

Independent Game Developers

Game Makers Data Magazine Special Article



<http://www.morphenteractive.com>

Who this is written for

The title indicates this is for independent game developers, which can mean a lot of things. My intention is that this is for anyone who has some interest in computer graphics or programming, who wants to make a computer game but has never done so before, and who doesn't have a lot of money. In fact, everything discussed in this series is either free or relatively inexpensive.

My hope is that this will help promote independent game development to keep the industry fresh with interesting new titles, while empowering individuals to work hard at something and fully reap the rewards. Whether your intention is to find an entry point into commercial development or a foundation for your own independent projects, I'll try to at least point you in the right direction.

Remember, this is meant to get you started. There are limitless avenues to travel in game development, and you may find yourself starting in one place and ending up somewhere completely different. This article doesn't discuss distribution or business models for selling your product.

Expectations

Be prepared for the fact that your first game may stink. But don't feel bad. The point is you're doing it, and you shouldn't feel bad if the results don't match those of a large team with many years of experience. If you can't handle making a bad game, you're not mentally prepared for the challenges involved in development. You aren't trying to do poorly, you're trying your hardest to succeed, and through that process, you learn why your creation failed and how to make it better. It's like anything else that takes skill and experience. The truth is, you'll never fully arrive at the nirvana of game development. There are always things you can do to make your next effort better, and if you have the right attitude, it can be an enjoyable process of learning and growing.

Roles

It might help to begin with a brief discussion of the various roles in developing a computer game. This doesn't mean you have to pick just one. I'll be the first to admit I love to play as many of these roles as possible. Some people like to say, "Jack of all trades, master of none," which has some truth to it if your goal in life is to be a master of one thing, but I think mostly it's said by people who aren't good at very many things and need to sound pretentious about their specialty. I believe everything you learn has merit and contributes to the store of knowledge you'll draw on for the rest of your life. I've spent my working career going back and forth between art and programming, and I'm constantly finding ways to integrate the two, breaking those stereotypes that say you're either an artist or programmer, not both. Since, as an independent developer, you don't have vast resources to hire all those singular masters,

you may find yourself doing everything from writing code to designing graphics to composing music.

If I had to break the process down into three general tasks, it would be design, content production and technical implementation. Design involves broad decisions on how the game is played and a roadmap for production and implementation. Content production involves the creation of assets like artwork and sound effects. Technical implementation is the process of writing code or using tools to assemble assets into a playable framework that follows the rules. All three areas are interrelated, and frequently, the development process requires some of these tasks to take place in consecutive order, while others can happen concurrently. The remainder of this series will focus on these three areas and how to get started in each. The series isn't split based on these three areas; I felt it would be more appropriate to discuss aspects of content production and implementation together in the same article.

Game Design

The design phase is the decision making process that answers the question, "What is the game and how do you play?" I can't underestimate its importance. You may be itching to create your logo and code in the splash screen, and by all means, get some of that out of your system up front. Do all of that emotional pride stuff for a few hours, then when you're serious about getting started on the actual game, get some paper and a pencil, and begin the brainstorming process.

Generally speaking, you have a few broad directions to take on design. First, you could implement an existing set of rules - something like chess or football. There are still design decisions to make, but for the most part, you know what the game is and how it's played. Any decisions you make are for the sake of style, enhancement or modification. Second, you could use an existing pattern or genre: first-person shooter, real-time strategy, RPG, simulation, arcade and so forth. These provide a starting point from which you'll customize your venture.

Patterns are a vocabulary for describing your plans. Instead of saying, "This is a game where players run around in a 3D virtual environment picking up weapons and attempting to shoot other players," you can just say, "It's a first-person shooter." Using patterns means some of the design decisions are already made for you because they've been proven to work. Obviously, you'll want to have your own unique angle that will be the motivating factor for someone to want to play your title, but a pattern gets you started with a general structure for design.

Most difficult, but possibly very rewarding, is the invention of a new pattern. You make up all the rules and structure for your game, and it can't be classified like any other. It's somewhat unlikely that it couldn't be classified at all, but you may find yourself saying something like, "It's an arcade game, but it's totally different from anything out there." It's very hard to come up with something that has never been done before - at least something that's fun to play. In fact, it might not be a good idea to abandon all existing patterns in favor of your own unique concoction. There's a reason why so many games implement concepts like "scoring points" or "inventory." Players learn to recognize these fundamental building blocks and can quickly get up to speed. If you invent something so unusual no one can figure out how to play without reading your hefty instruction manual, you're going to face an uphill battle to success. On the other hand, if you do invent something completely original and people love it, you'll be really cool and maybe rich.

Of course, all of the above starting points involve some form of synthesis. You're putting together rules for play. You can pick and choose from what has already been done and what you invent yourself, and in the end, you'll have some form of blueprint for how the game will be played and the elements that will be involved.

Once you have the basic principles on paper, you can start fleshing out the details. What are the specific characters, weapons and levels? If you can cast spells, what are they, how much mana does each one cost, and how much damage does it do? What monsters does the player need to kill, and which weapons are effective against which armor types? All of these questions need to be answered at some point, and it's good if you can think this through ahead of time, rather than waiting until you've spent a week modeling and coding before you realize you didn't need that part, or you have too many of these or too few of those. Sometimes, however, you may not know exactly what you need until you have a playable prototype.

I can't tell you how to design your game - that's your job - but I do want to help get you started. With the above branches, you can at least decide whether you want to create a title based on established rules or one that fits into an existing genre, or one that's completely original. The next section will mention some general elements of design that you might want to consider.

Game design elements

There are certainly more design elements than I describe below, but this is at least a way for you to start thinking in terms of your role as the "game master." You're the one pulling the strings behind the curtain. It's your responsibility to ensure players enjoy themselves, and it's helpful to think about the various tricks you have in your magic bag to accomplish this.

I break these elements down into two areas - the directions of communication between the software and the end user. The first area includes things the game does to influence the player; the second includes things the player does to influence the game.

Game Activity:

Competition - A game promotes competition for the player. This might be competition against another player or against the computer. Assuming the activity offers room for developing skills, competition can be a very powerful motivation, and can keep people playing the same title for years.

Exploration - A game provides interesting stuff for the player to discover. It keeps the player involved by offering different places to explore. This may include more than just locations. It could involve the discovery of characters, objects, special abilities and so forth.

Rewards - A game stimulates the player's emotions by offering periodic rewards. A new weapon, bonus items, level or mission, extra skill points, another clue to the mystery and so on. Rewards are a very powerful motivation if used properly. They can keep a player engaged for hours, continually trying for that next prize.

Entertainment - A game attempts to entertain the player in some way, often through cutscenes, back-story or level objectives that follow a plot. Entertaining doesn't necessarily mean funny. A serious title with a disturbing storyline still seeks to entertain by providing a well-plotted narrative. While many game theorists search for a way to implement the elusive non-linear story, most stick to the tried-and-true methods of unfolding a linear one. The success of entertainment is a direct correlation to the quality of content production. Because of its passive nature, my personal opinion is that this is a weak design element, though it's quite successful in offerings that mimic popular books or movies, due to the player's familiarity with the story.

Challenge - A game offers something difficult for the player to attempt to accomplish. It's almost comical to think about the ridiculous challenges a player will accept, particularly among the more creative platform games: "Collect as many gold rings as you can without touching

any spiked turtles and while sliding across icy patches and leaping over crocodile pits." Frequently, challenges come in series and are increasingly difficult as the game proceeds.

Role-playing - A game provides a foundation to allow the player to feel as though he or she were in some imaginary role. This doesn't have to be in the traditional wizard or warrior style. Any offering where the player makes a connection with a virtual persona involves an element of role-playing. For example, the player might take the part of a space station commander, a military special-ops agent, a ruthless pirate or a skateboarder.

Creativity - A game offers a customizable playground that enables the player to make something. Build a house, design a character, arrange an amusement park, lay out a military base, or paint or shape something.

Resource Management - Resource management involves some form of an economy where certain quantities of stuff can be traded for other things. For example, workers can chop down trees to obtain wood, which can be spent on the construction of a building, which can then house more workers.

Acquisition - A game provides a coveted commodity and demands the player make an attempt at acquiring it. The most common examples are points and money. Acquisition can be a subset of resource management.

Player Activity:

Remember that, as the designer, you set up the rules the player follows. Just because these are actions taken by the player doesn't mean you don't do anything. On the contrary, the designer has to build a game that allows and promotes enjoyable input.

Strategy - The player must make some intelligent decisions that can result in success or failure at a given objective. A game with good strategy will offer multiple techniques to accomplish a goal, and each technique can be nullified by others. In this way, competition is possible. Frequently, strategies emerge that are too powerful, and the rules must be adjusted to balance the offering's strategic aspects.

Reflexes - The player develops mechanical skills to recognize and respond quickly to events using the input device. Arcade games and 3D shooters frequently depend on reflexes.

Analysis - The player obtains information about the game world and makes logical conclusions that affect his or her decisions. This is related to strategy, but I see it as more of a discovery technique. For example, the game first informs the player that she needs the blue key to pass through the blue door. The player continues until she finds the blue key, then makes the analytical decision that she can now go back and open the blue door.

Randomization - This could be considered part of game activity, rather than the player's activity, but in many cases, it's the player who performs an action, and the software simply provides the facility to determine a random outcome. Very few actions are completely arbitrary. Usually a player takes some form of action in the strategic or reflex category, and the random factor modifies their success. For example, a player performs a strategic attack, and the game dictates there's a 40 percent chance of success.

Self-determined - The player makes a personal decision about the activities to take place. This is often related to the creativity element in game activity, where success is subjective. For example, a game may allow the player to decide whether to be an evil character or a good one. In either case, it has rules for success, and the player's satisfaction comes from his or her own decision on a course of action.

Game design tips

These are generic suggestions based on my personal experiences with game design.

1. Don't forget fun for the player. Your overall objective is to provide enjoyment. Everything you do should somehow be centered on that goal. The focus of the game isn't you, it's the player. You may think something is fun, but try to put yourself in the shoes of the average player, and decide if it's really fun or if it's just something you like for personal reasons.
2. Try to keep things as simple as possible for your audience. It'll make your game simpler to develop and easier for players to grasp. This doesn't mean you avoid anything complex. Some titles are very intricate, but they're a direct correlation to the player's desires. If an RPG fan loves and wants 20 different character attributes, and you can make each one noticeably effective, include them. But if no one can tell any difference between "Strength" and "Brawn," or no one cares to learn the difference, then you're better off simplifying the design.
3. Beware the dangers of numeric commitment. Inevitably, you'll find yourself estimating numbers of things - five player classes, eight game levels, ten magic spells, six gold keys. Just be sure to leave your design flexible enough to change those numbers when you find out how much time it'll take to implement all of those things. For example, be leery of building a level with 12 hallways branching from a central zone with portals at the end of each one to other levels. After a year and only three levels done, you may wish you didn't have nine empty hallways remaining. Sometimes, you can't predict how difficult something will be, but the sooner you can accurately predict your workload, the easier it'll be to make design decisions.
4. Find a partner. You may be a lone game designer, and that's fine if you are. I happen to work with another developer. If either of us comes up with a dumb or great idea, it's helpful to have the other person offer negative or positive feedback. Remember that this person has opinions just like you do, and you may not always agree. The process of discussing and agreeing on design elements is probably the single most valuable activity that can take place during your design stage. If you work alone, this discussion takes place in your own mind, and that can be successful, but sometimes it's beneficial to have a second opinion. Note that this could be someone who's also working on the project, or it could simply be a friend who enjoys playing games.
5. Record your design using a Wiki. If you've never used a Wiki website before, you should try it. It's an interesting concept where you have HTML pages that are user-editable. Anyone can quickly click an edit button and change the content of the pages, add text or links, create new pages and so forth. You don't have to know HTML, since it uses some very simple formatting rules you can learn in about 15 minutes. There's no creating, saving and uploading of files since everything is stored dynamically in a database; it's extremely fast and easy to write, edit and view information. As your design gets deeper and you add more data, you may find it helpful to organize it this way, particularly if you have more than one person collaborating on the project. The design will inevitably change as you work, and this encourages you to make alterations as they come up, rather than leaving outdated information in your documents because it's too much trouble to modify them and exchange them with other people.

There are many implementations of Wiki. The one I use is called [OpenWiki](#); it runs using ASP, XML and one of several database formats. It does require a teeny bit of knowledge in web database applications, and you need a place to host it. But once it's set up, anyone can easily use it.

6. Play the game as soon as you possibly can. Get out your dice and some paper if you can play it that way. The next section discusses prototypes, which are invaluable in early testing. The sooner you can try the game, the sooner you'll know if the idea is good or not. In many cases, this will require you to find other people with which to play, unless the concept doesn't involve foreknowledge that would ruin the experience.

Prototyping

Building a game prototype can be very valuable in certain cases. It's like an experiment: You have an idea for something, but you aren't sure about a few things, so you experiment to find out if you were right. It takes discipline to make a prototype because the work you put into it is usually thrown away. For that reason, many people think it's a waste of time and claim they need to "make some real progress." Ouch. I hate to hear that. Prototyping is extremely useful in answering questions about your design or technical implementation.

Some games require prototyping more than others do. If you know your engine very well and understand its limits, and if you have a solid grasp of your design (maybe you're working in a well-known genre or are implementing existing rules), then you might get away without a prototype. But if you have any significant questions about how the game will play or perform, you're well advised to create a prototype.

Prototype the things that make your game unique. In other words, if you're creating a first-person shooter based on a sci-fi storyline with aliens from Pluto as enemies, there's no point in trying to create a quick and dirty 3D FPS engine to see how it plays. You already know that running and shooting are core elements of a 3D shooter, and they've been proven to be fun and successful in dozens of other releases. But maybe a unique element in your design is that the aliens have radioactive blood and the player is slowly killed if he spends too much time near dead alien bodies.

Since it might be helpful to prototype this behavior, you could use an existing FPS engine (to which you might have absolutely no licensing rights, but it doesn't matter since this is a prototype), and then write a quick mod to see how it feels to leave dead bodies around the level and reduce the player's health based on proximity to these bodies. Is it annoying that the player dies for something as mundane as standing still? Can the player get trapped by his own trail of carcasses? Is that a fun, strategic element? These are questions a prototype can answer.

Don't want to take the time to write a mod for a 3D game? A prototype could be even simpler. For the example above, you could write a very crude, top-down 2D engine where you're represented by a blue dot and the enemies are represented by red dots, and you can move your blue dot around a simple board representing the level. When you're within a certain radius of a red dot, you can press a key and "kill" the enemy. (Who cares about aiming and ammo? That doesn't need to be prototyped, remember?) Then, the red dots can turn green to represent the radioactive dead body, and your health percentage can drop over time based on your closeness to the green dots. It's ugly. It's primitive. But if it can answer a useful design question and can be done quickly, it's worth it.

Remember, the point of a prototype is that it's fast to develop and it answers some specific questions that you may have about what is or isn't fun in your design, or what is or isn't technically possible. If you have no question that your game is fun and feasible, there's no point in prototyping. Don't make a prototype just because you think you're supposed to, but do be wary of assuming your idea is fun or technically possible. It might not be much fun, or it could be impossible to implement.

Note that in some cases, you may create a prototype for the purpose of proving a concept to an investor or publisher, in which case it's really a marketing demo, even if it does demonstrate a new gameplay element. In cases like this, you want your prototype to look good so it impresses those big shots with the fat wallets. This has little to do with our discussion of game design, though, and most likely, you'll have done an ugly, personal prototype before creating the more polished, public version.

Conclusion

This first article has discussed some introductory concepts for new game developers to consider, particularly related to design. The next article will introduce some techniques and tools to get started creating a 2D offering. The final article will follow a similar pattern for 3D titles.

Perry Board