robin hunicke

Put Your Game Development Education to Work

How to Choose Coursework for Your Game Career

"I want to be a game developer, what classes should I take?"

If you've taken the time to look for the answers to this question, you know that advice about game education is plentiful. In books, on the Gamasutra.com web site, at the International Game Developers Association's (IGDA's) "Breaking In" site (see For More Information), and posted in all manner of discussion forums — everyone has an opinion.

"Get a liberal arts degree."

"Study programming."

"Learn Maya!"

"Be sure to play lots of games."

"Read books."

"Make your own game!"

If my three-year tour with the IGDA Education Committee has taught me anything, it's that there is no simple answer. The real question is, which study plan is right for you?

In this article, I'll be using the committee's Game Studies Curriculum Framework to outline possible trajectories through traditional university coursework, to show you how they relate to specific careers in gaming. With these ideas, a little planning, and some introspection, you can design a program that's both successfuland tailored to your individual interests.

The Basics: Career Paths in Gaming

Last year, in "Getting into the Game: Skills and Careers in Game Development" (*Game Developer's 2002 Game Career Guide*), Ernest Adams described the various career paths within the industry and discussed what's required to break in. As he reported, the industry is new, and due to rapid growth, job descriptions vary from company to company and even from project to project. However, the basic breakdown looks something like Figure 1 on page 22.

As a developer, you will be expected to excel in your discipline and contribute to your part of the team. But you will also be expected to communicate with people outside these areas on a daily basis.

First, determine your basic skills and interests. Are you interested in the creative jobs of the development track, or would the strategic aspects of the production side suit you better? Do you consider yourself an artist or a programmer — or both?

Then, imagine yourself in your future role. Think about what motivates you, what makes you curious, and what drives you crazy. Will you be spending most of your time in front of a computer, or working with people? How will you fit into the team framework? Who will you need to communicate with? What kinds of problems will you be working on?

This kind of planning may feel a bit strange, but it's important. Otherwise, you might invest your time and money in a computer science degree only to find out later that your natural talent is sound recording, project management, or character design.

Getting Started: Programming

Programmers fill all kinds of roles within a specific game development project, from the coordination and display of assets to the construction and maintenance of proprietary tools and hardware. For this reason, the programming specialization is broken up into subcategories: graphics, physics, AI, and game and machine programming.

Any game programmer should be a well-rounded, experienced coder. Master the basic coursework available within a computer science department. Become proficient in a compiled language (such as C or C++), and know data structures, computer architecture, algorithms, operating systems, and networking. These foundation courses will help you understand the way programs and data are constructed, stored, and manipulated by the machine, all essential to game development.

Introductory Newtonian physics, linear algebra, geometry, and computational mechanics (often required for a computer science or electrical engineering major) will be essential to anyone interested in game coding. Introductory logic, basic statistics, economics, and probability will further strengthen your ability to think in terms of larger patterns and systems of behavior.

Develop your specialization skills with the appropriate electives. Aspiring physics programmers should be familiar with

ROBIN HUNICKE | *Robin is a Ph.D. candidate in the Computer Science Department at Northwestern University. A joint member of the Robotics and Interactive Entertainment groups, she studies AI and videogames.*

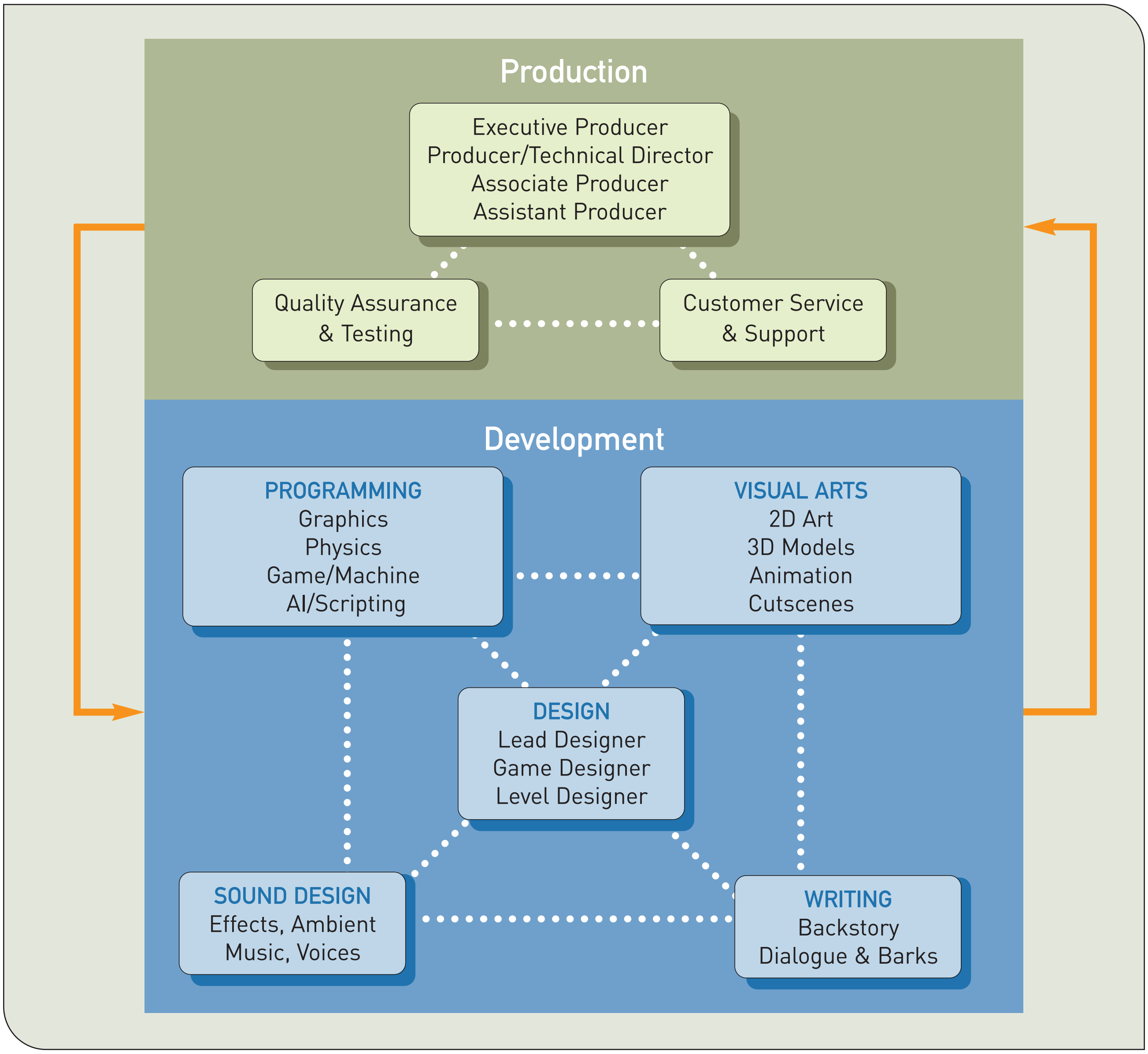

FIGURE 1. Game design is a highly interdisciplinary, collaborative effort. People within the production hierarchy work with each other to keep budgets, schedules, and feature and bug lists up-to- date. On the development side, artists and designers collaborate across disciplines to bring the game to life. Finally, the production and development teams cooperate with each other in order to keep the project on track and running smoothly.

advanced physics, math, and game physics algorithms. Graphics programmers should be experienced with current graphics hardware, rendering, shading, animation, and graphics system design. Project work with an open-source game, physics engine, or graphics library is good practice.

AI programmers should understand the basics of human and animal behavior and have experience representing these behaviors within machines. Take courses in classical AI, psychology, cognitive science, and sociology. Study film, theater, screenwriting, and creative writing to develop an eye for dramatics and believable fictional characters.

If you have an interest in machine programming or hardware hacking, take some advanced computer science or electrical engineering courses: assembly programming, compilers, computer electronics, computer engineering, programming languages, and program design. These classes will teach you how to write efficient, optimized code and build strong, extensible tools. The technical requirements for PC and console games are constantly changing. A fundamental understanding of how computers work "at the metal" will help anyone working with custom hardware or software.

Insider tip. In the real world, the need for efficient, well-designed programs that can be read by other people far outweighs the desire for clever or unique solutions. Take care to write clear and friendly code, and reuse what you can (reinventing the wheel is expensive and pointless). Use consistent

General Programming Coursework to Consider:
- Compiled language (esp. C/C++)
- Data structures, computer architecture
- Algorithms, operating systems, networking
- Newtonian physics, linear algebra, geometry
- Computational mechanics
- Introductory logic
- Statistics, economics, probability

Physics/Graphics:
- Advanced physics
- Graphics hardware, rendering, shading
- Animation and graphics system design

AI:
- Classical AI, cognitive science
- Psychology, sociology
- Film, theater, screenwriting

Tech/Hardware:
- Assembly programming
- Compilers, computer electronics
- Computer engineering
- Programming languages, program design

And Don't Forget To:
- Develop strong communication skills
- Build a portfolio of programs
- Play games

notation and comments, and pay attention to the feedback you get from others. Work on some group projects, and apply for an internship at a local game or software development company. Practice your communication skills by talking about your work with nonprogrammers. Above all else, strive to learn (and appreciate) the difference between a perfect answer and a tractable, workable solution.

And remember, interviewers will be curious to see what you've already done, so it helps to write lots of programs and create a portfolio from your best work.

Asset Production: Art

Game assets generally consist of art, animations, text, and audio components. A strong portfolio or reel that demonstrates your work is the best way to get a job in these areas. While it's important to know how to use software tools, you should focus on developing a personal style while broadening your knowledge of game aesthetics.

All game artists should be skilled at basic visual design, including coursework in drawing and painting fundamentals, sculpture, anatomy, physiology, and life drawing. Additional courses in animation, storyboarding, and concept and character design (where available) are also a plus. Classes in architecture, industrial design, film, or theater production will help you understand how settings, objects, and characters should work together to create coherent narrative space.

For 2D artists, coursework in human-computer interface design and 2D computer art generation will be helpful. Working knowledge of 2D graphics software (such as Adobe Photoshop) and experience with scanners and digital art integration techniques will also help transform a traditional art degree for use in a game production context.

Game artists working in 3D should be experienced with designing, constructing, and animating 3D models. You should learn at least one production-quality tool (such as Maya or 3DS Max). For students interested in this specialization at colleges that do not offer 3D art coursework, it is often possible to work with a local vocational school.

Asset Production: Writing and Sound

Writing for games typically involves producing high-level narrative and character treatments, developing backstory, and writing dialogue or "barks" for in-game characters. As you might expect, a background in literary theory and creative writing is of great help here. Game writers should understand basic character development, context-setting, backstory and setting design, as well as play- and screenwriting. Courses in interactive narrative, where available, will inform aspiring writers of the strengths and weaknesses of branching narratives, emergent narratives, and hypertext approaches.

Sound designers develop everything from music and ambient sounds to character speech and sound effects. Much like game programming, game audio technology is a moving target. As the state of the art changes, you will need to stay current with tools and techniques for manipulating analog and digital sounds, and know all about sampling rates, compression quality, and so on. Foundational coursework in sound design, composition, recording, mixing, and mastering is often available through technical, vocational, or community colleges. Additional courses in a music history and theory, sound effects, interactive scoring, soundtracks, and sound design for stage and film will complement this foundation.

Traditionally, game writing and game audio jobs are relatively limited, and working as an independent contractor on several projects can be more profitable than staying in one studio. So if you're looking to join the ranks here, consider a course or two in basic business.

Insider tip. Critical to all asset-production careers is an understanding of how game assets work together to create environments that engage the player. Aspiring artists, writers, and audio engineers should make a concerted effort to play games, watch other people playing games, and analyze how games evoke emotions within a player. If you can clearly communicate how your ideas relate to the fundamental aesthetics of a game, you will go much further, much faster, than those who can only make vague suggestions or complaints about "look and feel."

Game Design 101: Toward a Formal Theory

What is a game? How do games work? What makes them fun or frustrating? How can we understand games, talk about them, and work to evolve them as an art form?

These are the questions that formal game design theory attempts to answer. What follows is a brief introduction to the literature, theories and schools of game design that have emerged in the last 20 years.

Old School

The Art of Computer Game Design (1983)

In this landmark book, Chris Crawford defines videogames, outlines their basic structure, and investigates the affordances of interactivity. In particular, Crawford focuses on the relationship between game and player:

"A game is a closed system that subjectively represents a subset of reality. . . . Objective accuracy is only necessary to the extent required to support the player's fantasy. The player's fantasy is the key agent in making the game psychologically real."

Crawford's analysis laid the foundation for future work by separating interactive videogames from puzzles and other forms of gamelike entertainment, and describing the ways they might evolve.

I Have No Words but I Must Design (1994)

Greg Costikyan evaluates Crawford's design fundamentals and then adds a few of his own. His definition explicitly addresses the issue of *player goals* and *decisions*:

"A game is a form of art in which participants, termed players, make decisions in order to manage resources through game tokens in the pursuit of a goal."

Costikyan also asks the reader, What kind of information do players really need, and will they be able to find it in your game? What are you doing to help players compete, cooperate, socialize, or role-play? What have you done to make the player care about the outcome of the game?

FADT: Formal Abstract Design Tools (1999)

Doug Church proposes the development of a vocabulary for answering questions (such as those preceding) with precision and clarity:

"A design vocabulary is our tool kit to pick apart games and take the parts which resonate with us to realize our own game vision, or refine how our own games work. . . . Once you have thought out your design, you can investigate whether a given tool is used by your game already. . . . Using the right tools will help get the shape you want, the strength you need, and the right style."

This vocabulary is defined as:

"*Formal*, implying precise definition and the ability to explain it to someone else; *abstract*, to emphasize the focus on underlying ideas, not specific genre constructs; *design*, as in, well, we're designers; and *tools*, since they'll form the common vocabulary we want to create . . ."

Church reinforces the idea that formal abstractions can work across genres by developing FADTs for several games. In particular, he explains how videogames can empower the player by supporting *intention* and delivering *perceivable consequence*.

Recent Efforts

The 400 Project (2001)

Hal Barwood introduced the idea of game design rules at a lecture at the 2001 Game Developers Conference called "Four of the 400." Noah Falstein's subsequent column, Better by Design," in *Game Developer* magazine has evolved the project, focusing on ways in which concrete rules (such as "Provide clear short-term goals" and "Maintain suspension of disbelief") can be applied to various games. In addition to exploring individual rules, the project explores how rules trump each other, creating complex hierarchical relationships.

Design Patterns (2002)

This approach, championed by Bernd Kreimeier, proposes the development design pattern templates that resemble those of architect Christopher Alexander. At the right level of abstraction, reusable templates would help designers communicate with one another, document their insights, and formally evaluate various designs. Kreimeier investigates how we can build patterns that are "as abstract as necessary, but not more abstract."

MDA: Mechanics, Dynamics, Aesthetics (2000–2003)

For the last three years, Marc LeBlanc has conducted an interactive, two-day Game Tuning workshop at the Game Developers Conference. LeBlanc's MDA workshop framework uses FADT to analyze and tune games. It presents an approach in which games are viewed as formal systems and analyzed using formal models. MDA also holds that by focusing on the aesthetic and emotional experiences of the player, a designer can avoid common hazards and build better games.

Art Coursework to Consider

- Drawing, painting sculpture
- Anatomy, physiology, life drawing
- Animation, storyboarding
- Concept and character design
- Architecture, industrial design
- Film or theater production
- Human-computer interface design
- 2D computer art
- 2D graphics software
- 3D models: design, construction, animation

Writing Coursework to Consider

- Literary theory, creative writing
- Basic character development, context-setting
- Back-story and setting design
- Play- and screenwriting
- Interactive narrative

Sound Coursework to Consider

- Sound design, composition
- Recording, mixing, and mastering
- Music history and theory
- Sound effects, interactive scoring
- Soundtracks, sound for stage and film

For All These, Don't Forget To:

- Understand basic physics, math, and programming
- Learn basic business
- Play games

Finally, take a few physics, math, and computer programming courses. Knowing what goes on "under the hood" of game systems will make a big difference in your ability to understand your teammates.

Testing and Quality Assurance

If you play a lot of games and want to be involved in their design and creation — but don't really consider yourself an expert coder, writer, sound engineer or artist — a stint as a tester may be in order. It's a common entry-level position and will help prove to the higher-ups that you can be a dedicated team member, focused on the product despite long hours and repetitive work.

For the most part, you'll be asked to play a game as it's being developed, thinking and talking about its mechanics, dynamics, and overall play quality. You'll be reporting on your experiences with the game, writing bug reports, commenting on user interface, level design, and other game features within your session transcripts. Strong written and oral communication skills are essential.

The more you understand how games are actually constructed, the more refined your thinking about problem areas and potential fixes will be. So even if you're not a hot-to-trot programmer, take some of the programming and software management courses from that specialization.

As always, experience with a broad catalog of games is highly recommended. It will help you understand what ships (and what shouldn't) so that you can coherently report on what you test.

Testing is a great way to learn about how game products are designed, especially how a game's design evolves over time. Good testers can go far if they are willing to develop their analytical and design skills. Testers who enjoy maintaining bug lists, organizing reports, and designing tests can move up into more established QA positions, while those with exceptional ideas and an eye for design have been known to rise all the way to lead designer.

Insider tip. Testing is fundamental to the quality and playability of a game, and there are a lot of opportunities to improve

Where To?

When Crawford's *The Art of Computer Game Design* was first published, games were still little more than blips, bleeps, and crude sprite animations. Yet Crawford had a vision — a future in which games were a highly developed, expressive media on par with film and fine art. He said then:

"We are a long way from a computer game comparable to a Shakespeare play, a Tchaikovsky symphony, or a Van Gogh self portrait. Each of these artists stood on the shoulders of earlier artists who plunged into an unexplored world and mapped out its territories so that the later artists could build on their work and achieve greater things. We computer game designers must put our shoulders together so that our successors may stand on top of them. This book is my contribution to that enterprise."

What will yours be?

References

The Art of Computer Game Design
www.vancouver.wsu.edu/fac/peabody/game-book/Coverpage.html

I Have No Words
www.costik.com/nowords.html

FADT
www.gamasutra.com/features/19990716/design_tools_01.htm

MDA
www.gamedev.net/columns/events/gdc2003/view.asp?SectionID=1
www.algorithmancy.org

The 400 Project
www.theinspiracy.com/400_project.htm

Design Patterns
www.gamasutra.com/features/20020313/kreimeier_01.htm

— R. H.

Testing/QA Coursework to Consider
- Basic programming
- Software management
- Human factors, sociology
- Psychology, human-computer interaction

Don't Forget To:
- Gain written and oral communication skills
- Observe other gamers
- Gain broad gameplay experience by playing games

upon current techniques. Whenever you play or discuss games with other people, practice thinking about them in a formal, rigorous manner, and avoid overloaded words like "hardcore," "gameplay," and "fun" (see sidebar, "Game Design 101"). Also, carefully observe how different people play, so that your evaluations reflect more than your personal preferences.

Who knows, with a little experience and some classes in human factors, sociology, psychology and human-computer interaction, you could be the next Bill Fulton (see his publications at Microsoft's Playtest Research site under For More Information).

Looking Ahead: Careers in Business, Design, and Production

In truth, it's highly unlikely that you'll be hired right away to do high-level design or production work, or become CEO of a 100-person company overnight. However, if you're thinking about the long haul, or starting a project or company of your own, then you'll need skills in these areas.

Much of what's required for these positions is company- and even project-specific. It's fairly common for people in business, production, and design to start out low in the ranks, learn the ropes and rise up over time. While this makes it a bit difficult to take courses that are directly related, there are still lots of things you can do while in school to improve your chances.

Production

People in production roles make many small decisions every day. They spend a lot of time on the phone, running meetings, gathering information, managing schedules, and delegating tasks. While somewhat removed from the creation of code and assets, these activities have a major impact on how a team works (or doesn't work, in some cases). A good production staff can be the difference between a shipped, selling title and a multi-million-dollar flop.

Production work requires expert communication skills. Classes in rhetoric, written and oral communication, and technical and business writing will provide you with the means to communicate about your goals.

Production Coursework to Consider
- Rhetoric
- Written and oral communication
- Technical and business writing
- Phased project development and scheduling
- Group dynamics, workflow
- Organizational management

Don't Forget To:
- Work with people
- Read industry news and publications
- Play games

Management coursework in phased project development and scheduling (often available in advanced engineering or business programs) will help you avoid common mistakes. Classes in group dynamics, workflow, and organizational management will teach you how to build and maintain healthy teams.

Producers should be "good with people"; strong enough to make tough decisions, yet tactful and composed under pressure. Working with people face-to-face is the best way to practice these skills and develop an ability to think on your feet. Practical work experience in a professional context unrelated to games (such as law or business) is also a great way to develop these skills. If you think you'd like a spot on the front lines of game industry management, consider a production-oriented internship with a game or software development company, or in a field where similar high-pressure management situations occur.

Insider tip. Because production jobs are largely administrative, it's tempting to think of them as pencil pushing — boring and unimportant. In reality, these jobs are incredibly challenging. They require both an eye for detail and a keen sense of the big picture. This high-level perspective of the project can give production staff a clear view of what is practical for a given feature or component — and they often have final say in such matters. It's a good idea for people considering this track to read as much as they can about game design and development (such as the recently published *Postmortems from Game Developer* book — see For More Information).

Also, consider volunteering to work with industry organizations such as the IGDA and Interactive Digital Software Association (IDSA). Such work is a good way to meet people who are concerned about the industry's future and dedicated to positive change.

Business

The game industry is a hit-driven business, much like the film and music industries. It's difficult to predict what will sell and what won't, and overall, a minority of all shipped products are responsible for a majority of sales. While game budgets are small when compared to film, it is not uncommon for an original title to cost $10 million from start to finish.

Business Coursework to Consider

- Basic economics
- Supply chain management
- Marketing and sales, retail organization
- Business contracts, copyrights
- Trademarks and licenses
- Publisher-developer relations
- Content regulation, IP, and patent law
- Software production and management
- Product design and innovation methodologies
- Management of creative resources

Don't Forget To:

- Play games
- Observe gamers

Aspiring entrepreneurs should understand the basics of game industry economics — if for no other reason than to avoid making a bad investment. Take classes in basic economics, marketing and sales, retail organization, and supply chain management. Learn what you can about business contracts, copyrights, trademarks, and licenses. Study publisher-developer relations, content regulation, IP, and patent law.

Beyond this, it's important that you understand the technical structure of games and the key differences between creative and business software production cycles. Cultivate and maintain an informed view of the development process with coursework in software production and management, product design and innovation methodologies, and the management of creative resources.

Survey the technical landscape of the game industry, the current platforms and their markets (consoles, PCs, mobile devices, and handheld game equipment). Watch trends in gaming and keep tabs on sales figures. Are military shooters really still the way to go? Is this new design really that different from FRENZIED FIREARMS 3 or SNOTTY SNIPER 4? If you know the technical trade-offs and have a good grasp of what's already on the market, you're more likely to see through buzzwords and spot ideas with true potential.

Insider tip. Experience is the best teacher when it comes to appreciating quality, predicting trends, and separating the diamonds from the duds. But so is playing games and staying in touch with "online gamers," "hardcore gamers," or "kids." Take the time to visit fan sites, game retailers, and arcades (especially abroad). Do their demographics jibe with your image of the game-playing public? If not, reevaluate.

Design

Game design is probably the least-understood area of game development. It's the magic that isn't quite magic, an art that scientists and scholars still struggle to describe (see "Game Design 101" sidebar). Most people who consider a career in gaming have had ideas about how their favorite game could be changed or improved. And yet, very few people actually become designers. Fewer still are formally recognized for their work.

Is game design a genetic gift, or a skill that comes from a long, intimate relationship with the world of games and game development? No one knows for sure. But if you think you've got what it takes to be a designer, know that you'll have to work very hard — orders of magnitude harder than your competition — to prove it.

Where to start? Many designers begin as programmers, although that's not a hard-and-fast rule. A strong background in procedural thinking is essential to game design, and programming is a great way to exercise this part of your brain. Take a few computer science specialization courses, too. Learn a bit of AI, psychology, and human behavior, plus computer graphics and physics. All of this information will help you communicate with the people on your team.

And communicate you will. Designers are responsible for articulating everything from the game's high-concept and overall aesthetics to the formal treatment of individual game mechanics. They write lots of documentation and discuss their ideas with each part of the development team. As keeper of the "vision," a designer must write and speak with absolute clarity.

In addition to written and oral communication, courses in literature, film, and media studies will create a sounding-board for potential design ideas. Is your game's high-concept compelling and original, or a tired old cliché? Avoid boring ideas and washed-out themes: read books and comics, see films, and even watch a little TV.

Be a well-educated and well-informed gamer. Play classic board and card games, popular games from other cultures (such as mah-jongg or go), and physical games such as bowling, soccer, and pinball. Cultivate your understanding of what makes one activity fun and another one boring. Observe other people as they play, and ask them questions about their experiences.

Finally, get as familiar with the development process as possible. Read all you can, and when you get onto a project, talk to as many people as possible. Learn what decisions are being made and why. Participate in testing, and observe how changes to the user interface, character animations, or musical soundtrack affect the player experience. You will need to understand these things if you want to make good games.

Insider tip. While it would be nice to think that designers

Game Design Coursework to Consider

- Programming and software development
- AI, psychology, human behavior
- Computer graphics and physics
- Literature, film, and media studies
- Basic business

Don't Forget To:

- Gain written and oral communication skills
- Experience everything you can
- Read everything you can

have total control over a development project, there are many factors influencing its outcome. Marketing agendas, budgets, and scheduling have a huge impact on the way a title evolves. A successful designer should understand the financial underpinnings of the industry and appreciate how management choices influence the bottom line. Don't shy away from classes that make you think like a businessperson. A well-rounded designer can think like a producer, talk like a "suit," and still be an artist.

Beyond Coursework: Honing Your Skills

Once you've chosen a basic path and started taking your courses, it's important to develop the spaces between your classes. By this I mean that your coursework should serve as a starting point for further exploration of the game medium.

If you're interested in art, try designing some new characters for an existing title or franchise. See if you can incorporate the aesthetics of a title into your designs, or expand upon an existing theme. What other games fit into this look, feel, or genre? Play a few, then compare and contrast them. Take notes — you can often turn novel observations into an online post or article.

Programmers should definitely build games. Re-implement something from the Atari age, or get a few friends to help you build a game mod (using the UNREAL or HALF-LIFE engine, for example). Study the way that specific games implement specific features (lighting, motion, character behaviors, AI), so that you learn from the games you play.

For writers or designers, interactive fiction contests are a good place to start. Creating your own game and "design bible" is also good practice. Challenge yourself. What would a game about professional horse racing be like? Can you get around standard racing game themes? Eschew stereotypical characters and commonplace mechanics. Innovate.

School itself is a lot of work, and even small projects can take a while to come to fruition. But you will learn things when working with other people and developing your own ideas that you simply cannot learn in lecture.

Out There: Getting Your Job

In addition to your coursework and résumé or portfolio projects, there are a few things you can do to make yourself more visible when the time comes to get a job. In your sophomore or junior year, save up and volunteer for the Game Developers Conference (GDC). With student travel rates, some planning and a little luck, you can make the trip for about $500. Introduce yourself to speakers you admire, and collect business cards.

Keep tabs on the games and companies you like. Volunteer to beta-test games, look for internships, and build relationships with other students in your area. Join your local IGDA chapter. If there is no chapter, start a student chapter or club.

In March of your senior year, take the trip to GDC again. Be prepared to hand out business cards, portfolios, demo reels, and so on. By this time you should have a good handle on your prospects, places you're likely to go, and how to get there.

The End?

Game development is often viewed as highly technical — by now you should know that there's more to it than that. People who can work on teams, communicate clearly, and brainstorm new ideas (or graciously let them go when their time has passed) are always in short supply.

That being said, basic knowledge of computer programming is indeed essential for anyone interested in the production and development of videogames. Don't be afraid to try it.

When it comes to the big picture, the industry also needs people with advanced skills. Is a business, law, or graduate technical degree in your long-term plans? Maybe it should be. There's a growing community of game scholars and game journalists out there as well — perhaps you'll join them.

Of course, it's important to have interests other than coursework and videogames. Be curious. Read, take long walks, and always question your assumptions. Most of all, follow your passion. If you find yourself in your dream job and realize you hate going to work, quit and start over. Any gamer should know: it's never too late to try a new strategy.

FOR MORE INFORMATION

Gamasutra.com
www.gamasutra.com

IGDA's "Breaking In" site
www.igda.org/breakingin/home.htm

IGDA Education Committe
www.igda.org/committees/education.php

IGDA Curriculum Framework
www.igda.org/academia/curriculum_framework.php

Game Developer's 2002 Game Career Guide
https://www.gamasutra.com/php-bin/store.php?category=9

Postmortems from Game Developer, Austin Grossman, ed. CMP Books, 2002.
ttp://www.cmpbooks.com/scripts/store/vsc/store/products/1578202140.htm?L+/htdocs/cmpbooks/config/store+fzoz9044

Microsoft Games Playtest Program
www.microsoft.com/playtest/default.htm

Full S
4c