

Crap Games

> *Glorious Trainwrecks* is about bringing back the spirit of postcardware, circa 1993. It's about throwing a bunch of random crap into your game and keeping whatever sticks. About bringing back a time when you didn't care so much about "production values," as much as ripping sound samples from your favorite television shows to use in your game, or animating pictures of yourself making goofy faces on your webcam. Where every ridiculous idea you had, you would just sit down and code. When you would make up a "company name" to legitimize dorking around on the computer with your friends. . . .
>
> Together, you and I will bring the true spirit of indie gaming back. Yes, you! For this site is about nothing, if it is not about getting off your ass and *creating*. Wikipedia claims that they used to stage trainwrecks (with

empty trains, of course) for the amusement of the general population. Would the world not be a better place if we brought this tradition back?

So states Jeremy Penner on his website, *Glorious Trainwrecks* (at glorioustrainwrecks.com), founded in 2007. The site is interested in a videogame application of the "Crap Art" philosophy[59]: that professionals aren't the only ones who should make art, and that creation is a meaningful goal in and of itself. Games are more wonderful, more creative, and more inventive when they're thrown together around an idea with little regard for production values or painstaking creative choices. There was no place to explore the videogame as an act of raw creation, so *Glorious Trainwrecks* was built to provide one.

On the third Saturday of every month, *Glorious Trainwrecks* holds an event called Klik of the Month Klub, after Klik & Play, the simple, ugly game creation system (designed for children and people who have never made games before) that most of the participants prefer. But there are no enforced rules about what tools people can use, what they should make, or how much time they should take, though it's in the spirit of the thing to finish the game, from start to end, in two hours.

The experience forces participants to get past their egos and their meticulous plans for future epic games, to stop focusing on details and CREATE. Klik of the Month is about doing, not planning. Having to finish a game in two hours keeps authors from getting hung up on lesser decisions about their games. This is why Klik & Play is so well suited for the exercise: it's full of clip art, stock sound effects, and design shortcuts. All the resources are already there: just get your hands dirty.

The results are usually sloppy and haphazard. That's what the glory is: having made something raw and improvised. During different Kliks of the Month, I made a version of *Pong* where touching the ball makes you lose points (*Dodgeball*), a game about escorting vulvas back to their home planet (*Box Pushing Game*), a game about pushing sumo wrestlers into basketball nets with your belly (*Sumo Dunk*), and a pile more. In 2007, Glorious Trainwrecks ran a two-day event (the 100-in-1 Klik & Play Pirate Kart) that produced one hundred games. We did it again in 2010 and produced 529 games by over a hundred people. People have the desire and creativity to make games; they're only stopped by the roadblocks that get put in the way. Tear down the roadblocks and we'll have so many more games—and more important, more authors.

Game Sketches

What will help to tear down those roadblocks and allow games to be more creative—in the sense of, "Let's create something!"—are tools that allow more spontaneity. Klik & Play comes with a bunch of clip art that the author can easily appropriate toward

other ends. Game Maker allows the author to draw sprites right in the program before giving them rules and actions to perform. Stencyl allows instant access to an online database of resources other creators have already made. That's the solution, I think: to put as little distance as possible between the idea and a playable game. Certainly, many authors will spend a lot of time adding details to their games and developing their ideas. But there's value in ideas, especially in a form so young, and those ideas need to be put out there.

There's no way to sketch in games. In half an hour, I can sit with a pencil and paper and draw some dumb little comic strip. Videogames have no easy equivalent to that. And yet dumb little games are important because they enrich our vocabulary of ideas. The ideal game-making tool, I think, would look like a sketchpad: I could draw a landscape with a character in it, then give that character rules about how she could move across that landscape.

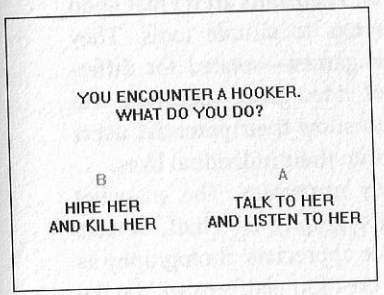
The editor that Fred Wood made for his game, *Love*,[60] comes close. *Love* is a simple little game about a stick figure who navigates a world by running and jumping; if she touches anything dangerous, she dies. To create a *Love* level, all an author has to do is draw it. The game will figure out how the stick figure interacts with the level. What stops the editor from being perfect is the fact that the author has to split the drawing into a few different images: a layer for things the player can touch and a layer for things that will kill the player on touch. But this is the right direction: if level design can be as simple as drawing a sketch, then game making can be as quick as putting an idea onto the screen.

And someone that doesn't know how to script or code probably knows how to draw—at least, to sketch. All of the most successful game-making tools for non-professionals I've seen

have looked like this. *Warioware: D.I.Y.* is probably the most mainstream commercial game-making software I've encountered, in that it was developed by a big publisher (Nintendo) for sale on store shelves. And while Nintendo makes it hard for people to actually distribute games they've made with the software (if the tool had a YouTube-like gallery where people could browse and play other peoples' creations, it would have changed a generation), the process of creating a game is brilliant.

Warioware is for the Nintendo DS, which has a touch screen and plastic stylus. Player-authors can draw sprites on the screen like they're drawing with a marker. Then they assign the sprites simple rules to tell them what to do during the game. But there are also a number of constraints that keep the author from being too ambitious: each game is only seconds long, the author only has a handful of objects to work with, and there's an equally limited number of sprites. This is a tool that's designed specifically to create very small games, and that's good, because small games are exactly what we need more of.

The proliferation of small games also serves another goal: it'll make games more personal. Ian Bogost, writing for *Gamasutra*, discussed the potential of games by way of talking about snapshots. The introduction of cheap portable cameras like the Kodak Brownie in the 1960s allowed people who were not professional photographers with access to expensive cameras and lenses to document anything, including things that held little or no value for anyone other that the taker:



> [Consider] You're Invited to Go to Heaven, a simple quiz game. . . . You're Invited is a rudimentary example of Christian evangelism.
>
> The game poses just a single question, "Who is the Lord of your life," and offers four answers: Chris Brown, Orlando Bloom, Zac Efron, and Jesus Christ. The "correct" choice is obvious, and it's tempting to write off this game as trite, even worthless.
>
> Its single question would seem barely to qualify it as a quiz game, a genre itself on the very fringes of the medium.
>
> But there is something deliberate and honest about its simplicity: this is not a game meant to inspire conversion or even head-scratching; it's just a little touchstone in someone's day for reinforcing what's really important to the believer.[61]

It might be hard to see the value in a thousand versions of *You're Invited to Go to Heaven*, but the real value of people producing digital games as quickly and easily as photographs is more subtle:

> The Brownie teaches us that snapshots aren't just good pictures created easily thanks to simple tools. They are also good pictures—or games—created for different purposes. The future of video game snapshots will require platform creators to show their potential users how to incorporate games into their individual lives.
>
> The result could be very important. The snapshot didn't just popularize photography as chaff, it also helped more ordinary people appreciate photography as craft. The successful game creation platform will be the one we can say the same of, someday.[62]

Making games more pervasive—not just games, but game

CREATION—will help us to better appreciate games and think about the craft and design of them. It'll also, I think, demystify game creation. We won't think of games so narrowly because we'll understand that they're capable of telling many different stories: stories about dyke cowgirls getting their girlfriends off booze, for example, or a record of a dream.

The ideal game-making tool, the game sketchpad, isn't here yet, but there are many tools that are close—or at least much closer than what engineers had to work with back in the day. But even given that, how do you, with little to no game making experience, get your feet wet and your hands dirty? How do you make a game?

Chapter Six
MAKING THE GAMES

How do you become a game author? By making a game! Get an idea, find some tools, and make something you can play. You can improve your craft later. At the start, just getting some experience putting together a game and making creative decisions about it will be incredibly valuable. I'm going to describe a few of the smaller, simpler games I've made and the process by which I made them.

Title, Interaction, Resolution: *Gay Sniper*

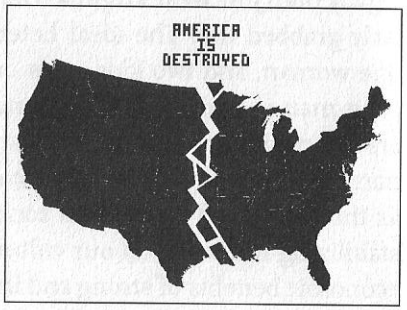
I made *Gay Sniper* in May of 2009 during a Klik of the Month Klub, the two-hour game design contest I described in the last chapter. I think I made it in about an hour. It was inspired by a video made by West Virginia 4 Marriage,[63] an organization that absolutely does not want queers to gain the right to legally marry in West Virginia. One scene in the video particularly grabbed me: The ideal heterosexual family—one man, one woman, and two kids—sits on a stoop blowing bubbles. (The man is white and the woman brown, so the filmmakers can position themselves as a anti-racist organization.) The narrator: "Marriage between a man and a woman is held up as the ideal in all of civilized society because of its profound stabilizing influence on our culture, as well as the important economic benefits of strong and intact families. But today that

ideal is under an unrelenting attack, and same-sex marriage in Western Virginia is a closer reality than you may think." As he says this, a crosshair appears on the screen with the hot couple in its sights, personifying the homo who wants to get married as a family-assassinating sniper.

The shot of the couple through the rifle sight, at least to me, was reminiscent of a videogame—a sniper game like Silent Scope, for example. I decided it *should* be a game: about a gay sniper who assassinates long-held American values.

I made *Gay Sniper* in Game Maker. Here's how the game plays: The game starts with a title card. Then the player sees the ideal family from the video through a crosshair. The crosshair is smaller than the screen, and the player can move the mouse to pan the crosshair over the screen. (The player can only see what's inside the crosshair. Anything else is blacked out.) The voice-over from the video—"marriage between a man and a woman is held up as the ideal in all of civilized society"—plays. When the player left-clicks the mouse over the family—to pull the trigger—a gunshot is heard, and the game changes to a screen that displays the text "America is destroyed" over a picture of the contiguous United States cracking in half. After a couple of seconds, the game ends.

It's a simple game, much like *You're Invited to Go to Heaven*. How exactly did I make it?

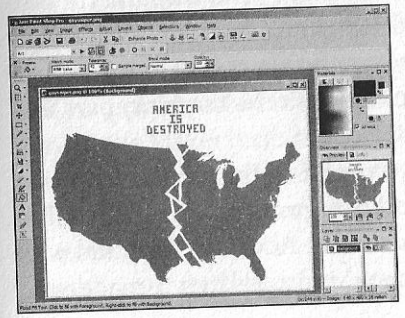


I'm going to avoid going into too many details in explaining how I made the game work. This isn't intended to be a guide to working with Game Maker: I'm interested in writing a more

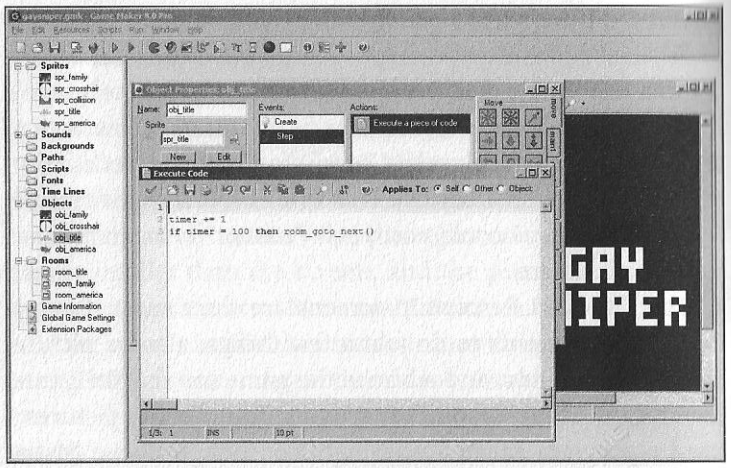
universal approach to thinking about game design and in making the decisions that guide an idea into a game. If you want more specific details on how to build a game in Game Maker you ought to be able to find them easily, by asking questions of other Game Maker users or by consulting a tutorial on the Internet.[64]

First, let's consider the shape of the game. It's made of three distinct screens: a Title screen that introduces the game's theme, the Crosshair screen where the player takes action, and a screen that displays the Resolution of the player's action (the destruction of America). This is the basic narrative shape. In Game Maker, I could easily use "rooms" to arrange these individual screens.

The Title and Resolution screens are the easiest. Each of these screens needs to do just a few things: show a picture, wait a few seconds, and advance the game (or end the game, in the case of the Resolution screen). To draw the pictures I used a program called PaintShop Pro, though Game Maker has a built-in drawing program that I could've just as easily used. A free drawing program like GIMP could have done the same thing. Here's a list of the pictures I drew: one that says "Gay Sniper" in red text on white, and one that says "America is destroyed" over a picture of a cracked United States. For the latter, I downloaded a map from the Internet, dyed it red, and drew a big crack along the middle. Then I used Game Maker's built-in timer system, called an "alarm clock," to make those images stay on the screen for a few seconds



so the player can read them before going to the next "room." (Though making a manual timer is just as easy as adding one to a counter on each frame, and then doing something when the counter reaches the right number.)



That's all these two screens do. The Title screen also plays the same little melody that opens the West Virginia 4 Marriage video (a light twinkly tune that helps us prepare to appreciate heterosexual marriage). I recorded that directly from the video using a free program called Audacity[65] and imported it into Game Maker. The Title and Resolution screens are easy, as there's no interaction. They just frame the action that happens in the Crosshair screen. Interaction is the tricky part, and also the most important part.

The Crosshair screen requires a few things: the narration from the video and the sound of a gunshot, for starters. The former I recorded off the video with Audacity, like I did with the melody on the Title screen, and the latter I made with

a free sound effects program called SFXR.[66] Next, the family in the background. I took two screenshots of the family, one in which they're about to blow bubbles and one in which bubbles are blown, and I used the timer to make the image switch every few seconds to give them a little bit of life. And then there's the crosshair itself: the interaction. There are two parts to the interaction: the player moves the crosshair by moving the mouse, and the player fires the gun by clicking. For the first part, I just needed a crosshair that moved to wherever the player's mouse cursor might be. I drew a round crosshair in Game Maker's drawing program and made the inside of the circle transparent and the outside black. Now I needed to have it follow the cursor, so I used Game Maker's scripting language to check the mouse's position at every "frame" of the game, then set the crosshair's position to mouse_x, mouse_y (x being the horizontal position of the mouse on the screen, y being the vertical).

But I only wanted what's inside the crosshair to be visible. So every frame, after I moved the crosshair, I drew four big black boxes, large enough to cover the screen: one to the top of the crosshair, one to the left, one to the right, and one below. Since we've told the game to check the crosshair's location at every tick of the game's timer, we can easily use that information to plan out where to put the boxes on the screen. An even easier solution would have been to just make the crosshair image twice the size of the screen, with black all around.

The second part of the interaction is left-clicking to fire the gun. Game Maker includes predefined "events" to check player input, such as pressing a key on the keyboard or clicking the mouse. When the player left-clicks, the simple solution would be to tell the "left-click" event to stop the narration, play the gunshot sound effect, and advance the game to the Resolution

screen. I wanted something a little more ambitious, though: I wanted the player to have to shoot the family. Clicking away from them should fire the gun, but not destroy America.

What I did was take one of the pictures I had made of the family and trace over it in two colors, one of which I set to transparent, indicating where the gun would miss, and the other, which outlined the family, indicating where the gun would hit. (This is called a "collision mask" in Game Maker.) When the player left-clicks, the game checks where the hidden outline of the family is on the screen, where the crosshair is, and if the two are touching. If they are, the "event" stops the narration, plays the gunshot, and goes on to the Resolution screen. Otherwise, it just plays the gunshot.[67]

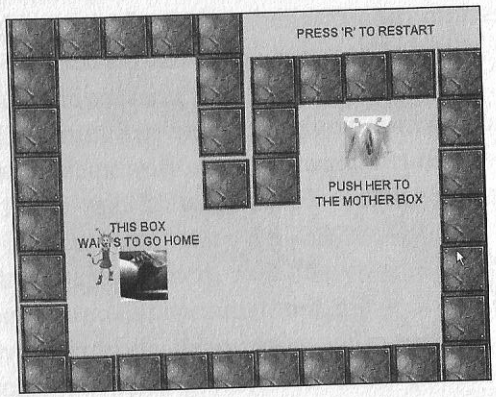
So that's the game: we have a screen that establishes the scenario ("Gay Sniper"), one in which the player takes action (moving the crosshair and firing the gun, inevitably shooting the idealized hetero family), and a final screen that resolves that action (with the inevitable destruction of America). What makes it a game is the Crosshair screen. It's the player who does what the video implies; she fills the role of Gay Sniper and fires the gun. This is why I want the player to miss if her crosshair isn't over the family: I want the player to have to consciously aim and shoot, to make her more complicit in the fantasy that the West Virginia 4 Marriage video suggests, and that I am forcing the player to act out. By forcing the player to inhabit a political ideology—because that's what games, uniquely, can do—I am pointing out how absurd it is. It's not a complicated game or a long one, but one I thought was worthy of bringing into the world.

Imperfection: *Box Pushing Game*

In November of 2009 I made a game called *Box Pushing Game*

for Klik of the Month. That was a particularly exciting Klik of the Month because some friends of mine were in town and we all got together in the same room to make games, and to play and give criticism of each others' games.

Box Pushing Game was intended as a parody of games like *Sokoban* where the player's goal is to push and arrange a number of crates in a constrained area. (*Sokoban* translates from Japanese as "warehouse manager.") In *Sokoban* the boxes move along a grid, and the conflict is that pushing the crates around will put them in the way of other crates, forcing the player to plan and manage them carefully.



In my game, the boxes aren't crates but vulvas. (*Box* is old slang for "vagina" and I'm not sure anyone knows why.) It was an idea I thought up in less than a minute during a conversation, and it was the perfect level of dumb for Klik of the Month.

So the ideal version of my game would be one in which the player pushes pictures of vulvas (downloaded from the amateur porn site *I Shot Myself*) neatly along a grid, managing them carefully to ensure that they all reach their destinations.

But in the interest of getting the game done in the two hours of Klik of the Month, I was using Klik & Play to make it. And that meant making a lot of compromises. But I could convey the concept of my game—*Sokoban* with hoo-hoos—without all the particulars having to be exact.

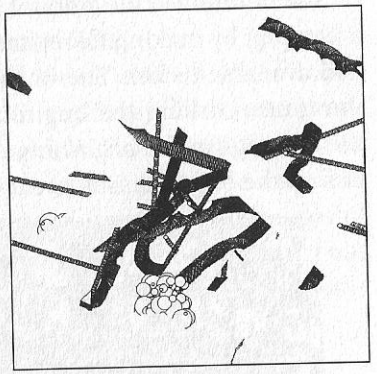
Instead of pushing the boxes along a grid, whenever the player touches a box it just shunts a bit in the direction the player was moving. Instead of being stopped by walls, it just bounces in the opposite direction with a CLANG when it touches one. Instead of having to fill a number of marked spaces with boxes, each level includes "Mother Boxes" (bright pink vulvas with eyestalks), each of which vanishes when a box comes into contact with it, taking the box with it. When all of the Mother Boxes are gone, the player goes on to the next level. And instead of checking whether one box is blocking another—which would take time to figure out how to do properly—when two boxes overlap, they resolve the collision by wiggling in random directions until they aren't touching anymore.

It's a really clumsy solution to the problem of keeping things from occupying the same space, but it doesn't distract from the idea of the game: push a box with a vulva on it from point A to point B. If anything, it gives the game more character and identity: another Klik of the Month participant described the rule as, "When my box touches another box it goes crazy."

Accidents are creative. At a more recent Klik of the Month that we held on the night of a thunderstorm, my friend Loren Schmidt (hereafter called "Sparky") was working on a quick implementation of Atari's old *Asteroids* game. (The player maneuvers to destroy asteroids by rotating a ship left or right and firing an engine for thrust.) When Sparky called me over

to look at it, he hadn't yet implemented a screen refresh—that is, everything that was drawn to the screen stayed there instead of going away. So the player's ship, when it moved across the screen, left a long snake-like trail behind it. Sparky really liked the effect and wanted to keep it.

I told him that he should make the asteroids white, the same color as the background, so that they'd be hidden until they passed over the trails that the player (or her ship's bullets) left on the screen, giving her an incentive to move around and cover as much as the screen as possible (while using bullets to gauge the presence of asteroids in her path). The "accident" of not implementing a screen refresh led to the author finding an entirely unique identity for the game.



Perfection isn't a useful goal; if anything, it keeps amateurs from getting their feet wet and authors from finishing their works. It's an ideal that hinders more than it serves. Imperfections, creative accidents, and compromises, on the other hand, give a game personality and individuality. Tolstoy's *Anna Karenina* begins with the line: "Happy families are all alike; every unhappy family is unhappy in its own way."[68] Difference is valuable, and creative accidents and jury-rigs help us achieve it.

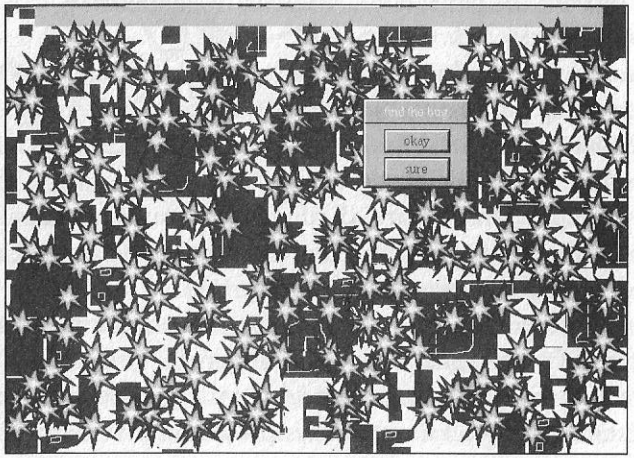
Imperfection is an invaluable tool when making games, particularly when making your first games. Think about the ways you can approximate an idea with the tools you have and the things you know you're capable of. Don't worry about

minutiae: you can express the idea of your game to a player without everything being exactly as you've envisioned it.

Use What's on Hand: *Find Shit*

My submissive's first videogame was a Klik of the Month creation called *FIND SHIT*. She had no prior experience with coding or with game making other than looking over my shoulder. How did she make her game? She was using Klik & Play, so she had access to all of the clip art that comes with it. She put a tiny dancing red bug on the screen. Then she covered the rest of the screen with huge red starbursts. The player's goal is to find the bug hidden in the loud, flashing, abrasive screen before the timer runs out.[69]

She minimized the amount of scripting she had to do (what a lazy pig) by making clever creative decisions about what she had available to her. The only things she had to script were the timer, clicking the bug to advance to the next level, and the message at the beginning of the level that tells the player to find the bug and asks her to click on a button to start. The

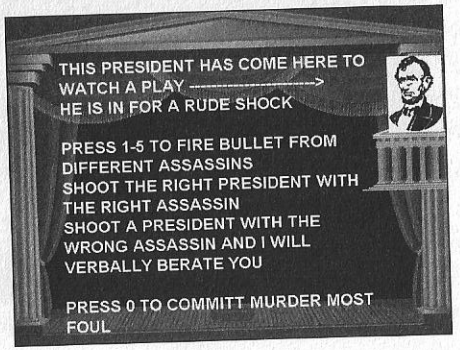


buttons are a kind of object available in Klik & Play to allow the author to ask yes/no-type questions of the player. After putting the premade button objects in her game, though, my pig discovered that they're broken: every choice calls the same response. So she made every choice an increasingly apathetic variation on "YES." (For example, the first screen asks the player to "FIND THE BUG," and lets her respond with "okay" or "sure.")

Klik & Play supports WAV audio files, so she put a bunch in. Playing three of them on top of each other, she found, makes Klik & Play totally freak out and begin playing glitchy machine noises. The shrill nightmare sounds perfectly fit the tone of the game, which is already visually abrasive.

Limitations and Creativity: *Shoot! Win!*

In February of 2010, a friend's fiancé made a game called *Shoot! Win!* This game is a shooting gallery of American presidents: images of Abraham Lincoln, William McKinley, James Garfield (represented by a picture of Garfield the cat), and other presidents bounce around the screen, while images of their assassins (John Wilkes Booth is a phone booth) pose along the bottom. The player presses keys to make assassins



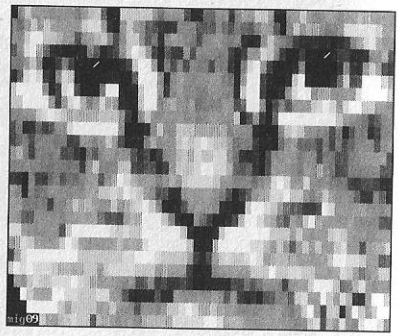
fire at the bouncing presidents, trying to shoot a president using the historically accurate assassin.

When a president is shot, a voiceover explains why that assassin did not historically assassinate that president and chides you for your incorrect choice. ("John Wilkes Booth did NOT shoot James Garfield.") There's a unique voiceover for every combination of victim and assassin (with five assassins and five victims, that's twenty-five vocal samples). These explanations make up the brunt of the game's content; they're what make the game interesting and entertaining to play.

All it took to create the vocal samples was a microphone, the historical background, and the willingness to record them. Again, the author keeps scripting to a minimum by making the focus of the game something he can easily create. And because the game is built around components that the author had the means to create, it's again unique, different from anything that an author who had the time, resources, and ability to perfectly mimic any existing game might have produced. Limitations, both self-imposed and otherwise, guide our creativity.

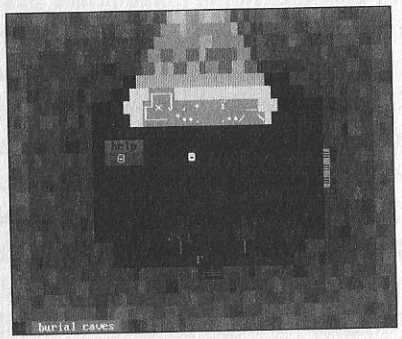
Working Within Limitations: "Flimsytown" and "Pestilence"

Tim Sweeney's *ZZT*, created in 1991, is a game-making tool that uses ANSI, the sixteen-color text mode that came preinstalled on PCs of the time, to display all its graphics. This limits *ZZT* authors to a choice of 256 characters—half of which are the letters of the alphabet



written in lower and upper cases and using different combinations of diacritical marks—rendered in sixteen possible colors. (Each character can contain two colors, a foreground and a background color, the former for the character itself and the latter for the negative space that surrounds it. Eight of the sixteen colors can only be chosen as foreground colors.) This may seem constrictive, and it is, but authors took the limitations imposed by the program as a creative challenge.

How do you animate a spinning turnstile, for example, in sixteen colors and 256 mostly alphabetical characters? You might have a character that alternates between an X and a cross. A twinkling star could be a dot that blinks as a +. A club (the playing card symbol) could be colored green and made to stand for a bush or tree. The symbol for pi could be a table. Three different "pattern" characters from the extended ASCII character set are similar to different weights of crosshatching, and when combined with a creative mind and an artful use of color can texture a scene impressively.



In fact, the scenes are more impressive for being drawn within the limitations of 256 characters and sixteen colors. Working within those limitations teaches the author to be

clever and creative with what she's given in order to make an impression. The limitations of *ZZT* also change the terms of the challenge. An author isn't trying to produce something visually beautiful in comparison to the entire body of contemporary videogames, those produced by small armies of artists and those consumed by millions of mainstream gamers, but rather to produce something visually effective within a small set of limitations and for a much smaller body of peers. This narrowing of the set of possibilities helps the author stay focused and avoid the pitfalls of over-ambition.

The same applies to scripting. Creating a variant of a game like *Lemmings* might not be technically impressive in most programming environments because it's already been done many times, but in ZZT-OOP—*ZZT*'s scripting language, which is far less powerful than most programming languages—it's a somewhat astonishing feat.[70] (That still doesn't mean creating a *ZZT* version of *Lemmings* is necessarily valuable; to do so would be an accomplishment on the order of creating an intricate copy of the statue *Rape of the Sabine Women* in LEGOs. Limitations force you to be inventive, but you can also fall into the trap of pursuing arbitrary technical goals rather than growing as an author and storyteller.) When a small and isolated community consolidates around a game-making tool, creators start to create more and more for each other (because who else is playing?). Their target audience becomes other creators who possess a detailed mechanical understanding of the game they're working with. As a result of authors creating for one another (and challenging one another), their creations become harder and require players to have more mechanical knowledge in order to play a game at all, until most experiences are too hard for most players.

This is true of a lot of levels made for *Knytt Stories* (Nicklas

"Nifflas" Nygren's expandable game about running, jumping, and climbing.) Many of the levels that circulate on Nifflas's online forums[71] are about really tricky, difficult maneuvering. So much of the technical ground of what is possible with the tool has already been explored and so many creators who work with *Knytt Stories* are familiar with the basic and advanced mechanics of the game that creators are naturally drawn toward designing for this group and taking as their starting point something far beyond what most novices using the tool would be capable of easily appreciating.

But this tendency to design for a select group can also push authors to further unexplored conceptual ground. Recently, I played a *Knytt Stories* level called "Pestilence."[72] In this story, the author has taken the terrain tiles that come with the tool and arranged them seemingly at random to create a broken-looking world. Power-ups that the player would normally expect to collect are locked inside walls, out of reach. The characters who populate the shattered landscape bemoan the hopelessness of their situation as though they were in glitchy videogame hell. (They are, in fact, trapped—the editor for *Knytt Stories* doesn't provide characters with any means of moving around the world or leaving the place in which the author set them.) Though the world seems to have little order, a player who explores will eventually be able to find one of several ways out.

This is the sort of experiment with form that rarely happens in a less restrictive format. With the exception of its music, "Pestilence" is made entirely from parts included with *Knytt Stories*; it's the way in which those parts are arranged that is unique. It's easy to arrange tiles in an unconventional order, but to do so with intention, to build a functioning, solvable world out of them? This is the sort of conceptual ground that, I believe, is usually only reached when possibility-space

is finite, and when a lot of that space has already been thoroughly mapped. Think of Los Angeles versus New York City: one city, with a great deal of continental terrain, expands outward; the other, built on an island, expands upward.

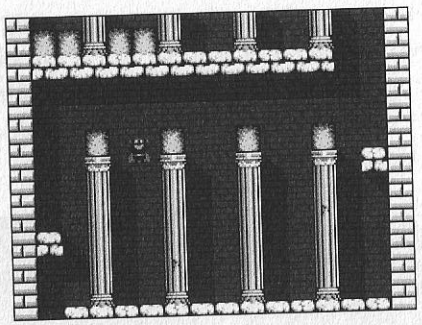
ZZT, unsurprisingly, has an equivalent level to "Pestilence." It's usually called "Flimsytown." "Town of ZZT" is the official level packaged with the *ZZT* editor and program. "Flimsytown" starts identically to "Town of ZZT," but quickly diverges, presenting a jarring and confusing experience that requires the player to actually exploit the bugs and glitches inadvertently built into *ZZT* in order to progress. For a while, supposedly, the website *Z2*, the most well-known online archive of *ZZT* levels,[73] included "Flimsytown" in its *ZZT* package in place of "Town of ZZT."

Be Derivative: *Mighty Jill Off*

> Hi, I'm Der Rivative. Come to see my work? I've got this medical paraphernalia here, which I've made into little objects. Butt-plug stool—some people say it's like Nayland Blake. No, it's mine, Mr. Der Rivative. I made it. Here you have scaffolded video representing my face in six monitors. Some people say Nam June Paik, some people say Bruce Nauman. No, I made it—Der Rivative. I like to lie in bed, as long as I can, let people in the gallery come and see me. Nothing like the Chris Burden on the shelf thing, it's a Der Rivative extravaganza. I invented it. I made it. People come to the museum and say, "Hey, there's a Der Rivative."[74]
>
> —Bob Flanagan

Nothing is original. You've probably heard the old quote about there only being seven basic stories (or three, or fifteen; the number always changes) a thousand times. Quantification aside, the purpose of the quote is to make the listener realize that what's interesting about the story is not the story itself but the way it's told. There are a lot of novel ways to tell stories with games. We've found very few of them so far.

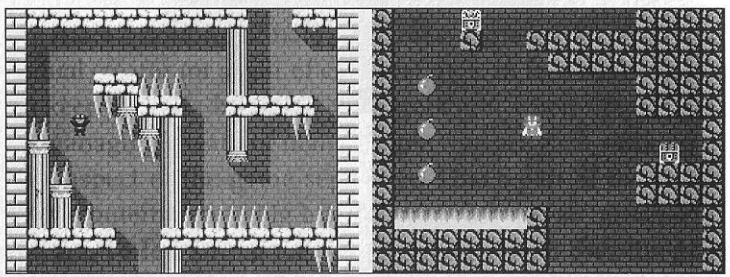
We're going to find many of them, probably, by accident. Don't worry about making an original game—just worry about making it yours. Many authors start by trying to replicate games they've already seen: *Super Mario Bros.*, *Mega Man*, and *Final Fantasy* are popular. And many of those games end up being very different from their antecedents because the authors didn't have the technical or artistic skill to mimic those antecedents exactly, and instead had either to approximate the parts of the game they couldn't replicate or to improvise something entirely new. Or to incorporate a single new idea that changed something significant about the basic model. Or simply to invest the game with their own personality.



One of my most successful games (in terms of press, not money; the game was released for free on the Internet) is *Mighty Jill Off*. It's based on a Nintendo game called *Mighty Bomb Jack*, published by Tecmo in 1986. *Mighty Bomb Jack*'s

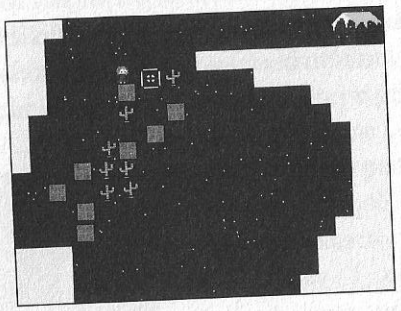
predecessor game, *Bomb Jack,* predated *Super Mario Bros.* and presents a very different vision of the way a videogame about jumping can work. While the emphasis in *Mario* is on the planning of a jump—the momentum Mario gains on the runway before the jump, the limited English he can spin on his jump while he's actually in the air—the character Bomb Jack is highly maneuverable while in the air. He can jump the entire height of the screen, reverse a jump on a dime, and flap his cape to keep himself from descending too quickly.

I made *Mighty Jill Off* during the first semester I spent at the Guildhall because I was bored to shit with their lectures and I needed an actual design task to challenge me. For *Jill Off,* I lifted the entire vocabulary of *Bomb Jack*—the high jumps, the reversals, the cape flap—and put it in the service of a story about a submissive masochist climbing a tower to prove herself to her Queen. As a pervert who missed her own submissive (who waited for me in California while I toiled at school in Texas), it was a subject that was both on my mind and relevant to the way I experience games: games about challenge are about the relationship between the player and the designer, the former trying to prove her capacity and will to the latter, the latter trying to continuously challenge the former while maintaining her trust.



Borrow whatever you can. If you were to try to complete a jigsaw puzzle with pieces collected from five different puzzle sets, there'd be blanks that the pieces you've borrowed wouldn't fill in. You'll fill in the blanks with your own personality. You'll smudge the corners in a way that is identifiably yours. And you'll create something new: maybe by design, probably—and preferably—by accident. The modern novel started when Cervantes took the stock idea of "a knight who has adventures" and put him into his own contemporary Spain. Modern music started when bluesmen took traditional folk songs and began to bend the notes. Modern sculpture began when Duchamp took a space in the gallery reserved for artwork and put a urinal in it. Setting out to make something utterly new can be a trap to a new author: you'll spend forever planning and no time doing. If you get busy without worrying about being original, you're liable to stumble across many more interesting ideas.

Make Weird Shit: *Cactus Block*



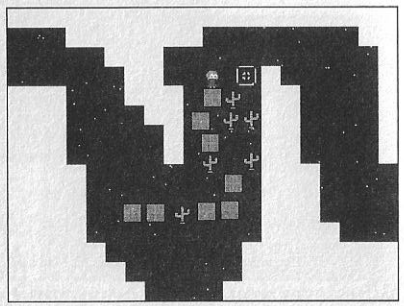
Put weird shit into your game. Make unusual creative decisions. You're not beholden to anyone, so why be conventional? Whenever you see an opportunity to make something a little

more distinctive, take it. These are the parts of the game, after all, that are most uniquely you.

Glorious Trainwrecks user chuchino made a game called *Cactus Block*[75] in February 2010. This is a simple game about a huddled figure trudging through the snow from the left side of the screen to her house on the right side of the screen. She can jump, but her house is located atop ledges and cliffs that are far too high to jump to the top of. Fortunately, the player has the power to create additional platforms—blocks—by moving the mouse and clicking on areas of open space. The goal, then, is to place the blocks appropriately in order to climb over the obstacles.

This is the point at which the author made a Weird Decision. About half the time, when the player clicks the mouse, a deadly cactus will appear instead of a block that's safe to stand on. (If the player touches the cactus, she must start the screen over.) That's a pretty silly idea, but this is the rule that defines the game.

Because of the unpredictable way that this introduces chance to the player's actions, the player has to plan carefully to accommodate for potentialities. Maybe I click in a less optimal platform location because there's a fifty-fifty chance of my click producing a cactus, and I don't want to block that prime location. Do I take the risk of trying to move forward quickly, depending on a lucky streak of blocks, or do I work slowly, making sure I always have a back-up plan? Do I try the last resort of jumping into



the air and clicking immediately under the protagonist, with an equal chance of producing solid ground or instant death? These are all interesting, unique player choices, and they spring from a single unorthodox creator choice.

As a Klik of the Month game, *Cactus Block* was made in two hours. It's not the same case as a big-budget commercial game, where lots of money and hours have been invested in a project, and where a publisher expects a return on its investment. If you do something wacky in a two-hour game and it doesn't quite work out, you've gambled and lost just two hours of your time. And I use the word *lost* reluctantly, because I don't consider any game a total loss, even if the results don't work very well. A "loss" here means that you've made something unique and that you have that much more experience putting a game together.

Beyond the potential for interesting design generated by making something weird, any opportunity to do something weird is also an opportunity to put some of yourself in a game. Most of my games star perverts, even if all the perverts are doing is jumping up a tower. Some hobbyists try to emulate what they think of as "professional" game development and divorce all aspects of themselves from their games. That's ridiculous! Your game comes from you; your game should contain you. A game you make should have your signature on it. It should reflect something of your personality, your humor, your values. After all, you're only beholden to yourself. Make something that's your own.

What to Make a Game About?

Your dog, your cat, your child, your boyfriend, your girlfriend, your mother, your father, your grandmother, your friends, your imaginary friends, your summer vacation, your winter

in the mountains, your childhood home, your current home, your future home, your first job, your worst job, the job you wish you had.

Your first date, your first kiss, your first fuck, your first true love, your second true love, your relationship, your kinks, your deepest secrets, your fantasies, your guilty pleasures, your guiltless pleasures, your break-up, your make-up, your undying love, your dying love.

Your hopes, your dreams, your fears, your secrets, the dream you had last night, the thing you were afraid of when you were little, the thing you're afraid of now, the secret you think will come back and bite you, the secret you were planning to take to your grave, your hope for a better world, your hope for a better you, your hope for a better day.

The passage of time, the passage of memory, the experience of forgetting, the experience of remembering, the experience of meeting a close friend from long ago on the street and not recognizing her face, the experience of meeting a close friend from long ago and not being recognized, the experience of aging, the experience of becoming more dependent on the people who love you, the experience of becoming less dependent on the people you hate.

The experience of opening a business, the experience of opening the garage, the experience of opening your heart, the experience of opening someone else's heart via risky surgery, the experience of opening the window, the experience of opening for a famous band at a concert when nobody in the audience knows who you are, the experience of opening your mind, the experience of taking drugs, the experience of your worst trip, the experience of meditation, the experience of learning a language, the experience of writing a book.

A silent moment at a pond, a noisy moment in the heart

of a city, a moment that caught you unprepared, a moment you spent a long time preparing for, a moment of revelation, a moment of realization, a moment when you realized the universe was not out to get you, a moment when you realized the universe was out to get you, a moment when you were totally unaware of what was going on, a moment of action, a moment of inaction, a moment of regret, a moment of victory, a slow moment, a long moment, a moment you spent in the branches of a tree.

The cruelty of children, the brashness of youth, the wisdom of age, the stupidity of age, a fairy tale you heard as a child, a fairy tale you heard as an adult, the lifestyle of an imaginary creature, the lifestyle of yourself, the subtle ways in which we admit authority into our lives, the subtle ways in which we overcome authority, the subtle ways in which we become a little stronger or a little weaker each day.

A trip on a boat, a trip on a plane, a trip down a vanishing path through a forest, waking up in a darkened room, waking up in a friend's room and not knowing how you got there, waking up in a friend's bed and not knowing how you got there, waking up after twenty years of sleep, a sunset, a sunrise, a lingering smile, a heartfelt greeting, a bittersweet goodbye.

Your past lives, your future lives, lies that you've told, lies you plan to tell, lies, truths, grim visions, prophecy, wishes, wants, loves, hates, premonitions, warnings, fables, adages, myths, legends, stories, diary entries.

Jumping over a pit, jumping into a pool, jumping into the sky and never coming down.

Anything. Everything.

Get Your Hands Dirty

So get to it. Make a game. Your first game will be rough and

derivative. It may not change the world, but it will be yours. The next chapter will be an attempt to walk you through your first game step by step. It's intended to be as abstract as possible so that you can apply it to whichever game-making tool suits your needs best. You can find out the details of making that tool do what you want through reading the documentation, asking questions of the community surrounding that tool, or best of all, fiddling with the tool to see how it works.

As for what tool to choose: Appendix A lists a bunch of different game-making tools that are available, describing what they're good for and what the experience of working with them is like. It's by no means a complete overview of all the game-making software out there—there are far more tools than anyone could document—but I've included those that I think are the most useful and approachable. There will also be tools that are created after this book is published. Nothing that I'd consider the ideal game-making tool exists yet, but I'm confident we'll keep getting closer.

It's possible that your interest in digital game creation is purely academic and doesn't extend to becoming an author. In that case, I hope that what you take away from this book is that the videogame isn't the creation of a corporation, but of an author, that this form is important, and that people are using it to do exciting things.

In 2008, Nicklas Nygren (a.k.a. Nifflas, author of *Knytt Stories*) attended a festival called EDGE in Umeå, Sweden, with a laptop, a printer, and a digital camera. He asked other festivalgoers to contribute to a game he was working on. They drew pictures, photographed themselves, and sang into a microphone. Over the course of the day, he created a game called *Det Officiella Edge-Dataspelet*, or *The Official Edge Videogame*.[76] Every image and sound in the game was created by someone

Nifflas had pulled from the crowd: the protagonist is a photo of someone's head in profile, the stages are drawn in crayon, the music and sound effects were made by strangers' mouths.

This is what game creation is becoming: small, personal, and made by people's own hands and mouths instead of by corporations with teams, managers, and software renderers. What we call a videogame is not a product. It's the creation of an author and her accomplice, the player; it is handmade by the former and personally distributed to the latter. The videogame is a zine.