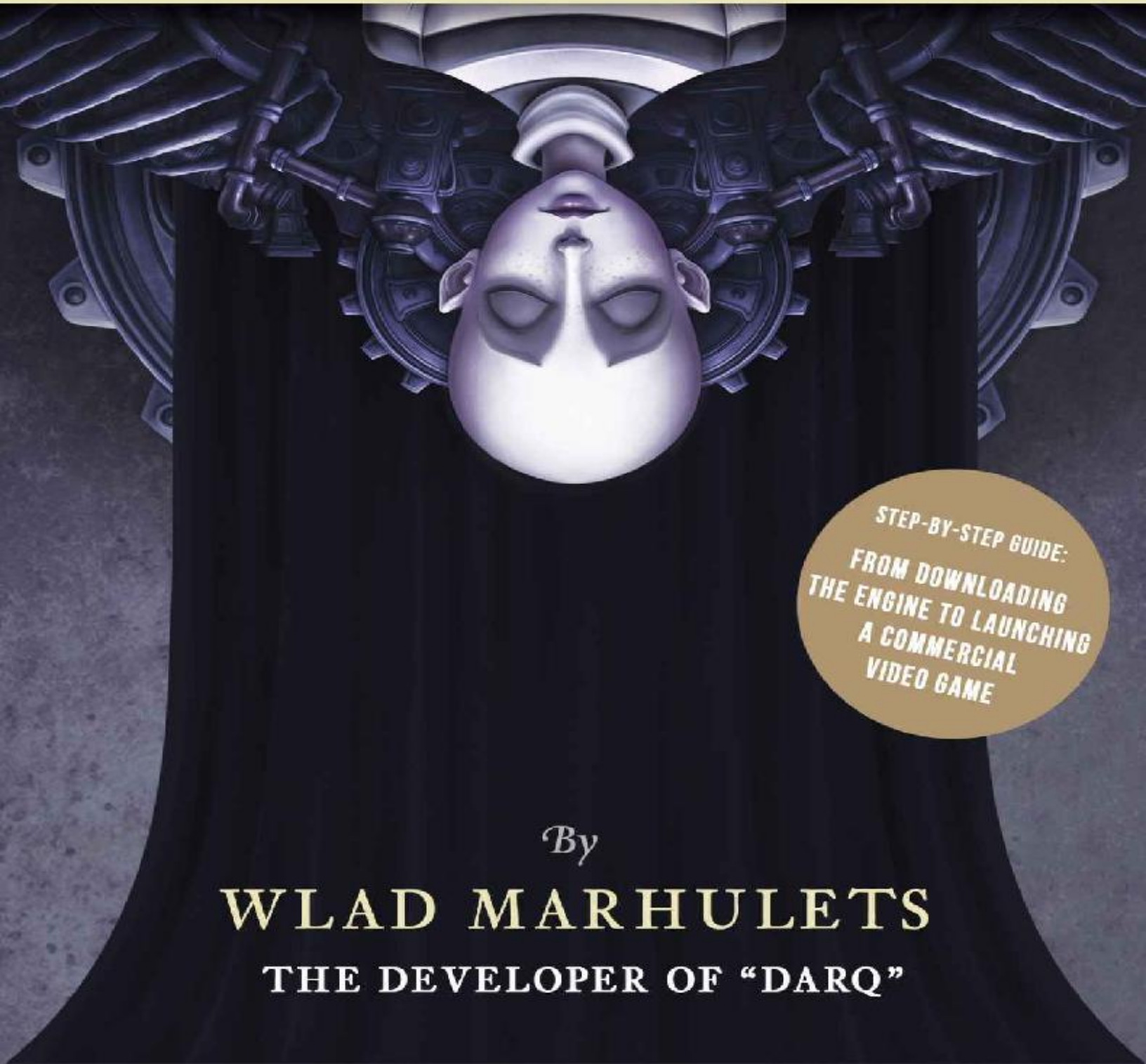
GAMEDEV

10 STEPS TO MAKING YOUR FIRST GAME SUCCESSFUL



STEP-BY-STEP GUIDE:
FROM DOWNLOADING THE ENGINE TO LAUNCHING A COMMERCIAL VIDEO GAME

By

WLAD MARHULETS

THE DEVELOPER OF "DARQ"

GAMEDEV

10 Steps to Making Your First Game Successful

by
WLAD MARHULETS

the developer of
DARQ

Dedicated to my dear friend and mentor,
John Corigliano

Who is this book for?

If you know *nothing* about game development, you're basically *me* before I started working on my first game *DARQ*. This book assumes no knowledge of game development on the reader's part.

As a first-time developer **with no prior experience in coding, modeling, texturing, animation, game design**, etc., I managed to launch *DARQ* to both **commercial success and critical acclaim**. With zero dollars spent on marketing, it was featured in major media outlets, such as **IGN, Kotaku, PC Gamer, GameSpot, Forbes**, and hundreds of others. *DARQ* won a number of industry awards, such as *The Best Game of the MIX / PAX West*, and received a user rating of 9 out of 10. *DARQ* was one of the top 50 most wishlisted games on Steam before it launched. It made it to the "Top Selling," "New and Popular," and "Featured and Recommended" tabs on Steam. Ultimately, *DARQ* became **#42 Most Shared PC Video Game of 2019**, according to Metacritic.

In my book, I share with you exactly how I did it. The book guides you through a step-by-step process of making a game: from downloading a game engine to releasing your first commercial title. If you're an aspiring developer wanting to make a living from making games, this book is for you.

Endorsements

There are many books on game development, but none of them address the mindset you need for success. Wlad is amazing for his perspective on learning, drive, and the mental frame needed to not just complete games, but to make fantastic ones like his very own DARQ. His philosophy very much reminds me of what it was like in the early days of making games for Blizzard.

Mark Kern
Former Team Lead for *World of Warcraft*, Producer of *Diablo II and Starcraft*

Reading this book is the shortest route toward a solid understanding of how to make indie games, both from creative and business perspectives.

Quentin De Beukelaer
Game Designer of *Assassin's Creed IV: Black Flag*, *Assassin's Creed Unity*, *Ghost Recon Breakpoint*, *Narcosis*

Wlad did the impossible by making his first game successful. Whether you're a complete beginner or a seasoned indie, this book is a comprehensive guide to the business of game development.

Bjørn Jacobsen
Sound Designer of *Cyberpunk 2077*, *Hitman*

This book captures the process and creates a valuable resource for upcoming developers and creators alike.

Piotr Babieno
CEO of Bloober Team, *Layers of Fear, Blair Witch, Observer, The Medium*

If you want to make games for a living, this book is bursting with wisdom that will move you closer to realizing your dream. I watched Wlad do it, and now you can too.

Richard Gale
3-time Emmy-winning Director

I hope this book inspires others to follow their dreams, no matter what the odds are. Wlad's story proves that it's possible.

John Corigliano
Oscar, Pulitzer Prize, and 5-time Grammy Award-winning Composer

Contents

FOREWORD
INTRODUCTION
CHAPTER 1: MINDSET
CHAPTER 2: PREPRODUCTION
CHAPTER 3: FUNDING
CHAPTER 4: DEVELOPMENT
CHAPTER 5: BUSINESS AND LAW
CHAPTER 6: MARKETING AND PR
CHAPTER 7: PUBLISHING AND DISTRIBUTION
CHAPTER 8: PREPARING FOR LAUNCH
CHAPTER 9: LAUNCH!
CHAPTER 10: WHAT NEXT?
AFTERWORD
ACKNOWLEDGMENTS
RESOURCES
COPYRIGHT

Foreword

As a composition faculty member at the Juilliard School, I first met Wlad in the Spring of 2007. He came to New York from Poland, wanting to become my student. He didn't speak a word of English, and even if he did, his severe stutter would have made it impossible for him to communicate. He failed the written exam miserably, as he couldn't understand any of it. Also, his chances weren't looking good from a financial standpoint. Juilliard is one of the most expensive music schools in the world, while at the time, Wlad's net worth was around three hundred dollars.

To make matters worse, he had to take an English language exam, which he subsequently failed multiple times in a row. Any other person in his position would have given up or never tried auditioning in the first place. Despite all his struggles, I saw something in him. It was his determination that impressed me. Ultimately, I decided to give him a chance.

Before I knew it, Wlad managed to learn English, passed his exam, significantly reduced his stutter, and was given the honor of receiving a full scholarship to study at Juilliard. That's how he became one of my three students. I was surprised to learn that Wlad had only studied music for five years before he met me. Once I started teaching Wlad, it wasn't long before his music career skyrocketed as major orchestras around the world began performing his compositions. His incredible ability to learn fast can be explained by talent. However, from what I've seen, talent played just a minor role in his journey. It was his mindset, work ethic, and determination that got him to where he is now.

After graduation, Wlad went to Los Angeles to pursue the career of a film composer. By that time, I knew what to expect of him. Within a few years, Wlad was working with established directors and writing music for big Hollywood movies. Later, however, I learned that while he loved writing music, he found working in the film industry draining. Just as he

signed with one of the biggest music publishers in the world, he decided to try something completely new: game development.

He knew nothing about making games. When he told me about his decision, I supported him fully and was happy he had found something he truly loved. It didn't worry me that Wlad was about to enter a completely unfamiliar industry with no knowledge, no skills, and no connections. After all, I'd seen him do it before.

I'm thrilled that his game *DARQ* turned out so well. I'm also not surprised. Against all odds, Wlad's unique mindset allows him to make his wildest dreams come true. I hope his book inspires others to follow their dreams too, no matter what the odds are. Wlad's story proves that it's possible.

John Corigliano
Oscar, Pulitzer Prize, and 5-time Grammy Award-winning composer

Introduction

As of writing this book, I'm not a veteran of the video game industry. In fact, I've only released one game so far: *DARQ*. I started working on it in late 2015 when I knew *nothing* about making video games. At first, I had no intention of ever turning it into a commercial project. It was supposed to be a new hobby that I decided to try in between film scoring gigs. One month later, I had a little prototype that barely resembled what *DARQ* looks like now. A friend of mine encouraged me to put together a game trailer and showcase it on Steam Greenlight. While Greenlight is no longer around, at the time, it was a voting platform that allowed Steam users to pick games they were interested in buying in the future. I was blown away when *DARQ* became one of the top ten most upvoted titles on the platform. *What if I could actually finish this game?* – I thought. I had no prior experience in coding, game design, level design, modeling, texturing, or animation. I also didn't know anything about marketing, community management, public relations, law, team management, accounting, or business in general. I had just a few months' worth of savings and a burning desire to change my life.

Three and a half years later, *DARQ* made it to the top 50 most wishlisted games on Steam, beating multiple triple-A titles in the race to the top. It went viral numerous times and attracted the attention of major media outlets, such as IGN, Kotaku, PC Gamer, GameSpot, Forbes, and hundreds of others. *DARQ* launched to both financial success and critical acclaim. It ended up winning several awards, including *The Best Game of the MIX* at PAX West and *Best Sound Design - Audience Choice* at Game Audio Awards. It was featured on the "New and Popular" as well as "Top Selling" tabs on Steam's front page for multiple days. A day after launch, the game made it to the "Featured and Recommended," the largest spot on the Steam front page. As a cherry on top, *DARQ* ended up being the 42nd most shared video game of 2019, according to Metacritic.

In the title of this book, I suggest that your first game can become successful. While it is an ambitious goal, it's been done before by multiple indie developers. What does a successful game mean to *you*? Coming from humble beginnings, I know what it's like to live in constant fear of running out of money. I know the stress of facing eviction, and I know what it feels like not being able to afford food and necessities. Do I consider *DARQ* release successful? Absolutely. After all, I broke even on all my expenses within a few days after its release. While *DARQ* might not compare in popularity to some hit indie titles like *Minecraft* or *Stardew Valley*, it gave me a sense of financial security and control in my life that I had never experienced before. Above all, it allowed me to continue to do what I now love the most: making games.

In this book, I share with you what I've learned throughout the development of *DARQ* and beyond. If somebody had given me this book at the beginning of my journey, it would have saved me at least a year of development time. In addition, I made a lot of strategic mistakes during the development, so it sure would have been nice to have a guide like this to warn me of all the challenges that awaited me.

Keep in mind, this book is not a complete recipe for success. The main ingredient of the recipe is *you*. It's your willingness to work hard, make sacrifices, acquire new skills, develop new habits, and take calculated risks that is going to make your first game successful. I'm here to guide you through this process and to share with you what I've learned over the years.

If you need further guidance or have questions after reading this book, feel free to reach out to me directly on Twitter (@UnfoldGames). I'll do my best to help you.

• CHAPTER 1 •

Mindset

It all begins in the brain. Paradoxically, to become a successful software developer, you first might need to reprogram your brain, so it becomes a partner, not a saboteur. Chances are, you didn't grow up in a perfect environment where self-confidence, self-growth, and self-discipline were nurtured. Maybe you have a habit of procrastinating, a short attention span, or other habits detrimental to your well-being. Looking back now, all the success I was fortunate to have in my life was the result of building the right mindset. Admittedly, there are other factors, but by developing the right mindset, you create the foundation upon which the success is built. If you don't believe you can make your first game successful, you are unlikely even to try. If you doubt yourself every step of the way, how can you convince others to believe in you? If you don't expect a lot from your first game, how can you expect your customers and the media to take your project seriously? Building the right mindset is the first step to achieving any long-term goal. Making a game is a marathon, not a sprint. It's going to take a long time.

You will need every bit of self-confidence and a set of useful habits to get through this grinding process. This chapter will teach you how to build the kind of mindset that will support you on your journey.

Excuses

All of us are capable of incredible things, yet most settle for the minimum. Did you notice how you always have just enough money to get

by, but not much more? Why is it that most of us don't have an excess of wealth, health, happiness, and fulfillment? It’s because of self-limiting beliefs that prevent us from reaching our full potential. Having self-limiting beliefs makes people give up on their dreams before making any effort to achieve them. Have you ever had thoughts like: *I haven’t started my own business because nothing I do ever works out*, or, *I'm too bad at math to learn how to program*, or, *I’ve never been good at marketing; that’s why my games wouldn’t sell.* You can’t win a battle if your mind makes you surrender before it even starts.

During the development of *DARQ*, many people reached out to me saying they'd also like to make games but couldn't because of various reasons. The truth is, the circumstances will *never* be perfect. Games don’t get made in the absence of obstacles. Quite the contrary: overcoming obstacles and dealing with hardships is a part of the process of achieving any long-term goal, such as making a game. Your circumstances won't improve until you accept full responsibility for your actions. When you let obstacles stand in the way of your dreams, you willingly give up the control over your life. While victimizing yourself might give you comfort in the short term, it won’t help you improve your situation. Don’t spend your life hiding from your dreams behind self-imposed limitations.

Before you rush to say that your limitations are not self-imposed, and that they truly prevent you from reaching for the stars, let me tell you a little bit about myself. I was born in Belarus, one of the few remaining dictatorships in the world today. Surrounded by poverty, national pessimism, and hopelessness, my family and I were lucky to escape the regime and relocate to Poland. At the time, none of us knew the language. Over the next few years, I experienced some of the most challenging obstacles that one can face. My parents ended up having one of the most malicious and destructive divorces the world has seen. I’ll save you the details, but suffice it to say that my parents’ divorce made headlines in the local newspapers. With years of emotional abuse and constant turmoil, I developed severe stuttering. It was so bad that even trying to say a single word would take me up to 30 seconds.

To make matters worse, we also went bankrupt and struggled to pay for food. At certain times, we couldn't afford utilities, like water, electricity, and gas. Winters were unbearable since we had no heating. Wearing layers of winter clothing was the only way to fall asleep.

My mother ended up abandoning me, and my father passed away not long after. Being trapped in the middle of the worst crisis of my life, I set a goal of becoming a successful music composer. To clarify, I had only studied music for five years at that point.

I knew that I had to do something—*anything* to improve my situation. I was facing starvation and homelessness. Since I didn't have much to lose, I decided to go to the American Consulate to apply for a visa. My plan was naive: I wanted to audition for the most renowned music school in the world—The Juilliard School. It's based in New York, so I had to figure out a way to get there. Not only did I not know a word of English, but I also couldn't afford a train ticket to get to the American Consulate. But I didn't let it stop me. I borrowed a bit of money and went to Warsaw for my visa interview. The interview was awkward since apart from not knowing English, I practically couldn't speak at all because of my severe stutter. The odds were stacked against me, and even more so since the American consul didn't speak Polish. However, once I mumbled "J-J-Juilliard School," the consul unexpectedly approved my visa application. I didn't understand what he said, but I'm guessing it was something along the lines: “Wow, Juilliard! I used to study music in the past too!” It was pure luck—the kind of luck one never gets to experience in the absence of action.

I proceeded to borrow more money and booked a flight to New York. Armed with 300 dollars in cash and much foolishness, I left Poland and ended up in the middle of Manhattan. I was there to make my dream come true, and I was ready to do whatever it took. The next morning, I was at Juilliard auditioning to study with Oscar, Pulitzer Prize, and five-time Grammy Award-winning composer John Corigliano. I failed a music theory test because I didn't understand a word of it. I also failed my English exam several times in a row. Giving up seemed like the only reasonable thing to do, but I didn’t. It was a period of incredible struggle—sharing the full story would require a book of its own. Against all odds, however, John Corigliano ended up accepting me as one of his students, and the school awarded me a full scholarship. Again, luck played a role. This time, it was earned by perseverance, stubbornness, and overcoming the fear of failure.

As the first semester was still half a year away, I had to figure out a way to survive in New York before I could move into the Juilliard dorms. It was a challenging time during which I went through a period of actual starvation and came close to living on the streets of Brooklyn. Ultimately, all the years

filled with seemingly unconquerable obstacles led me to the biggest opportunity of my life: studying with one of the most renowned composers in the world. As the school semester started, so did my career as a musician. While some students were busy partying and binge-watching shows, I worked every waking hour to improve my skills as a composer. As a result, major orchestras around the world started performing my music while I was still a student.

A few years later, I had my bachelor's diploma in my hand and was headed to Los Angeles to sign with one of the biggest film music agencies in town, representing such composers as John Williams and Ennio Morricone. I believe I was their youngest client at the time. Oh, and that stuttering thing? After a lot of research and hard work, it's almost unnoticeable now. What about my English? I didn't speak a word of it when I first arrived in New York. Now, I'm doing things I didn't think would be possible for me, like giving public speeches and sharing my story with you through this book.

I could have made all the bad things that happened to me into an eternal excuse. A starving orphan with no money and no ability to say a word—what a great identity it might have been! Nobody would have questioned my right to live a mediocre life as a semi-functional individual. However, somehow, I managed to turn all the obstacles that stood in my way into opportunities.

While you can't control what happens to you, you can control what meaning you apply to it. It's up to you to decide whether your circumstances are obstacles or blessings in disguise.

Find Your Purpose

I was in the early stages of my film scoring career when I first discovered game development. At the time, I was convinced that I had already found my life's passion: scoring films. You could say I was doing quite well, considering I was fresh out of college. But if I were to rate my desire on a scale of one to ten for this line of work, it would be an eight. Once, writing music for films was a dream seen through rose-colored glasses. I had imagined it would be the greatest thing in the world. Using

the principles described in this chapter, I managed to turn this dream into reality. Soon, my name started showing up in the credit rolls of Hollywood movies. However, not long after, I realized that I didn't love the process of scoring films. While I loved writing music, I found the collaborative nature of the business to be mostly stressful, emotionally draining, and sometimes abusive. Besides, while I was lucky to be making a living writing music, I was still experiencing quite a lot of financial uncertainty, as film scoring gigs didn't come regularly. Living in constant fear of running out of money only added to my disenchantment with the dream of becoming a film composer. It took a long time until I could admit to myself that I simply didn't want it badly enough. As a result, I didn't do everything I could to succeed as a film composer. I did okay, but not amazing, and I take full responsibility for that.

Every bit of success I got to experience in my life was paid for with hard work, sacrifice, and determination. *DARQ* was no exception. Having my first game achieve both financial success and critical acclaim was the most difficult thing I've ever done. What helped me get through this turbulent journey was my love for the *process*. As I was entering a completely unfamiliar industry, I had to learn everything from scratch: coding, game design, 3D modeling, texturing, animation, as well as the business side of the video game industry. I didn't know where this unstoppable curiosity was taking me, but it made me feel like a kid in a candy store. There was a lot to explore, and all of it was incredibly exciting. Gamedev tutorials became my new obsession, and so a new goal was born: *I will create a game and make every effort to ensure its success.* It scared me at first, but I quickly realized that I was more intrigued by the process of making games than scoring films. My desire to make *DARQ* was ten out of ten.

I knew that the odds of achieving my goal were slim. At first, putting a lot of time into developing *DARQ* made me doubt myself and wonder, *what if years of hard work end up being a complete waste of time?* However, the fear of failure eventually went away. I finally realized that even if the game were to fail miserably, it would have been the happiest years of my life. I was finally doing something that I loved completely. My reason for taking this major detour in my career was very strongly defined. After all, I discovered my true purpose. It had taken me thirty years to find it.

As a first-time developer, you'll be competing with giant corporations, thousands of smaller independent studios, and solo developers. Some of

them have shareholders to please, while others might have taken loans and invested their entire life's savings into their games. Everybody participating in the race to the top has different levels of motivation. In the end, those who *really* want it will learn, grow, and prosper. While others, who only kind of want it, will eventually give up.

If you're entering the video game industry with no prior experience, I encourage you just to try it for a bit. Don't spend any money, and don't commit to a project yet. Get your feet wet, so you can get a sense of what it's like. Once you have a better understanding of what's involved, be honest with yourself and answer this question: would you be making games if you had millions of dollars in your bank account? Or better yet: would you be making games if you were terminally ill and had just a few more years to live? If you answered yes, you have found your true calling—your life's purpose.

Goal Setting & Risk Management

While everybody has desires of some sort, in most cases, they are nothing more than wishes. *One day, I'll have the body of my dreams*, or, *tomorrow I'll start learning a new language.* People wish for various things, but there's no urgency behind those thoughts. Wishes don't have deadlines, and they don't have execution plans. Goals do.

It's time to turn your dream of becoming a successful game developer into an actual goal. Make it specific. Put a date on it. You will probably miss your deadline, but having one makes your goal real and creates urgency.

Many veteran developers would advise you not to aim too high, so you don't spend too much time and money on a game that will likely fail. This popular advice comes from a good place and has your financial, physical, and emotional well-being in mind. It seems like common sense to first make several small, low-quality games, release them, sell a few copies, and learn from mistakes. Eventually, you would then go for something that has more commercial potential and would take much longer to make. I think it's solid advice, especially if you're a grown-up, with a family to feed and a mortgage to pay. But if you're young, have no children, and your living

expenses are low, you may be able to afford to take more risk, if it's justified. To clarify, I'm not advocating for gambling away your life savings. Instead, I encourage you to consider taking a calculated risk if the risk vs. reward ratio is in your favor.

Aiming to make a mediocre game fast almost guarantees that it gets lost among thousands of other low-effort games released every day. Such games don't get noticed by the press and influencers, so your brand will suffer from the get-go. On the other hand, making your first game ambitious is a high-risk proposition. But it’s possible, and my story is not the only one to prove it.

> *The thing that defines running a business is risk. Your job, running a studio, is to take risk and mitigate risk. That's it. That's the whole job.*
>
> **Rami Ismail** | @tha_rami | Developer
> *Nuclear Throne, Super Crate Box, Ridiculous Fishing*

If you're young, your biggest asset is time and low financial obligations. If things go wrong, you can always start over. If your game fails, you might lose some money, but you'd have enough time left to try again. However, if things go right, you can hit it big. After all, there's only so much time in life. If you continue to set your bar too low, you will simply run out of time, energy, and motivation. Only you know how much risk you can afford to take when starting on this journey. If time is your primary asset, use it to your advantage, and adjust your risk tolerance accordingly.

When I was just starting on *DARQ*, my risk tolerance was low. In the early days of the project, I had no skills and no knowledge of the industry. It would've been insane to quit my job and put my modest savings into a game I didn't know how to make. I did *not* do that. Instead, I continued to do gigs in the film industry to pay my bills, and in the meantime, I worked on *DARQ*, trying to learn as much as possible. As I began to see more and more evidence that my game had the potential to become successful, I increased my risk tolerance. Eventually, *DARQ* started going viral, winning awards, and getting featured in the top media outlets. That’s when I finally decided to quit my day job and work on it full time. Seeing *DARQ’s* Steam

wishlists grow made me commit more and more to the project. As a result, I increased my risk tolerance further and got a personal loan to increase the size of my team. Every step of the way, I was adjusting my risk tolerance as I was monitoring *DARQ*'s popularity. While I did take risks, they were always well calculated.

When it comes to setting a goal, I recommend making it a big and inspiring one. I don't suggest you start with a big project though. Instead, I'd keep the scope small, but try to achieve the highest quality possible. Reaching high-quality standards might take time, but it will be worth it. Your game *has to* be of high quality—it's as simple as that. Otherwise, nobody will care.

Setting big goals is incredibly beneficial. It's unlikely that you will ever achieve more than you dare to dream. So, dream big. You might miss your target, but at least you'll give it your best shot and maybe come close to it. Don't be afraid of going for the top, as long as you carefully manage risk along the way.

Your goal should be big enough that it scares you a bit, but not to the point that you begin to doubt yourself. Ideally, you want a goal that makes you think: *I have no idea how I'm going to achieve this, but I'll do my best to figure it out*. At this point, your goal might seem impossible, and that's understandable. After all, you lack both skills and knowledge to make it happen. But skills and knowledge can be acquired.

Now that you have your goal and a deadline, it's time to come up with a strategy. Make a list of everything that separates you from your goal. Lacking knowledge? Write down, *watch tutorials, read articles, attend seminars*. Not having enough connections in the industry? Write down, *meet and keep in touch with ten industry professionals*. No funding? Write down, *research game pitches, crowdfunding, publishers, investors, and loans*. You get the idea. Create a list and make it as complete as it can be at this stage, to the best of your ability. Divide it into steps and put a deadline at the end of each task. And then…simply execute. You might need to revise the plan as you gain experience and learn more about the industry, but that's okay. Having a bad plan is better than having no plan at all. Take action towards your goals, even if the action is wrong. Wrong actions can lead to valuable experience. Lack of action is the killer of your dreams. Don't let your dreams be wishes. Achieving big goals is a messy and

turbulent process. You will make mistakes, miss deadlines, revise your strategy, pivot, fail, then fail some more, but ultimately, you'll make it.

Talent vs. Sweat Equity

The role that talent plays in every success story is usually grossly overestimated. When people display a high level of skill in some area, it's generally assumed that they were born under a lucky star. This couldn't be further from the truth. Success is just the tip of the iceberg. Achieving a goal is merely a short moment of reaping the rewards of incredible sacrifices and thousands of hours of hard work behind the scenes. You may hear my story and think, *of course, your game turned out well—you're talented.* But *DARQ* didn't always look and play the way it does now. I wasn't born knowing how to code, make 3D models, texture, draw, write music, understand business or marketing. My Twitter feed goes back to the early stages of *DARQ*'s development. At first, my skills were poor, and not many people cared about what I was posting.

Talent is not the sole factor that will determine your success, but it certainly is *a* factor. So, what exactly *is* talent? I define talent as a multiplier of work. For example, if you're born with an artistic inclination, you're likely to learn art-related skills at a faster rate. Needless to say, hard work still needs to be applied for it to happen. But a person with less talent can achieve better results if they simply put in more work. It happens all the time, not just in the video game industry. It's not talent that determines who gets to be at the top of their field. It's the desire to get there that makes the select few *outwork* everybody else. When going for something big, be ready to work hard, whether you're talented or not.

I spent over ten thousand hours developing the skills required to make my game successful. I'm curious to see what results *you* can produce if you give it all you've got. Don't risk your health and make sure to get adequate amounts of sleep and exercise, but do understand that it might take thousands of hours to learn new skills and ultimately to make your first game do well.

Developing games takes a lot of time, even for a triple-A studio with hundreds of employees. Time is a non-negotiable investment that you must

make when starting on this journey. No matter how talented you are, you can make your first game do well if you put in enough sweat equity. To clarify, sweat equity is the investment of time and effort into a project. If you were born with many talents, use them to accelerate your learning. If you don't particularly excel at anything, you can still learn to delegate tasks effectively and to focus on managing your team instead. Either way, you have no excuses for not pursuing your gamedev dream if it’s indeed something you want.

Habits

Very few activities you do during the day are a result of a well-thought-through decision. Your morning routine, things you buy in the grocery store, your beliefs about yourself—all of it runs on autopilot and is determined by your habits. The more you repeat a particular behavior, the more it becomes ingrained into your brain. Behaviors and thought-patterns in which you engage frequently become habits and ultimately form your identity.

If your habits make you procrastinate on important tasks, get distracted easily, doubt yourself, engage in self-sabotage, or give up on tasks before they’re finished, your gamedev dream is at risk. Fortunately, habits can be changed thanks to neuroplasticity, which is your brain's natural ability to rewire itself.

How do you get rid of a harmful habit? Simply make a daily effort not to engage in your established routine. Refusing to use the neural pathways that correspond to the unwanted habit weakens the connection until it eventually fades out. Rewiring your brain requires a lot of mental effort at first, but it becomes a lot easier as you continue with the process.

So, what are the habits that can help on your gamedev journey? The ability to focus for long periods of time would be one of them. Waking up early and getting out of bed immediately without wasting time would come in handy as well. Those are the habits that I had to develop myself. They helped me immensely on my journey. What about health habits? You're about to spend thousands of hours staring at the computer screen.

Developing good health habits is essential, both when it comes to exercise and nutrition.

Research from University College London found that on average, it takes 66 days to develop a habit. Again, at first, doing the right things will feel like an uphill battle and will require a lot of self-discipline. But with time, as the new neural pathways begin to form, things will start running on autopilot. That's the beauty of habits. Once you create the right habits, they become an integral part of your identity. You will be taking actions that bring you closer to your dream, and it won't take much self-discipline to continue to do so. Developing the right habits is not easy, but it’s a long-term investment that will continue to pay dividends throughout your entire life.

Make a list of habits you would like to get rid of and those you would like to develop. During the first week, stay vigilant and disciplined—this period is especially challenging, as your brain will fight against any change in behavior. Before you know it, you will be able to take advantage of your newly formed habits and continue to take actions required to achieve success without as much mental resistance.

Time Management

Your ability to manage time and use it efficiently will have a direct impact on the probability of your game's success. As a first-time developer, taking on the goal of finishing a commercial game is a major time investment, so naturally, you want to develop a set of skills to release your game sooner rather than later.

Many people have an adverse reaction to the term *time management*. It may seem restrictive and uncomfortable at first. However, the beauty of time management is that it allows you to have time for things that matter to you the most. And it's *you* who gets to decide what those things are.

We tend to live our lives as if we were immortal, and that's because it feels comfortable. However, it’s essential to realize how valuable a resource time really is. Life expectancy varies greatly from country to country, but let's assume you're located in one of the more developed countries in the world, where the average life expectancy is 80 years. You graduate college

and enter the workforce at 22. That gives you 58 years of life remaining. What about the average age of retirement, which is 65? That cuts your professional life down to 43. We're not done yet. Subtract the time you spend sleeping, and now you have 29 waking years left. That's the best-case scenario, assuming you're able to develop games full-time. If you're stuck at a job you hate and plan to work on your game during the weekends, it reduces your actual free time to 8 years. As you get older, your energy levels, motivation, and focus will likely decline, and you will get less and less out of the time you have left. Do you still feel like time management is a bad idea? What about the things you *really* want to do in your life? Being aware of how precious time is allows you to make educated decisions, and sometimes necessary sacrifices, to accomplish the things that truly matter to you.

I know it's unpleasant to realize how little time you might have left to make your dreams come true. However, I find it incredibly helpful and also necessary in the long run. You simply cannot achieve big goals without hard work. And hard work requires a lot of time and sometimes sacrifice. To make *DARQ*, I had to sacrifice a lot. I limited my social life. I cut down my expenses. I stopped watching shows and playing video games. When I started working on *DARQ* full-time, I tried working at least 100 hours a week. While I made sure to take occasional breaks, I realize that what I did might be seen as extreme. But it was *my* way of doing it, and I'm not encouraging you to do the same. Whatever schedule you adopt, avoid burnout at all costs.

> *Having worked in the triple-A industry, I wasn't new to crunch. However, my experience did not prepare me for this diffuse and permanent sense of urgency I felt when directing Narcosis, from 2015 to 2017. I was managing a team of six highly motivated people. We had comfortable funding, and above all, I was finally making my own game. Dream situation, right? I dived into this adventure wholeheartedly, investing every hour I could in either working on it, or thinking about it.... Until I was burned-out. It took me a couple of months to come back from it. Ultimately, with the*

support of friends and colleagues, I was able to overcome the emotional turmoil and release Narcosis to a warm reception.

Quentin De Beukelaer | @atelierQDB | Game Designer
Assassin's Creed IV: Black Flag, Assassin's Creed Unity, Ghost Recon Breakpoint, Narcosis

Working long hours might not be a good fit for everybody. Besides, it's not sustainable. There's no sense in being unhappy or feeling deprived just for the sake of making a video game. While I did have an extreme schedule, I enjoyed it. If you have a need to live a more balanced life, by all means, do that. As I said before, running a game development studio is a marathon, not a sprint.

As a first-time developer, you'll probably be wearing many hats. In other words, you'll be performing more than one job throughout the development of your game. Most indie developers perform tasks that are usually handled by entire departments of highly trained and experienced professionals in triple-A studios. If you decide to go completely solo, you have to be aware of the impact it will have on your most valuable resource: time.

DARQ was not a solo project, but it was close to it. About 5% of modeling and animation was done by a number of talented contractors. The entirety of sound design was outsourced as well. The rest of the game, which included 95% of modeling and animation, as well as all programming, texturing, level design, music, community management, marketing, PR, and publishing, was done by me. Given that half of the development time was spent mostly on acquiring skills, managing my time effectively was an important factor in bringing the project to the finish line.

Regardless of how many hats you're planning to wear, you should try to reduce the development time at all costs. Here are nine tips that will help you do that:

- Create a log to monitor how you spend your time. Once you get a full picture of how many hours are wasted on trivial tasks, you can begin to make adjustments.

- Before committing to a game idea and getting into asset creation, create quick prototypes to test your ideas. Making a prototype shouldn't take more than a day. Getting a feel for your idea in such a short time is invaluable, as it can help you determine whether or not it’s worth implementing. Potentially, it can save you months of development time.

- Set weekly and daily goals and try to hit them no matter what. This will keep your perfectionism in check. Developers tend to have an emotional attachment to their creations. It results in spending too much time on details too early in the development cycle.

- Make sacrifices. Only you know what activities can be limited or completely cut out from your schedule. Only you know what your priorities are. Do you *really* have to attend that party on Friday night? There’s no right or wrong answer, and it’s up to you to decide.

- Plan for leisure time. Listen closely to your body and avoid burnout. Keep your physical and mental health in check. Dedicate some time to getting out and catching up with friends. Spending too much time alone with your game can lead to loss of motivation and depression. Don't overwork yourself to the point when the amount of daily stress overshadows your original desire to make a game.

- Once you have a playable level or a prototype, invite a few people to test your game, preferably strangers who won't refrain from giving you their honest feedback. Emphasize that you're looking for honesty. If a feature you just spent a week implementing is not fun, you want to know that as soon as possible. Try to detect any potential problems early, so you don't build upon a faulty foundation.

- If you're planning to do community management while making your game, set aside a limited time for it. For example, one hour,

once a day. When you're working on your game, make sure your social media, discord, and email notifications are off. Get in the flow state and keep your work free of distractions.

- Consider outsourcing tasks that you're not good at and tasks that are time-consuming. If some tasks don't interest you, it's better to delegate them. I understand there could be financial challenges to overcome, but there are multiple ways to go about it. We'll discuss that in more detail in Chapter 3.

- Experiment with your working hours and determine when you feel most productive. I had never thought of myself as a morning person, but when working on *DARQ*, I discovered that waking up early and starting my workday before 5 am turned out to be the most efficient way of getting things done. There was something magical about starting to work when everyone else was asleep. It reinforced the idea that there was a good reason for me to be up so early. It was also amazing to see how much I could accomplish by noon, and there was still the whole day left ahead of me. You may find that your productivity increases during the day or at night. The key is to find what works for you.

Say you need three things to make a game: the motivation, the money, and the knowledge. If you are missing the money, you can still make a game, sitting quietly and chipping away at it after work for thirty minutes. If you are missing knowledge, you can still make a game — you can hire people with knowledge. If you lose the motivation, no amount of money or knowledge will help. Protect your motivation. Be careful of burnout. Remove people who demoralize your team. Nothing will kill your game faster than losing morale.

Rami Ismail | @tha_rami | Developer

It’s important to find a way of working that fits your personality. Happiness and well-being should be your first priority. Otherwise, you’re risking losing motivation, and that’s a dangerous territory to enter.

Working on a project that you truly enjoy will ensure high levels of motivation throughout the development. At least, that’s how it felt to me. Motivation was never a problem while working on *DARQ*. Although I had a few periods when I felt like taking a break, they never lasted more than a couple of days. Not working on *DARQ* made me miss it, and I would quickly get back to it. That’s why it’s important to enjoy the process. To me, the journey towards the goal was the ultimate reward.

Key Takeaways

Your ability to take action, persevere, overcome obstacles, and believe in yourself will become the determining factors on your gamedev journey. Don’t underestimate the importance of mindset when it comes to achieving any long-term goal.

Take full responsibility for your life. Blaming circumstances and coming up with excuses will make you feel powerless.

While talent plays a role, it’s not worth much without the “sweat equity.” It helps to be talented, but it’s your willingness to work hard that will make the most difference in your life.

Everybody has dreams, yet only few dare to turn them into goals. Goals have deadlines and execution plans.

Most decisions you make during the day are automatic, like hitting the snooze button, procrastinating on important tasks, seeking distraction when working, etc. Get rid of bad habits and create new ones that will help you on your path.

Time management is a valuable tool that allows you to focus on the activities that truly matter to you.

Prioritize your happiness and well-being; don’t work more than you can handle. Avoid burnout at all costs.

• CHAPTER 2 •

Preproduction

So you've decided to become a game developer—great! If you know nothing about it, you are basically me, three and a half years before I released *DARQ*. What do you do next? This chapter will take you through a step-by-step process of pre-production—the most crucial part of the journey. Now is the time to make a lot of decisions. What game are you going to make? What genre is it going to be? What platform will be your primary target? What will your monetization strategy be? Preproduction is the stage where you can create a clear plan that will increase your odds of success. Planning ahead will allow you to avoid investing your time and money into a potentially disastrous project. I'll show you how to prepare for the production and to plan for what happens post-launch. Yes, we'll do all that before you write your first line of code. I wish somebody had taught me this before I started working on *DARQ*.

Market Analysis

You're about to enter a new world. You probably already are a consumer of video games, but what do you know about the video game industry from the business perspective? The industry evolves fast, so there's no formula for making a successful game. Trends and market conditions constantly change, so no advice will stand the test of time. However, there are principles that will help you make the right decisions, no matter what the market conditions are.

I invite you to do as much research as you can. Try to find answers to the following questions:

- What are the current market trends?
- What games are selling on the platforms you're targeting?
- What are game genres that show an increase in popularity?
- How much does the video game market grow each year?
- Is the growth accelerating or slowing down?
- What are the most successful indie studios, and how is their business going?
- What strategies made them successful?
- What are their estimated sales?
- What are they doing that you might be able to do better?
- How many people did it take to develop a game that is similar to the one you're considering making?
- How long did it take to make it?
- What was its budget?
- What are the most popular platforms and what is their distribution model? What cut do they take? What is their demographic? Is it mostly male, female, or mixed? What age group do they represent?
- Based on the current state of the market, what trends do you anticipate in the future?

> *"The Medium" and its dual-world mechanics can only be fully realized on the next-generation hardware. Rendering two worlds at the same time requires a lot of processing power, so it simply couldn't work the same way on PS4 and X-Box One. That's why we had to be forward-thinking in our design process. Essentially, we were developing a game for the hardware that didn't exist at the time. It's important to anticipate not only trends but also future technological advancements. This approach allowed us to create unique gameplay that distinguishes our title from the rest.*

Wojciech Piejko | Lead Designer

The Medium, Observer

You should find the answers to these questions before you start working on your game. In 2007, the whole industry's worth was estimated at 45 billion USD. Some analysts believe it will reach 200 billion USD by 2023 (Source: Global Games Market Forecast by NewZoo.com). With such incredible growth, trends tend to change fast. The answers to these questions will continue to change as the industry evolves. Your success largely depends on your ability to anticipate new trends, technological advancements, and changes in consumer behavior.

I want you to understand that as you've decided to make a commercially viable product, you've entered the world of business. You need to treat what you do as a business and continue to educate yourself in this area. The process of game development is mostly an artistic endeavor, and you should enjoy it as such. But keep in mind that now you're also preparing to run your own studio, create a brand, register your trademarks, develop and protect your intellectual property, hire contractors, develop a marketing strategy, and enter into partnerships with other businesses, investors, publishers, retailers, etc. It might sound overwhelming, but you'll learn how to do all that in time. The next few chapters will guide you through this process.

What Game Should I Make?

I'm old enough to remember a time before digital distribution when indie games couldn't really exist. Physical distribution was expensive and risky, so publishers played it safe with designed-by-committee blandness. Take advantage of your nimbleness and the opportunity to express your creative vision. Don't just make what everyone else is making or what you think will sell. Embrace the spirit of indie game development and make something that otherwise could never have existed.

Dan Adelman | @Dan_Adelman | Biz Dev
Chasm, Axiom Verge

This is a fundamental question, and the only person who can answer it is *you*. You're about to dedicate yourself to a commercial product that will take a long time to develop, so think carefully.

If you commit yourself to a project that has low market potential, you're setting yourself up for failure, at least in a business sense. If you try to replicate the success of a popular game, chances are you'll create a soulless clone that nobody, including you, will be excited about. While I'm not going to tell you what game you should make, I will help you consider the factors that will likely influence the outcome of your decision. Are these factors set in stone? Of course not. Trends are always changing, and there are exceptions to every rule. You can still make a successful local multiplayer platformer for Steam. It's just highly unlikely that it will succeed because this genre is among the least popular within the PC market. So, what factors should you take into account when making this important decision?

Scope

One of the most common mistakes made by first-time developers is underestimating the amount of time it takes to make games. You may be a fan of open-world RPG titles, but it's a terrible idea to attempt to make one as your first project. Even if you settle for a small game, it will still take an enormous amount of time and effort to get it done. Be very careful when deciding on the scope of your game. Does it really need to have all the features you have in mind? Does it need to be online? Does it have to have a complex crafting system, or is this a feature that is not essential to your game idea? Don't assume you can make a clone of your favorite triple-A game that had hundreds of people working on it for multiple years. It's better to make a simple game that is highly polished than to quit a large project because it's simply too big to finish. While there are many paths to success, there's one thing that all popular games have in common: they all get finished.

Marketability

Another common mistake made by first-time developers is trying to market their games way too late. You need to consider the marketability of your game as early as possible. Unless you have a large marketing budget, you have to rely on the organic reach on social media, the press, content creators, streamers, and word of mouth. The most effective piece of marketing material is an animated GIF showcasing a few seconds of gameplay. GIFs are commonly shared on Twitter, and other social media. If they show something truly eye-catching, they can go viral. Unlike Twitter videos, GIFs begin playing automatically as they show up in the feed. As a result, they tend to attract the most attention. Twitter is also where the gaming press likes to hang out, so getting a lot of retweets on a GIF often leads to press coverage. *DARQ's* first coverage in Kotaku, IGN, and PC Gamer came from some of the GIFs that went viral. For this to happen, your game needs to be very visual. It doesn't have to look like a triple-A title with high-resolution textures and detailed 3D graphics. But video games are a visual medium, and your core gameplay idea needs to be presentable and easy to understand in a GIF format. Is this a must? No, and as always, there are exceptions. However, using GIFs will give you a better chance at gaining exposure and establishing a following.

> *You should ask yourself two fundamental questions: What makes your game special? What would a player remember about it? When we started working on Observer, the game didn't have the mind-hacking mechanic. A cyberpunk horror game about a detective didn't make for a memorable hook, so we worked hard to find a unique angle for the game. Everything changed when we added the Dream Eater feature – the ability to hack into people's minds. From that point on, Observer became known as "that game about cyberpunk detectives connecting to people's minds." The difference*

was enormous, and it allowed the game to stand out from the crowd.

Wojciech Piejko | Lead Designer
The Medium, Observer

The hook and competitive advantage

"The hook" usually refers to a core mechanic that makes your game stand out from the rest. It's often a mechanic that adds a new twist to an established genre. A good hook makes everybody say "wow" as soon as they see it. Ideally, it should have a novelty factor and be quick to understand. As an indie developer, you need it badly. If you decide to make another strategy game with nothing that differentiates it from other titles in the genre, it's not going to have a broad audience. Please understand that you're not only competing with new releases. Most players are already engaged in playing their favorite titles. They also have backlogs of games they don't have time to play. You have to give them a reason to play your game. Your game has to offer something that others don't. For example, the hook of *Braid* is the ability to rewind time infinitely. That's what separates it from other puzzle/platformer games. *Super Hot*'s hook is that time only moves when you move. Without that feature, it would've been just another first-person shooter. *DARQ*'s hook is the ability to walk on walls, ceilings, and manipulate the environment. Without its "hook," it would have been just another puzzle game. Please note: all these examples are extremely visual and easy to understand within seconds.

Innovation always comes with the risk of limiting your potential audience, but it's a risk *you* can afford to take. Since your budget is likely to be low, you don't need to sell millions of copies to break even. Serving a smaller niche can still make your game profitable enough to make it a lucrative and sustainable business.

Core emotion

What is the core emotion you want your players to experience when playing your game? As a game designer, you need to know the answer to that question at the very start of your project. The emotion you're

trying to target should become the determining factor behind every decision you make from now on. Simplifying your idea to one emotion will help you answer a lot of questions: *Is this art style right for my game? Does this music support the vision of my project? Why should we use a checkpoint system, or why shouldn't we?* Games that are built around a single emotion tend to feel immersive and consistent.

For example, *Darkest Dungeon* accomplishes this masterfully. Every aspect of this game works together to evoke the emotion of hopelessness. It's not often that a game shows you the emotional toll of dungeon crawling. The overall mood is pessimistic and dark. The art design alone is enough to convey the game's core emotion. It's worn and somber, featuring hand-painted characters with an abundance of thick black lines and shadows. It helps support the idea of the psychological damage the heroes endure as a result of your greed-driven orders. Notice that after a successful mission, the comeback to town is accompanied by grim music. Battles feature quick two-frame animations that keep you on the edge of your seat. Instant attacks with no frames in between serve as a reminder of how quickly your party can get wiped out. The lack of checkpoints and permadeath during missions makes the stakes high, and your chances of making it through feel low.

Length

It's generally desirable for your game to be long, or better yet, infinite. With this in mind, you should find a way to extend the length of your game as much as possible. If you're making a linear game, one way to do this would be lowering the bar when it comes to asset quality and detail, so that more content can be produced faster. You could also increase replayability by randomizing certain sections of the game, or using procedural generation so that each playthrough is unique.

Genre

Look at the list of the best-selling games on the storefronts you're targeting. Should you try to mimic those trends? Not necessarily. After all, the trends change rapidly and trying to chase them might not be a good strategy. However, being aware of what genres are *unpopular* and

what makes them unpopular is important. After all, you don't want to release a game that nobody wants to play. Ideally, you should choose a genre that not only fits into the parameters discussed earlier but also interests *you* as a consumer. Make a game that *you* want to make. Just make sure there's an audience for it. For example, it's a lot easier to make a tycoon game if you've been playing tycoon games all your life. You'll naturally have more ideas that can make your creation truly unique and interesting to your target audience. It's also a lot easier to keep your motivation levels high when you're working on something that excites you.

Strengths and weaknesses

Knowing your strengths and weaknesses is incredibly important when making a plan for your first game. Chances are, you will be doing most of the work on your first project, so your skillset needs to match the game you're making. Are you good at art? Great—make a game that is extremely visual and requires little coding. Are you a good programmer? Fantastic—make a game with simple art but interesting game mechanics. Needless to say, you don't have to do everything by yourself. You can complement your weaknesses by hiring contractors or forming partnerships with other developers.

Unique look

Your game doesn't have to look hyper-realistic, but having a distinct art style is a significant advantage. It makes your game instantly recognizable among thousands of other generic-looking titles. Think of games that don't necessarily look hyper-realistic but are visually distinct. You can't help but picture them once you hear the title. *Minecraft*, *Return of the Obra Dinn*, *Factorio*, *Thomas Was Alone*, and *Terraria* are a few examples. As most games tend to look generic, being instantly recognizable from a visual standpoint will make marketing your title a lot easier.

Recognizable characters

Don't be afraid to exaggerate features to create memorable and recognizable characters. A good character design prioritizes simplicity

of shape with unique, recognizable features. The advantage of having a likeable and recognizable protagonist is unparalleled.

Creating a protagonist in a game world creates a myriad of opportunities for game derivatives. Be it second installments of a game, DLC, merchandising, books, board games, comics, film, and animation – the options are endless. There was a time when you had to wait around and pitch these opportunities to specialist parties, but now you can literally become your own multi-faceted brand with every game you design using readily available digital tools. Creating a game that can evolve into a successful brand is the dream that we all covet.

Scott Millard | Managing Director at *Feardemic*

Platforms

In theory, the more platforms your game appears on, the more sales it can generate. But it's also not that simple. For example, mobile games are consumed, marketed, and monetized very differently than PC games. The Steam player base is mostly male and leans towards darker games. Nintendo Switch has an equal gender representation, and their audience prefers family-friendly games. Each platform has its own demographic, genre preference, monetization practices, verification process, and barriers of entry. As a first-time developer, you might not be able to afford to launch your game on multiple platforms at once. Signing with a publisher might make it possible, but that comes at a cost. The pros and cons of signing with a publisher will be discussed in more detail in Chapter 7.

Porting your game to consoles could be time-consuming and costly. Whether you do it alone or with the help of a publisher or a port-house, it's just not as easy as releasing your game in the PC market. With platforms like Steam, GOG, Epic Game Store, and Itch.io, you can choose to self-publish your game with little effort involved. Therefore, I recommend

making your first game with the PC market in mind. If your PC launch proves to be successful, you can consider expanding to other platforms. The advantage of this approach is that you'll have the option to delegate porting to a port-house company using your PC revenue.

I'd recommend avoiding the mobile market, at least as your primary platform. With thousands of apps released every day, it's currently oversaturated and will only become more crowded in the future. Gaining visibility on mobile platforms mostly involves paid advertisement. Signing with a major publisher might be necessary to have a shot at getting your game noticed. Also, the pricing of mobile games is problematic. Apps are expected to be free, so you would have to design your game around in-app purchases and ads. I can't see this process being inspiring or something that you could do as your life passion. Maybe some can, but this approach is not for me. If you choose to sell your mobile game at a set price, you can't expect to charge for it what you would on any other platform. Even a seven-dollar price tag is considered expensive. It's a top tier reserved for big franchises and triple-A ports. Also, bringing a game from mobile to PC might be problematic because it will always be seen as a "mobile port"—something that is not respected among the PC player base. Mobile games can be extremely profitable, but the barrier of entry is so high that it mostly remains the domain of a few big corporations.

Target Audience

Now that you have a better idea about what game you're going to be making, you should research who your target audience is. Apart from the PC market demographics, you should also know the target audience for your game's genre. For example, pixel art games try to target older players who grew up playing similar-looking games in the past. Fast-paced shooters with flashy graphics tend to be made with a younger audience in mind. Teenagers and people in their early 20s tend to hang out on TikTok or Instagram, while older players are probably spending more time on Twitter, Facebook, and Reddit. It's important to understand your target audience as much as possible. You should be able to answer the following questions:

To what age group does your audience belong?
This will affect the choice of your social media channels and the tone of your communication.

What are the main countries you are targeting?
Apart from translating your game to your target audience's languages, you will need to consider certain cultural factors. For example, China bans games that show blood or religious symbols. Australia doesn't allow drug and alcohol references.

What motivates your audience? Are they looking to relax after work, or are they competitive in nature?
Knowing your audience's lifestyle preferences will not only affect how you market your game but also numerous game design choices. Determine how your game fits into your target audience's lifestyle. Consider such factors as personal values, interests, hobbies, and personality type.

What games are they playing now?
Your target audience is already playing their favorite games. Why should they stop? Your game needs to bring significant innovation to the genre and offer something that other games don't.

One of the biggest mistakes I see first-time developers make is trying to appeal to everyone. It results in creating weak brands and generic games that stand for nothing and communicate nothing. Knowing who your target audience is will allow you to serve them better, provide something of value, and communicate more efficiently.

Engine

There has never been an easier time to launch a career in game development. The tools are immensely powerful, making game development accessible on an all new level. People often ask me

where to get started. The problem is an embarrassment of riches. There are so many game editors, engines and tools that it can seem daunting. My advice is this: Don't worry about picking the "correct" or "best" tool. First of all, such a thing does not exist, and second, all the name brand engines out there are more than enough that you really can't go wrong starting with any of them. Just start. That's the trick. Take the dive, and learn and reasonably master one tool. You will find that once you know one tool, learning the next is simpler, faster, easier, and you soon will know your favorites.

Mark Kern | @Grummz
Team lead for *World of Warcraft*,
Producer of *Diablo II* and *Starcraft*.

Before starting on your project, you will need to decide what engine to choose. Most developers use either Unity or Unreal.

While there are many other lesser-known engines to choose from, I encourage you to select either Unity or Unreal, just because of the fantastic teams behind them that continue to improve their software regularly. Courses and manuals are available for free, which makes learning universally accessible. Both Unity and Unreal have been around for a long time, so there are large communities of developers who are there to help each other when problems or questions come up.

The choice between the two engines mostly comes down to personal preference. Both have incredible real-time rendering capabilities, as well as an extensive set of features. Game development is still incredibly hard, but it's easier and more accessible than ever before.

One of the main differentiators between the two engines is the programming language. Unity uses C#, while Unreal relies on C++. I encourage you to check out both of them. Keep in mind that both engines can be used for free while you learn and develop your game from start to finish. Once you start generating a significant amount of revenue, both companies want you to pay.

At the time of writing this book, Unity is free to use as long as your studio generates less than 100,000 dollars in revenue or funding a year. If your studio makes between 100,000 dollars and 200,000 dollars, Unity wants you to pay a monthly fee of 40 dollars. If the number is greater than 200,000 dollars, the monthly fee goes up to 150 dollars. Price tier upgrades come with special benefits, like discounts in the asset store, free courses, free plugins, etc.

Currently, Unreal Engine is free to use as long as your game doesn't exceed one million dollars in gross revenue. If it does, Unreal wants you to pay a five percent royalty. Epic Games, which owns Unreal Engine, recently announced that the five percent royalty would be waived if you sell your game on the Epic Game Store.

When I was starting my gamedev journey, the payment structure behind each engine was completely different. When deciding between the two, I chose Unity. I quickly developed an appreciation for the simple interface, well-written manuals, and the active community of developers who were always there to help when I needed it. Thanks to the free C# tutorials provided on Unity's official website, I never had to take paid courses.

Keep in mind that both Unity and Unreal may change their business model and their pricing strategy in the future. Before you make a decision, you should visit both websites and check their current offers. Download both engines to get a feel for them and play around. This way, you'll be able to make an educated decision and choose an engine that's right for your project.

Software

Your game engine is where all assets and code come together, but what other software are you going to need? It depends on the type of game you're going to be making, your personal preference, and your budget. Before we delve in, I want to make it clear that I'm not getting paid for endorsing any of the software or resources in this book. My recommendations are based purely on my personal experience.

If you're making a 2D game, **Photoshop** for drawing and **Spine** for animation would probably be your main tools. **Spine** integrates well with

both Unity and Unreal, so it makes for an easy workflow.

If you're working on a 3D game, you will need some good 3D modeling and animation software. **Maya** seems to be the industry's choice. **3Ds Max** is often used as well. Both are pricey, but luckily, there's a free and open-source alternative: **Blender**. It's not particularly user friendly, but it's a powerful piece of software that became my personal favorite when it comes to environment design. **Blender** heavily relies on keyboard shortcuts, which might make it difficult to use at first. However, once you learn all the useful shortcuts, you'll be able to get a lot done quickly.

If you're looking for a digital sculpting tool, it doesn't get any better than **ZBrush** by Pixologic. It's both the industry standard and one of my favorite pieces of software when it comes to asset creation. When paired with a tablet, it becomes a powerful and intuitive application for character and environment modeling. It makes retopology a breeze and offers a variety of other user-friendly features, like normal map baking and UV unwrapping. It's remarkable how easy it is to turn detailed character meshes into low-poly models with perfect topology, ready to be rigged and animated. All *DARQ* characters were made in **ZBrush,** and I can't recommend it highly enough.

When it comes to texturing, **Substance Suite** is all you will ever need to make your models look as good as those in triple-A games. The suite comes with a number of tools. **Substance Painter** is similar to **Photoshop**, except that it allows you to paint across multiple channels at the same time, like diffuse, normal, height, roughness, metal, emission, and others. There's also so much you can do with smart materials, particles, masks, and brushes—it works like magic once you get the hang of it. **Substance Designer** is a more complex tool. It uses a node-based system to create tileable, dynamic and modular textures. They can be saved as materials with exposed variables that can later be used and modified in other applications, such as **Substance Painter**. Finally, **Substance Alchemist** makes it possible to create materials from photos and high-resolution scans. Combined with a constantly expanding library of materials, **Substance Suite** is a powerful texturing software.

For audio functionality, you can settle for whatever your engine has to offer, but in most cases, it won't be enough and will require a lot of additional coding on your part. If sound plays an important role in your game, you're going to want to use audio middleware, such as **FMOD** or

Wwise. The former is free to use if the project's budget doesn't exceed 500,000 dollars. I made it my choice for *DARQ* because its simple interface felt very intuitive. **FMOD** has an excellent series of YouTube tutorials that will show you what it can do for your game. **Wwise** seems to be the industry standard, although its interface didn't seem as user-friendly to me. Arguably, both offer the same functionality. **FMOD** and **Wwise** integrate well with Unity and Unreal, so do your research and decide for yourself. Apart from the middleware, you will probably need a simple audio editor to make small edits. I use **Audacity** for that; it's free and open source.

Regardless of what kind of game you're making, if you're going to be painting, 3D sculpting, or texturing, I highly recommend getting a drawing tablet. You can get a decent one for as little as 150 dollars—it makes a world of difference.

Budget & Schedule

Even the most experienced industry professionals struggle with setting realistic deadlines and calculating budgets accurately. There are so many variables in the process of game development that following an exact milestone schedule or staying within the budget is often impossible. One thing remains constant about the whole process: things are going to take longer than expected, and you will probably need more money than initially planned.

When I first started developing *DARQ*, it was supposed to be a simple 2D point-and-click game. My goal was to finish it within half a year while working on it part-time. However, the more potential I saw, the harder it was for me to settle for something mediocre. I ended up starting it over from scratch multiple times, and ultimately made the transition into 3D, having to learn 3D modeling, rigging, animation, and other skills that come with the territory. In the end, *DARQ* took about three and a half years to develop. It sounds like a long time, but about half the development was part-time. Also, a lot of time was spent on learning new skills. Let it serve as a reminder that a seemingly simple game can take a long time to finish. Even if you keep your game's scope small, you'll be surprised by how long it will take to finish it.

While your schedule and budget might be way off, there's a benefit to planning your production. I encourage you to set a rough deadline for the prototype, various production milestones, polish stage, storefront integration, beta testing, and launch. Almost certainly, you will have to modify your schedule as you go, but having self-imposed deadlines can increase your productivity. Having no deadlines will make it harder to focus on the big picture. A production schedule will prevent you from obsessing about little details that ultimately make little difference.

As for the budget, try to calculate how much money you will need to finish your project. There are a lot of factors that come into play when it comes to that. What are your current expenses? Do you have to pay rent, or do you have the ability to live with your parents or split living expenses with roommates? Are you planning to work solo? Are you hiring contractors? Are you considering quitting your job at some point? The more work you do by yourself, the longer it will take, and the cheaper it will be. Are you young enough to consider time a less valuable resource than money? I know that at this point, you lack the experience to create an accurate budget, but having a number in mind will help you plan at least for the near future. Don't worry if the budget you come up with far exceeds your personal savings. There are multiple ways to get funding, and we'll discuss that in detail in Chapter 3. For now, have a rough estimate, or just a guess. You can adjust both your budget and schedule as you get a better sense of your workflow. As you make progress with the game, you'll become more accurate at making projections.

One more bit of advice on the subject: the last stage of development is likely to take much longer than you think. Creating the pause menu, testing, bug fixing, localization, storefront integration, and polishing take a lot of work. Don't rush to release your game in a half-finished state. The last 10% of the development can make a difference between a successful launch and a flop.

Price Point and Monetization

While making the game of your dreams is a romantic idea, you also have to make sure that it generates enough revenue to justify all the time and money spent in the process. Breaking even should be your worst-case scenario. Ideally, you want your game to generate enough revenue to fund your future projects.

At this point, you should have at least a vague idea about what kind of game you want to make. Now, it's time to do more research. What are the other key titles that are similar to your game idea? How much did they cost at launch, and how many hours of gameplay do they offer? What is the replayability value of those games? What other features do they offer? Are you going to be able to match what they provide and potentially add more features? If your game can bring similar value, you should price your game to reflect that. Unless your game offers can't match the experience provided by your competitors, don't rush into thinking that having a lower price will help you sell more copies. The opposite may be true. Launching your game at a low price can create a perception of low-quality.

Having said that, keep in mind that players want to be treated fairly, and rightfully so. Delight your customers by offering more for less. Don't charge more than your game deserves. That's a bad way to introduce your studio to the world. Greed will be met with bad reviews and high refund numbers. Playtime, retention, replayability, and quality are the key factors you should consider when pricing your game. Also, consider implementing multiplayer if it applies to your game. It can be quite a challenge for a first-time developer, though. Do your competitors offer it? If you're unable to add it to your game, make sure to reflect that in your price. Can't match your competitor's full controller support and other features? Adjust your price accordingly.

Keep in mind that a higher price will result in a longer lifespan of your game. Following the launch, you will be able to run promotional discounts, which will result in sales spikes. Companies usually discount their games gradually, increasing the discount amount over the years. Pricing your game too low doesn't give you much room to profit from a 50% discount. Games with higher base prices can still generate a lot of revenue, even with a 75% discount.

Also, you can decrease or increase the base price of your game after launch. An increase in price is likely to cause a backlash, unless it's the result of the game going from Early Access to full release and is announced

well in advance. A decrease in the base price is rarely noticed and might not have a massive impact on sales. It’s better to keep the base price unchanged for a long time and to offer time-limited discounts. Once the game is considered “old,” it may be appropriate to lower the base price to stay competitive as the newer, more expensive games in your genre are released.

It’s good to have a target price point in mind when you start developing your game. You should have a clear idea of the features you will have to implement to fit into that price tier. Before the release, it’s a good idea to confirm that the price you chose is adequate. Ask some of your followers what they would pay for this game. If what they say significantly differs from the number you have in mind, you should reconsider your price point.

One of the ways you can maximize your revenue is to release frequent updates. Timely bug fixes are a must, but content updates can be an excellent way to please your player base and give content creators and streamers new reasons to keep playing your game and showcasing its new content.

Large content updates could be released as DLCs (downloadable content) and made available for purchase. DLCs can include additional levels, new playable characters, cosmetic changes, a soundtrack, and other kinds of content. It’s good to think about potential future expansions while you’re still at this early stage. Having a series of DLCs planned out is a common practice when designing a game and should be considered a part of preproduction.

If your game becomes popular, selling merchandise or licensing your IP to a third party can become an additional stream of revenue.

Replayability

Getting more playtime out of your content is very desirable. Not only do players get more bang for their buck, but it also helps to keep your game in the spotlight longer. Apart from things like procedural generation and randomization of various game parameters, there are other ways to add replay value.

Achievements

If you're releasing on Steam, GOG, or consoles, those are relatively easy to implement. While some don't care about achievements, there's a large community of players who enjoy achievement hunting. Trying to unlock all of the achievements available in a game often becomes a reason for replaying it entirely. To my surprise, the *DARQ* community expressed a lot of interest in achievements. Many requested that I add more to the game. There's also one more benefit to including achievements in your game: it results in more video content, guides, and discussions attempting to figure out how to unlock some of the most difficult ones. To achieve good results, your game has to be designed around achievements, instead of treating them as an afterthought. Make sure your achievements are actually challenging to unlock. Integrate them within your game well. Easy achievements are unlikely to generate a lot of word-of-mouth buzz, and they can annoy some players. Who wants to be taken out of the game's immersive experience with a pop-up notification saying: "Congratulations, you've completed the tutorial!" Make your achievements hard to unlock and reward players who take the time to unlock them. In-game rewards could include extra skins, secret weapons, concept art, or anything else.

Speedrunning

A speedrun is a playthrough of an entire game, or a part of it, in the shortest time possible. Speedrunners represent a large and active community that is always on the lookout for games that allow for showing off their skills. Many of them make a living by speedrunning games in front of live audiences on Twitch. So, how do you make a speedrun-friendly game? It's simple: make it challenging. If it was easy to beat a game, speedrunning it wouldn't be impressive. *Dark Souls* series and *Cuphead* happen to be among the most popular games in the speedrunning community. You can probably guess why. They are extremely difficult, so speedrunning them generates a lot of excitement. Apart from making your game challenging, try to avoid linearity. For example, if your game is split into levels, make each level have more than one way of beating it. Consider designing various shortcuts that a speedrunner might discover. Perhaps shorter paths could introduce more risk or increase the number of enemies. Also, adding

randomization would increase the challenge of a speedrun and make the whole process a lot more fun for the community. If you choose to design your game with speedrunning in mind, expect to see plenty of video content from the community.

Modding
While modding could be a challenge to implement for a first-time developer, it can potentially make a big difference in your game's lifespan. The modding community is enormous and is responsible for the success of many titles. In some cases, modding is the main reason certain games manage to stay popular over multiple years or even decades. Having your community produce new content for your game is an incredible opportunity. New content results in more video coverage and more attention for your game.

Team Assembly

While most indie developers work in small teams, some manage to make successful games completely solo. The decision is yours, and it will mostly depend on your skillset and financial resources. If you choose to work with a team, you'll likely be the one assuming the role of project manager. Given that your first project is probably not going to be developed in a central office, you'll be managing your team remotely.

Here's how the process of team assembly could look:

Determine the tasks you want to delegate
Try to follow some coding and art tutorials. Are you having fun? Do you see yourself enjoying the process of coding in the future, or is it torture for you? Do you see yourself improving at asset creation with enough practice, or is this something you will want to delegate? Decide which tasks you'll be able to complete by yourself and which tasks you will need to outsource to contractors or employees.

Find candidates
You can either post job offers online or reach out to the professionals you're interested in directly. Artists, designers, coders, composers, and other specialists have dedicated online communities; all it takes is a little browsing. You can also find various specialized groups full of potential contractors on Facebook and Reddit.

Interview & sign contracts
Interview your candidates and look through their portfolio and work experience. The quality of their work matters much more than their education. Their attitude and communication skills are as important as their technical skills. Once you're ready, sign both NDA and Work for Hire agreements. More on that in Chapter 5.

Establish your way of communication
There are a lot of options available. You can keep in touch with your team on **Discord**, or have a private **Facebook** group where you keep everybody posted on the project's status. Assigning tasks and setting deadlines could be done in a shared **Google Spreadsheet**, **Trello**, or another similar application. You can take care of version control via **GitHub**, while file exchange can be done through **Dropbox** or other cloud services. You can choose to communicate with your contractors privately via phone, email, or **Skype**. It's also a good idea to have a group video chat once in a while—it helps everybody feel a part of the team, and it keeps morale high.

Game Design Document

The game design document (GDD) includes detailed documentation of your game concept. It might not be necessary to have one if your team is small, but writing a GDD is a helpful exercise regardless. Above all, it will help *you* to achieve much-needed clarity when it comes to the vision of your project. It can also come in handy when talking to investors and publishers. Whether you work alone or with a team, it's essential to know what you're making before jumping into production.

Here are the sections that a typical GDD would contain.

Summary of the game
Give your readers a description of what the game is all about. What is the genre? What emotions are you looking to evoke? What other titles inspired your game concept, and what makes it different? This section doesn't have to be lengthy. Two paragraphs would do the job.

Gameplay
This is the main section of GDD, where you describe how the game plays, main mechanics, game loops, player's progression, game goals, etc.

Characters
Who is the protagonist? Who are the supporting characters?

Story and narrative
What's the story arc? Where does it take place?

Art style
What is the game going to look like? If possible, provide custom-made concept art and sketches. Alternatively, present a collection of image references and describe how they refer to the game. Is the game 2D or 3D? Is it realistic or stylized?

Sound and music
What kind of style of music are you going for? What role does it play in the game? What sound effects are going to be used?

Technical description
What engine, software, and middleware are you using?

Platforms & monetization strategy
What platforms, storefronts, and retailers are you going to partner with? What is the price tag for this game? Are there DLCs planned?

Demographics
What is your target age group? Where is your audience located? Is the game going to be translated? If so, to what languages?

Extra ideas
Are there any features you're planning after the launch? Do you want to implement multiplayer or modding support in future updates? What other ideas do you have that are not on the current to-do list?

Key Takeaways

Learn as much as you can about the market you’re about to enter.

Consider various factors when deciding what game you’re going to make: such as scope, marketability, competitive advantage, and others.

Decide what engine you’re going to use. You'll likely go for Unity or Unreal. Either of them would be a great choice.

Estimate your game’s budget and create a schedule for major development milestones.

Adding achievements, making your game speedrun-friendly, or creating modding tools are effective ways of increasing your game’s replayability.

If you’re not going to be a solo developer, decide what tasks you want to outsource. Interview as many candidates as possible and assemble a team that will assist you on this journey.

A game design document might not be necessary when working with small teams, but it could help you achieve clarity as to the vision of your project.

• CHAPTER 3 •

Funding

As a first-time developer, you should do your best to keep the production costs as low as possible. Pitching your game to investors and publishers while your game is in early development might not be a good idea. Since you're a beginner, you're considered a high-risk investment. Even if you manage to get some money early on, you will have to give up a lot in exchange—potentially, a big revenue cut and the IP rights. It would be a terrible way to start a business. If you want to receive funding from either an investor or a publisher, you have to have something to show—a proof that your game will make money. Ideally, you want to have a polished demo, good wishlist numbers, and an active community built around your game. Therefore, it's best if you find a way to develop a portion of the game and grow your community without needing much capital. However, getting your game to the finish line without spending a dime is virtually impossible. So, let's look at the options that might be available to you.

Reducing living expenses

Before we explore different ways of getting funding, let's first look at your expenses. Is there any way you can reduce your monthly burn-rate? After all, if you can cut down your cost of living, you'll have more money to spend on game development. For example, if you buy a large latte every morning, it amounts to about 1,500 dollars a year. If you spend three years

making your game, that’s 4,500 dollars’ worth of lattes that could cover your development expenses. Imagine all the Facebook and Instagram ads you can buy for this much money.

If your skillset allows you to become a solo developer, your total budget for the game will equal the total of your living expenses. This is a dream scenario for somebody who is young and just starting out. Additionally, if you still live with your parents, you don’t have to worry about rent, which makes this journey so much easier. Your main focus should be acquiring skills as fast as possible while working on your game. At this age, you still have a lot of time ahead of you, so use it to your advantage. Work hard, make your first game, and try to market it by yourself. You will learn a lot in the process. If at all possible, consider releasing it without involving a third party and keep 100% of net revenue. This is a rare privilege to be entering the industry with such an advantage.

If you’re older and your living expenses are substantial, consider doing everything you can to reduce them. Moving to a smaller apartment and sharing it with a roommate can bring your monthly burn-rate down significantly. If you happen to be located in a city where cost of living is high, consider relocating to a smaller town.

If you’re serious about making your dream come true, you have to be willing to endure short-term inconveniences and sacrifices. Depending on the amount you have saved up, you may be able to get through at least a part of the development. As your game evolves and your community grows, it’ll get much easier to get a loan or to make a deal with a publisher or an investor.

Patreon

If you’re actively involved in building your community, your following will grow as you continue to develop your game. If you do your job right and provide value to your audience, your followers may consider supporting your development process. Patreon is the most popular platform that allows fans to support their favorite content creators. You may find the platform to be a good fit for you, especially if you make a lot of YouTube videos or simply post behind the scenes content on Twitter or Instagram.

While this approach might take some time to produce significant results, it can certainly work. A number of indie developers make enough money through Patreon to pay their bills while working on their games full time.

Part-time development

If you're currently employed, you shouldn't rush to quit your job. If game development is your true passion, you'll make time for it. You can work on your game and continue to learn and increase your skills every waking hour whenever you're not at your job. Again, consider making some sacrifices. Cut down on partying, watching TV, or other unproductive activities that don't bring you closer to your goal. If you want to make your game successful, restructure your life accordingly. By the time you have a playable demo, you can start looking for funding so you can quit your job and focus on your game full time. Or maybe you have enough money saved up to make that move without involving an investor or a publisher? Whatever you do, make a responsible decision based on your risk tolerance. Quitting your job is a risky move, so give it a lot of thought before you do it.

Grants

Try to find out if you're eligible to receive a grant. Your government may have a program that provides funding for game development or small businesses in general. If nothing like that exists in your country, look for other resources. Also, be on the lookout for disruptive technologies and new consoles. Companies that bring new tech to the market need content to build their customer base. No developer would spend time creating content for a platform that has no audience. That's why such companies often give grants to encourage developers to create content for their platforms. Not long ago, there were plenty of grants available for the development of VR games. Nobody knows what the next big thing will be. Whatever it is, it might become a source of free money for your studio.

Personal loans

One of the ways you can get access to capital is to ask relatives and close friends for a loan. If you're lucky, you may end up with a long-term zero-interest loan. Who do you know is in a position to write you a check? Asking for money might not come naturally to you, but you're not risking anything by doing so. Get out of your comfort zone and give it all you've got. You don't want your game to fail just because you didn't try hard enough.

Crowdfunding

When done right, crowdfunding can be an effective way to both fund your game as well as gain massive exposure. Some of my developer friends managed to raise substantial amounts of money through Kickstarter and IndieGoGo.

Here's what I learned from them:

- Crowdfunding platforms allow you to raise funding from your fan base, not strangers. Don't assume that if you build a Kickstarter page, people will just show up and give you money. This misunderstanding is the main reason why so many campaigns fail. Only consider crowdfunding if you already have a large community of people who are excited about your game. Can you get pledges from strangers? Yes, but only after your campaign becomes popular thanks to the first wave of pledges that comes from your fans.

- Running a campaign is a full-time job. It starts a few weeks before it launches and ends a few weeks after. Making the page look visually appealing, creating a trailer, and coming up with the rewards takes a lot of time. Also, shipping rewards and keeping your fans updated is very time consuming. Keep in mind that crowdfunding takes a considerable amount of time that could have

been spent on development. Is the amount you're asking for worth this kind of time investment?

- Kickstarter takes 5% of your funding if your campaign succeeds. You get none of the funds if your campaign fails. IndieGoGo allows you to keep pledges from partially funded campaigns but penalizes you for failing to reach your goal by taking a hefty 9% cut. If you do reach your goal, the fee drops to 4%. Now, think about other deductions. Money raised through crowdfunding is considered a taxable income, so a portion of your pledges will be spent on taxes. And of course, there are rewards. Did you offer any physical merchandise? You have to deduct that from your total as well. And then there are shipping costs. Be aware that international shipping costs vary dramatically depending on the destination. If you're not careful, some pledges might actually end up costing you money.

- The key to crowdfunding success is to get a significant part of your campaign funded within the first day. If your page looks appealing and the conversion rate of visits to pledges is high, you may get featured on the platform's homepage. That's how you can gain some extra visibility and potentially acquire more pledges from people who didn't know about your game before.

- In order to get a lot of traction within the first day, you may need to expand your reach and contact other indie developers who already have succeeded on Kickstarter or IndieGoGo. It's relatively easy for them to send an update to thousands of their backers mentioning your campaign, especially if your genre and target audience is similar to theirs. The chances of getting such a boost from somebody you cold-email are low. You might get better results if you build those relationships in advance, or at least offer something in return.

- By the time you run your campaign, you should already have your store pages set up, ready to collect wishlists. If things go well, you are going to have a lot of people looking at your campaign page.

Not all of them are going to pledge their hard-earned money, but at least some of them might be interested in wishlisting your game. Place the appropriate links in your description so you can convert all that traffic to wishlists.

In conclusion, running a crowdfunding campaign is a lot of hard work. It's also somewhat risky. Asking for a small amount might make people question your chances of finishing the project. Asking for a large sum of money increases your risk of not reaching your goal. If you fail to reach your goal, you can't delete the page after your campaign is over. It will remain in the Google search results forever. Having a failed campaign suggests that there isn't enough interest in your game. It can be seen as a red flag by investors and publishers if you ever decide to seek funding. It might also potentially discourage journalists from covering your game.

Publishers

Many indie developers dream of signing with a publisher. Is it a good idea? It may be, depending on your situation. Publishers provide many services that you may need. For example, they can:

- Provide funding
- Provide feedback
- Conduct quality assurance rounds
- Take over marketing or delegate it
- Fund your participation in video game conventions
- Secure distribution deals
- Assist with console ports or delegate them to port houses
- Assist or take over community management

While it sounds great in theory, keep in mind that the more you ask of them, the more they will ask of you in return. As a first-time developer, you are a high-risk bet, so you may be offered deals that might not benefit your studio in the long term. While some publishers have excellent track records and are known for treating their business partners fairly, others might try to

take advantage of your lack of experience. If you're ever given a contract to sign, have a good attorney take over negotiations for you. Preferably, choose someone who has expertise in the video game industry. Game publishing will be discussed in more detail in Chapter 7.

If you're determined to work with a publisher, I would advise you to wait until they approach *you*. If you get contacted by a publisher, it means they have already researched your game. They know it has an audience, and they know they would be missing out if they didn't contact you. In every negotiation, it's better to be the less interested party. Letting publishers come to you gives you leverage that you wouldn't have otherwise.

When *DARQ* became one of the most upvoted games on Steam Greenlight, it attracted a lot of attention from the press. As a result, my social media following started to grow fast. Within half a year, I had received a dozen offers from various publishers, including a few major companies. I negotiated with some of them but ultimately decided to go solo. I have nothing against working with publishers though. It's just a matter of finding the right partner. As of writing this book, I don't rule out the possibility of partnering up with a publisher for console releases of *DARQ*.

Personally, I never reached out to a publisher and never had to pitch my game. All the publishers I had the pleasure of meeting reached out to me first. I believe many will reach out to you too when your game starts showing up in their Twitter feeds. Publishers watch upcoming games very closely; they might have already seen yours and are keeping it on the radar. You won't help your case if you try to pitch it before they're confident the game is worth their time. Waiting is sometimes the best thing you can do.

Investors

As a business owner, you will be facing seemingly impossible obstacles on a regular basis. Financial problems can be especially draining, as they take your focus away from the project. It's important to have a support system that can help you overcome whatever issue you may be struggling with. Whether it's family,

friends, co-founders, or investors, don't hesitate to ask for help if you need it.

Toms Martin | Investor (Cyclone Capital)

If cash is all you need, it might be a good idea to look for an investor, not a publisher. With an investor, you keep control over your project throughout its lifetime. You control both the pricing, discounts, and bundle deals. Unlike publishers, investors are unlikely to ask you for the IP rights and won't put their name on your game. Your company will be in the spotlight, not theirs. Also, most investment deals expire, either when a certain cap on earnings is reached or after a number of years have passed.

As your game starts accumulating significant amounts of wishlists on Steam, you're likely to hear from investors. If you need money now, you can try to approach them yourself. Needless to say, letting them approach you first gives you more leverage in negotiations, but there are times when you simply can't afford to wait and need to act fast.

Deals with investors usually involve a one-time cash infusion in exchange for revenue share. There's likely to be a recoupment period in which the investor receives a larger cut until their investment is paid back. After that, the revenue share would drop to a smaller cut. Additionally, you might want to negotiate the cap on the investor's total earnings, so they are not attached to the project eternally. As with any other contract, you want to get a good attorney to negotiate on your behalf. You may think it's expensive, but not getting legal advice when you need it will likely cost you much more in the end.

Apart from individual investors, you may come across various investment funds. The most popular one is **Indie Fund**—their investors are the veterans of the video game industry.

Alternatively, you may want to consider personal investors: your family, friends, and colleagues. People who are close to you can become your business partners and provide you with funding in exchange for revenue share or equity stake in your business. Before you give up on this idea, think carefully if you know anybody you can ask. Chances are, you do. All it takes is getting out of your comfort zone and asking.

Once you have a proven record of profitable games, you may consider making a deal with an angel investor or a venture capitalist. In both cases, you would be giving up an equity stake in your company in exchange for capital. Angel investors are private individuals who tend to make smaller investments, typically under one million dollars. Investing is not their full-time job, and they typically don’t want to be directly involved in the company’s day-to-day operations. Venture capitalists are full-time investors who look for opportunities to acquire equity in companies that have high growth potential. They tend to provide large amounts of capital (over one million dollars) to help a company grow rapidly. In addition to an equity stake, venture capitalists usually expect to have some level of control, such as voting rights.

Key Takeaways

There are many ways to fund a game. Don’t worry if you don’t have much money saved up.

One way to make your game profitable is to limit your expenses. The less you spend on the development, the more money you will make in the end.

Don’t rush to quit your job. Being able to pay your bills while you develop your game in your free time is not a bad option when you’re just starting out. Once you get funded either by a publisher or an investor, you can quit your job to work on your game full-time.

While most publishing and investment deals are based on revenue share, there are ways to fund your game without having to give up a percentage of revenue. Patreon, Kickstarter, grants, and personal loans are often used by indie developers to stay afloat.

If you’re in need of substantial capital, consider making a deal with a publisher or an investor.

If you’re asked to sign a contract of any sort, always have a professional attorney on your side.

• CHAPTER 4 •

Development

Now that you’re done planning your project, it’s time to get to the fun part: making your game! You may feel intimidated at first, but the good news is that learning game development from scratch *is* possible. Making your first game successful *is* possible. Winning awards for it *is* possible. I’ve done it, others have done it, and so can you.

Principles of Good Design

Game design is the process in which both artistic and technical elements of the game come together in fulfillment of the designer's vision. In other words, it's a set of principles according to which your game operates. You may already have a game idea that you’re excited about.

While ideas are important, their value cannot be determined without seeing the finished product. It's the execution that matters and creates value. Similarly to a film director on a set, who makes both artistic and technical choices while following a screenplay, a game designer's role is to make a series of decisions that bring a game idea to life.

What makes for good game design? The answer largely depends on the type of game you're making. You should research the best-selling games in your niche and carefully examine the game design choices that were made by their respective designers. It doesn't mean that you should simply copy the design of those games. Game clones that don't bring anything new to the genre rarely do well. Unless done for educational reasons, cloning games is

also morally questionable. However, before you decide to break genre-specific design conventions, you first need to know what they are.

While each genre has a set of design conventions, there are also general design guidelines that are known to improve a player's experience. None of them are set in stone, though. My goal is to simply give you a set of principles that historically have proven to produce good results. As a game designer, it's up to you to follow them, expand on them, or come up with your own set of principles. After all, breaking rules is the prerequisite to innovation.

- **Focus on one core mechanic and develop the world around it**
  Most games can be narrowed down to a single mechanic. For example, jumping is the core mechanic of the classic platformer *Super Mario Bros*. If you were to use jumps as a means to get over gaps only, the game would quickly become tedious. But the brilliant designer Shigeru Miyamoto knew better. He used the core mechanic of jumping to break bricks, get power-ups, kill enemies, avoid projectiles, and even complete levels by jumping onto flag poles. He built multiple levels, each presenting a unique set of challenges that utilized jumping to overcome them. In other words, the core mechanic never changes, but the context in which it’s used keeps evolving. That’s what keeps it fresh.

  Just like in any other art form, it’s preferable to focus on a single idea and develop it intelligently rather than jumping from one idea to the next. Think of your game as an essay. It needs to introduce a single idea, develop it, provide arguments, add supportive details, and end with a satisfying conclusion. Presenting too many ideas at once is an equivalent of changing the subject every sentence. It might seem exciting, but it’s not an effective way of communication.

- **Teach the player how to play your game**
  The best way to teach your player how to play your game is to introduce controls and game mechanics gradually. Don’t overwhelm the player with everything your game has to offer. If possible,

consider giving your player the joy of discovery instead of telling them exactly what to do by displaying a wall of text. Triple-A games usually settle for text hints, but you don't have to. Your players don't need to be told what to do every step of the way. Instead, design your game in such a way that the player can discover the main mechanics and controls simply by exploration. Then test their understanding of the mechanics by repeating the setup with a slight variation. Allow them to figure out how to use the mechanics they just learned to overcome new challenges. This principle might not work well in a complex strategy game, but it can be quite effective in games with simple controls. Text hints deprive the player of the joy of discovery. In addition, they're less memorable than the direct experience. If the player figures out how to use a game mechanic, they're much more likely to remember how to use it next time.

I utilized this principle in *DARQ*. The game begins in a small apartment, and there's only one action available—to lie in bed. That's how the player discovers the action button. By pressing it, the protagonist gets teleported into the world of dreams, which starts in a narrow dungeon passage. There's seemingly no way out. However, when the player approaches a wall, the character touches it, as if preparing to do something. That's how the player discovers the ability to perform a wall-walk, which is the core mechanic of the game. What happens next is the repetition and the reinforcement of what the player has just learned: the action button and the wall-walk button. The player uses the action button to put one of the bridges in a vertical position. Since that traps the player between the raised bridge and an obstacle, the player quickly figures out that any vertical surface can also be walked on using the wall-walk button. This little tutorial section allows the player to learn how the game works through discovery, with no text at all. It allows for keeping the player engaged and surprised. Having seen lots of streams and video playthroughs of *DARQ*, those discovery moments are usually accompanied by strong reactions.

- **Make your game hard to master**
  A sense of progression is one of the most motivating factors that keeps players engaged. Most of the games that utilize this principle have XP systems, leveling up, skill trees, and other ways of making your character more powerful as the player makes progress in the game. Some games take an entirely different approach. They make your *real-life* skills a part of the equation.

  Dark Souls series is a popular action RPG franchise that is known for throwing you into the world where enemies seem way too powerful, and boss fights are so challenging that they will make your heart pound. The game doesn't get more forgiving as you go, but it does get easier as your real-life skills improve. Mastering a game like that requires excellent reflexes, perfect timing, knowledge of enemy attacks, and a lot of time.

- **Give the player a clear goal**
  Create the main objective as well as smaller intermediate goals. Communicate them to the player effectively, so there's no confusion as to what the next objective is. You can accomplish this with cutscenes, quest journal, dialog, or other means appropriate for the type of game you're making.

  Stardew Valley utilizes this principle masterfully. You're given the objective to restore greatness to your grandfather's farmhouse. As a sandbox game, *Stardew Valley* doesn't give a lot of importance to the main objective, but it is there to provide the game with a sense of purpose. And there are plenty of short-term goals: farming, exploration, leveling-up, mining, interacting with the locals, quests, etc. There's always something to do, which results in high player retention.

- **Reward your player often**
  Both punishment and reward are powerful motivators that keep players engaged. However, if you punish your players more often than you reward them, they might feel too discouraged to continue. Reward your players for making progress in your game. The reward

depends on the type of game you're developing, so think of something appropriate for your genre. It could be secret collectibles, points, power-ups, cut scenes, new weapons, artifacts, etc.

- **Establish visual language**
  Games rarely benefit from complete realism when it comes to environment design. If games were to strive for realism, they would be hard to navigate. One of the most popular methods of guiding the player through complex environments is through the use of light. Areas that are more lit naturally attract attention. The player is much more likely to head towards a lit area than a dark one.
  You can also think about establishing a visual language in which certain objects or colors have meaning. I'm sure you're familiar with the cliché of exploding red barrels. They're red because they're easy to spot in the middle of a fast-paced battle. *Tomb Raider* remakes make use of ropes wrapped around doors to indicate bow targets. *Mirror's Edge* uses color to help the player navigate the environment and to make split-second decisions in fast chase sequences. Orange surfaces, such as walls, guide the protagonist in a general direction. Red surfaces indicate important areas critical to progression through the level.

- **Use landmarks**
  Navigating unfamiliar environments in video games is often difficult. Your field of view is smaller than in real life, and assets tend to be used repeatedly throughout the world. To solve this problem, game designers often use landmarks. Also known as points of interest, they help the player navigate the level. Landmarks should be visible from afar and visually distinct from the rest of the surrounding architecture or landscape. Ideally, they shouldn't look the same from every direction, and there shouldn't be too many of them in close proximity. Making them instantly recognizable will help the player get a better understanding of their position in the world.

  Journey, a critically acclaimed indie adventure game, places a point of interest in the opening scene. Starting in the desert, with no sense

of direction or purpose, a traveler spots a giant mountain that takes up a large portion of the sky. There's a large opening on top, with a bright light shining through it. Although the mountain is miles away, there's nothing else in the view to catch the player's attention. Within seconds, the game manages to give directions one can't ignore. The player has to head towards the mountain to find out what the source of the light is. The landmark remains visible throughout various sections of the game and serves as a navigation tool.

- **Provide immediate feedback**
  Make sure to communicate clearly when the player has achieved a certain state, such as completing a quest, leveling up, taking damage, or being heard/spotted by an enemy, etc. Ideally, you want to provide feedback in different ways at the same time: using visuals, sound, and maybe event controller vibrations. For example, solving a puzzle in *DARQ* always ends with a camera shake, a distinct sound effect that is louder than other sounds, and a change in lighting.

- **Make randomization feel fair**
  Random number generator (RNG) is an algorithm often used in games to add a degree of randomness to the gameplay. RNG can control the amount of damage inflicted, the number of enemies spawned, etc. There are types of games that heavily rely on randomization.

  The problem with using true randomization is that it can produce unwanted results that feel unfair to the player. For example, if a player attacks an enemy and misses 20 times in a row, they're most likely going to stop playing your game and leave a negative review saying how unfair it is. Missing 20 times in a row is a rare but perfectly plausible outcome of using RNG.

  Why is it a bad idea to rely on the default RNG algorithm? To help you visualize it, let's imagine RNG as a simple coin toss. If you toss

a coin, you'll get either heads or tails. The probability of getting either is 50%. But if you continue to toss the coin indefinitely, it's only a matter of time before you get a 100 tails in a row. That would be an unlikely outcome, but it's bound to happen eventually. It's because each coin toss is an independent event, and the probability of getting tails remains at 50% regardless of the previous outcomes. In other words, the chance of getting tails doesn't decrease as you continue to get multiple tails in a row.

Why is this a problem? If your customers spend a lot of time playing your game, it's only a matter of time before a portion of them experience the unlikely outcome of the coin-toss effect. For example, if your game relies on the default RNG algorithm to decide if the player manages to dodge an enemy's attack, some players will experience a hundred failed dodges in a row. That would certainly *feel* unfair.

The solution to this problem is to modify the RNG algorithm. Coming back to the coin analogy, you should apply some restrictions to the tossing experiment. What if each coin toss was *not* treated as an independent event? What if you put a cap on how many times the same outcome can be repeated? For example, you can make ten consecutive tosses always produce five heads and five tails in random order. Once ten tosses have occurred, reset the cycle and start over. Even though such an algorithm wouldn't produce a truly random outcome, it would eliminate the risk of frustrating your players.

Feel free to experiment to find the best solution that works for you. The goal is to create a *perception* of randomness—something that *feels* both random and fair.

- **Have a strong game loop**

  "Game loop" simply means a cycle of activities the player will be doing when playing your game. That cycle would repeat in a loop, so your goal is to make it both engaging and varied. One of the most popular game loops consists of three parts. Most games follow

it in one way or another:

- **I. Action** - the player goes on a quest/takes a risk/wins a boss fight
- **II. Reward** - the player gains experience/levels up/gets gold
- **III. Progress** - the player spends gold on new gear/gets a new quest

As you can see, each part of the loop has a different emotion behind it. It's a cycle of tension and release. Needless to say, not every game has quests, leveling up, buying gear, etc. It's just an example. Think about how this core loop applies to your game concept and what you can do to create at least three different kinds of gameplay that are contrasting in nature and create a sense of progress.

There's so much more to learn about game design. If you're interested in exploring this topic in more detail, why don't you head to GamedevBook.com and subscribe to my newsletter, so I can send you a comprehensive list of 100 game design principles (it's free).

Best Way to Learn

I've been asked this so many times: *How did you learn all this stuff?* It always surprises me to hear this question. It implies that there's a trick to it, or some kind of secret.

I never went to school to learn game development, and I never took paid courses. *DARQ* was my first project, and although I started it from scratch three times, I had never done smaller games or tutorial projects before. I went into game development without knowing *anything* about coding, game design, level design, modeling, texturing, animation, project management, budgeting, accounting, and marketing. All theoretical knowledge on these subjects is available online for free. My primary resource for learning coding was the official C# manual on Unity Engine's website. In most cases, using their official manual would be more than enough to learn the basics of coding. I'm sure there's plenty of free resources available for

Unreal Engine users as well. When you start coding, your code will be terrible—accept that. The only way to improve is by doing, so acquire as much experience as you possibly can by working on your project. When it comes to other skills like modeling, texturing, game design, and animation, I mostly used YouTube tutorials. There are so many to choose from. At the end of the book, I list resources that I found particularly useful on my journey.

If all the knowledge is freely available online, what is the missing component? It's hard work. You still need to put a lot of hours into developing the skills you want to master. Making quality games doesn't happen overnight. No game is guaranteed to succeed, but if you want to turn the odds in your favor, you need to master self-discipline and work consistently every day. That's why the first chapter of this book is dedicated to developing the right mindset. The results will follow if you're willing to pay the price. Hard work always pays off sooner or later. Everybody who has achieved success had to pay the price, and you will have to as well.

If you have no prior experience with anything even remotely related to game development, start with reading coding manuals and watching video tutorials. If you learn something new, put it into practice immediately. Write some code, see how it works. Get a feel for it. Don't put too much pressure on yourself—it doesn't have to be perfect. The goal is to enjoy the process of learning and to have fun. If you don't enjoy the process, you're likely going to give up before you finish your game. So, focus on fun. Be an explorer of this new exciting territory. Find joy in learning new software, and don't let thoughts of self-doubt get in the way.

As you get a better understanding of how the engine works, what the principles of programming are, and how to make low-quality game assets, try to make a simple prototype that showcases the main mechanics of your game. Don't worry about quality at this point—just try to make it work to the best of your ability. Once you have it working, start over. You just learned a whole lot of new skills and managed to apply them. I'm sure you're now able to make a better version of the original prototype. Can you think of ways to improve what you made previously? Can you make your assets look better? Can you structure your code differently? I want you to see how much you've learned since the first prototype—it should give you a confidence boost. You can learn anything you want, and that's just the beginning.

Focus your project around features and content that you enjoy making. If you do not enjoy your work, you won't do it well and you won't put in the hours. When you enjoy your work, you do not need discipline.

Oskar Stålberg | @OskSta) | Developer
Bad North, Townscaper

If in the process you discover that you simply don’t enjoy some of the tasks you’re trying to master, it’s a sign that you need to delegate them. Don’t force yourself to do things you don’t enjoy—it’s unsustainable. The only way you’re going to get through this long and turbulent journey is if you really like the process.

Iterative design

You may be wrong to think that you could work secretly in your cave, like a genius poet until you produce this great game that you'll share with the world. Instead, see yourself as a cook, in search for a delicious recipe, relentlessly cooking the same dish many times, testing it, and thoroughly integrating feedback.

Quentin De Beukelaer | @atelierQDB | Game Designer
Assassin’s Creed IV: Black Flag, Assassin’s Creed Unity, Ghost Recon Breakpoint, Narcosis

It’s virtually impossible to develop the best possible version of the game without revising it countless times through trial and error. The process in which the game gets built in cycles is called iterative design. Each cycle usually involves four steps that repeat over and over again throughout the development:

1) **Prototyping**
   In this step, a quick prototype of a proposed feature gets created.
2) **Playtesting**
   The prototype is tested by the team.
3) **Reevaluation**
   The team determines if the proposed feature is good enough to be implemented into the game.
4) **Implementation**
   If the prototyped feature makes it to the game, it gets implemented.

The process involves a constant reexamination of every aspect of the game. Is the gameplay engaging enough? Does this feature make the game more or less fun? Is the UI text big enough? It's important to get back to the drawing board to reevaluate your decisions. Cutting out useless features and making gradual improvements to the project is an efficient way of developing video games.

Test your game often. Have others test it too, and ask for honest feedback. Regular testing rounds can help you detect design issues early and potentially save you lots of development time.

For example, the first level I ever made for *DARQ* was, for the most part, grayscale. A lot of assets looked generic, and there were no unique elements in the environment that would help the player navigate the level effectively. As a total newbie, I was completely unaware of these issues. When I invited a few people to playtest the level, it became clear to me that my creation was simply terrible. My playtesters were confused. I kept hearing: *Wait—did I visit this location before?*

The first round of testing was brutal; it made me realize how much I needed to learn before I could have a playable level. But the harsh feedback was also necessary—it was a real wake up call. It made me change my approach to level design and art style completely. I started experimenting with adding a little bit of color and making the environment look more distinct, with unique details and memorable points of interest.

It's important to approach each failure as an opportunity to learn and to get better. This attitude allowed me to stay equally excited and motivated throughout the hardships of development. The more iterative cycles the game went through, the better it got.

Accessibility

This is an important topic, and giving it the attention it deserves is simply the right thing to do. A certain percentage of players have various impairments that will prevent them from experiencing your game unless you add a number of accessibility features. Don't worry, most of them are relatively easy to implement, so it's mostly a matter of awareness.

1) **Motor Impairment**

   Build an input system that allows for remapping controls. Many motor-impaired players use specific controllers or keyboard configurations that allow them to play games using either one hand or no hands at all.

   On the launch day of *DARQ*, one of the first playthrough videos I got to watch was made by a disabled YouTuber. He was paralyzed from the neck down. He managed to reconfigure controls in such a way that it became possible for him to play the game just by pressing buttons with a pencil held in his mouth. His skills were simply incredible, and I couldn't believe what I was seeing. Watching him play my game was one of the most rewarding moments of the launch day.

2) **Vision Impairment**

   There are three commonly recognized color blindness types: red-green color blindness, blue-yellow color blindness, and complete color blindness. The simplest way to make your game accessible to color blind people is to avoid solely relying on color in your game design. While color is a powerful tool to convey information, try to add supportive elements, like shapes or sounds. For example, if green boxes are power-ups and red boxes are explosives, make sure you make their shapes and sizes different too. Also, put graphical elements or icons on them to help further distinguish between the two.

   Another way to deal with color blindness is to allow the player to

change the color pallet of your game depending on the kind of impairment. This topic is quite complex, but luckily there are third-party tools available on Unity Asset Store or Unreal Engine Marketplace that do all that work for you.

Apart from color blindness, there are other kinds of vision impairment. If your game has a lot of text, make sure you add a feature that allows players to change the font size. It will be greatly appreciated not only from an accessibility standpoint but also as a matter of personal preference. Many players choose to sit far from their screens, playing with a wireless controller. Without such a feature, they wouldn't be able to read the text in your game. It will become especially important later when you're ready to start porting your game to consoles. Console players tend to sit far from their screens by default.

3) **Hearing Impairment**

   If your game has spoken dialog or cutscenes, make sure there's a way to enable subtitles. Many players who are not hearing impaired choose to activate subtitles anyway, so it would be a welcome feature. Remember to add the ability to adjust the font size of the subtitles.

There are more advanced features that can improve accessibility of your game. The best way to explore solutions is to reach out to TheMighty.com—a supportive community for people facing health challenges. Look them up and ask to speak directly with people who have various accessibility needs. They would be happy to guide you through the process of making your game more accessible.

Good Coding Practices

I could probably write a separate book on this topic. There's just too much to cover, so I'll leave it up to you to continuously improve your programming skills by reading manuals and watching tutorials. I still want

to give you a few tips that I wish somebody would have given me when I was starting out. If you're new to coding, don't worry if you don't understand most of them. These tips will come in handy when you become more familiar with programming.

Don't worry about writing bad code at first
It's unavoidable. A year from now, you will feel ashamed to look at your old code. But that's the only way to learn.

Keep it simple
If you're the only programmer on the team, there's no reason to overcomplicate your code just to make it look more professional. For example, there's no need to write complex systems for things you're only going to need to use once.

One script for one task
Keep your scripts simple and task-specific. For example, have your character controller separate from camera functionality. Keep your character animation code separate from all the rest.

Avoid dependencies
Avoid creating dependencies between scripts. In other words, don't make one script rely on another script to work. This way of coding will allow you to reuse scripts multiple times throughout the game. It will save you a lot of time as a result.

Add comments to your code
When you come back to a script you wrote a year ago, you may not remember how it works. So make sure to add comments when you code. On the other hand, don't add more comments than needed. Excessive commenting takes precious time that could be spent on other, more important tasks.

Use good naming conventions
Don't make the mistake of calling a variable *x* or *y*. You might remember what it does the day you write it, but a week later, you'll find your code completely illegible. It's better to have long descriptive names that make the code easy to read. For example, a variable named

isCharacterGrounded is easy to understand. A function named *LevelUp* is self-explanatory. The name length of variables, functions, and classes has no impact on the performance of your game, so take advantage of that. Also, descriptive names drastically reduce the need to add comments, so it will save you a lot of time.

Think of the input system as early as possible

Implementing controller support might seem relatively simple at first, but if we're speaking of supporting all controllers and keyboard remapping, you should probably purchase a third-party solution for handling input. There are many available online, whether you're using Unity Asset Store or Unreal Engine Marketplace. I didn't buy one, and I regret it to this day. I had to write my own system for keyboard remapping and partial controller support. It's an incredibly complicated and time-consuming process, so buying one will save you a lot of trouble.

Figure out how to handle localization

If you're planning to translate your game to multiple languages, you should think of a good system to accomplish that task. There are various ways to approach this. Ultimately, all solutions come down to replacing all your in-game text with string variables. Usually, an array of strings would be loaded from a corresponding language file, saved either in .XML or .JSON format. It might sound like a difficult task, but it's actually relatively easy to write your own system. It was one of the first things I did in the early days of working on *DARQ*—it just took a bit of research. If you feel like it's too difficult to handle now, you can buy a ready-to-use system online.

Think about saving, loading, and checkpoints

There are numerous ways to approach data persistence. Study this topic and don't delay implementing it. You want to have a working save-and-load system before your game grows in content. If your game is complex and there's a lot of data to save, it might be too difficult to write your own solution from scratch. Again, you can purchase one of the systems available online.

Study other people's code

Once you have a basic understanding of programming, and once you can read code without feeling completely lost, it's time to see how experienced developers approach writing character controllers, camera controllers, game managers, and other kinds of scripts. Study the scripts that came with your engine and purchase a few more to compare different coding styles. Delve deep into them and learn as much as you can by following the code. It will teach you a lot about how coding is done in game development.

Focus on reusability

Last but not least, try to write code that you can reuse in your future projects. Scripts like camera controller, game manager, dialog system, and others can be coded in such a way that they could be later imported to your upcoming games. That's what triple-A studios do, and you should too. There's no need to reinvent the wheel each time you make a game. It will save you a lot of time, and you'll be happy you did it.

Sound and Music

Good sound design and an effective music score can elevate your game significantly. As a first-time developer, you're probably more focused on the technical aspects of your project, but don't forget to give enough attention to game audio. Don't treat it as an afterthought. Many indie developers make the mistake of using free sound effects and stock music. Don't be one of those developers. Such an approach will make your game feel amateur.

With my background in film composition, I had a good understanding of music and sound in general. While writing a score for my own game was very tempting, in the end, I didn't feel *DARQ* needed much music. Ultimately, I decided to do what was right for the game, not my ego. I ended up writing just one piece of music for the credit roll. I did, however, compose music for all of *DARQ* trailers, which allowed me to edit my footage exactly as I wanted. If I had used stock music, *DARQ* trailers would have turned out very differently.

As potentially self-serving as it might sound to say, it's probably impossible to overstate the creative potential of a composer in a game. These musicians are not just providing underscore but another voice to help weigh in on how the story is evolving, how the mechanics are tuned, or how the art direction is communicating, etc. For 'Journey' I got to be on the front lines of the development team for all three years. I offered my thoughts and let their ideas have an imprint on my music. The results wouldn't have been the same otherwise.

Austin Wintory | Grammy-nominated composer
Journey, Assassin's Creed Syndicate, Abzû

As a first-time developer, you have a lot to prove: both to your customers and to the press. Do everything you can to make your game feel as professional as possible. When it comes to music, consider collaborating with a composer. Unless you're an experienced composer yourself, you should delegate music composition. The role of a composer can extend far beyond writing music. A composer can become an equal part of the team who provides valuable feedback from an audio standpoint. Such feedback could be very useful and might influence your approach to game design, pacing, and storytelling.

While I didn't need much music for *DARQ*, I knew that sound design had an important role to play in my game. One day I got an email from Bjørn Jacobsen, a sound designer who at the time was working on *Cyberpunk 2077*. Interestingly, he emailed me just as I was about to sign a contract with someone else. He said he liked the *DARQ* trailer and was interested in working on the project. I shared with him my vision for the game, and it became clear that creatively, we were on the same page. Sound ended up playing a key role in *DARQ*'s success.

Similarly to other disciplines, sound design is a craft that takes years to master. When done badly, it can ruin the immersion and

make the entire game feel unprofessional. On the other hand, good sound design has the power to elevate the game to new heights.

Bjørn Jacobsen | Sound Designer
Cyberpunk 2077, Divinity: Fallen Heroes, DARQ

When you hire a composer or a sound designer, make sure your creative visions align. Highly creative people do their best work when given a lot of freedom, so avoid micromanaging them. Instead, encourage them to share their ideas and feedback. As audio professionals, they may offer a unique perspective that could improve your game.

The Process of Development

Once you've completed a few prototypes and determined that your concept is worth pursuing, it's time to begin developing your game. This step will take some time, especially since you're just starting to learn about game development. Things will get easier with time as you gain more experience. The tasks you decide to take on, with time, will become second nature. Be patient and enjoy the process as much as you can.

A lot of developers work crazy hours. So do I, most of the time. You already know that making games is a marathon, so you need to find a *sustainable* way of working hard and also taking care of yourself. It's important to stay in peak condition, both physically and mentally.

If you're stuck, don't hesitate to reach out to other developers and ask for help. Most of the time, you'll get a reply. Nobody is going to write a large piece of code for you, but if you can't figure something out or have trouble spotting a bug, reach out to somebody more experienced and ask for help. Indie developers make for a great and supportive community. Many will be glad to help you unless they happen to be in the middle of a pre-launch crunch. I often help developers who reach out to me. If you need coding advice, marketing guidance, or feedback on your store page, I'll be happy to help.

If you're the one who's coding the game, use placeholder art before worrying about custom-made assets. With placeholder art, you can get straight to work without wasting any time. That's a common practice in game development, and it will be especially useful to you since a part of the goal for early development is just learning and gaining experience. Polished game assets simply don't matter that much at this point. You can always swap placeholders with real art later, once you have a working game.

Vertical Slice

If you intend to get funding from publishers and investors relatively early in your development cycle, you should consider making a vertical slice of your game. A vertical slice is a small portion of the game that is as close to the final product as possible. In other words, it's a demo that is not available to the general public. While most of the game might remain unfinished, this one section should be as polished as the finished game would be. The vertical slice is meant to show off all game mechanics, and feature custom made art, sound effects, music, well written and recorded dialogs—whatever applies. It's OK to use placeholders for elements that are not ready yet, such as music, but make sure they are of high quality. Having a vertical slice is a powerful tool that can show off your game to business-oriented people who might not be familiar with the iterative nature of game development. Showing something that looks and feels great can help convince your potential investors and publishers to fund your project.

Having a polished vertical slice also allows you to start building your following early. You shouldn't post screenshots of your game while it's in the early prototype phase, but as soon as you have something that looks polished, you can begin building a community. With a vertical slice, you can generate good-looking screenshots, GIFs, and maybe even a teaser trailer. With a set of eye-catching marketing materials, you can start growing a following as your game is still in early development (more on that in Chapter 6).

Quality Assurance

Once your project starts to resemble a complete game, it’s time to put it through a lot of testing. Team up with a group of beta testers who specialize in finding and reporting bugs. If you don’t have a budget for that, at least ask your friends and family members to test it for you. You can also ask some of the most active and dedicated members of your community. Just be sure to reward them for their willingness to help. A copy of the final game and a mention in the credit roll is the least you can do to show your gratitude.

As a developer, you’re the worst candidate to test your game. As you continue to work on your game, you develop your own playstyle and a tendency to do things the same way each time you play. In other words, you’ve already seen all the outcomes that your playstyle produces. However, adding new people to the mix will help you discover lots of bugs because everybody plays differently. Your alpha and beta testers will interact with your game in the most unexpected ways. Watching them trying to break your game can be a stress-inducing experience, but it’s better to have your game compromised now than on launch day. Learning about newly discovered game-breaking bugs might give you the impression that your game is falling apart and is far less stable than you had imagined. Don’t worry—it’s normal. Just collect bug reports and continue to fix issues.

Apart from gameplay bugs, you might want to watch out for hardware-related issues. The game can run beautifully on your computer, but it might glitch or crash on older computers that have less memory or use older versions of the operating system. You need to test your game on various machines, both high-end, and low-end. If possible, try it on various CPUs and GPUs. It will make you aware of potential hardware incompatibilities before the launch. If you can’t fix hardware-related issues, at least make sure to indicate what your minimum system requirements are. Ideally, you want your game to run at sixty frames per second or more on a PC.

If your game offers controller support, make sure you test each controller thoroughly. For example, if you’re supporting both X-Box and PlayStation controllers, make sure to test older and newer versions, both wired and wireless. Check if your game responds accordingly when a controller gets plugged in or unplugged. A common practice is to have the game pause automatically when a controller gets disconnected.

One thing to keep in mind is that some players use ultrawide displays. You might have gotten used to testing your game on widescreen resolutions (16:9 aspect ratio). Still, a certain percentage of players may be disappointed when they discover that your game doesn't support ultrawide displays, which come in a variety of sizes (most commonly, 21:9 aspect ratio). If possible, add widescreen support; it shouldn't be a problem in first-person or third-person games, where the camera can rotate freely. However, not all games can support ultrawide resolutions for gameplay-related reasons. In some cases, the field of view in a side-scroller is essential to the game working properly. If that applies to your game, settle for a less desirable solution. Simply add black bars on the sides of the screen, so your game is still displayed correctly in the center. You can also make the bars thematically appropriate; they can display concept art or certain UI elements.

After the exhausting period of fixing bugs, eventually, you'll get to the point in which your game is mostly stable and ready to be released. Don't be surprised if some bugs sneak into your final build. It's virtually impossible to release a bug-free game. Your first customers will usually be your biggest fans, so they will be forgiving of bugs if they don't crash the game. Your ability to patch your game quickly will be appreciated.

Optimization

Both Unity and Unreal Engine provide profilers—handy tools that let you analyze the performance of your game and detect potential issues that result in framerate drops. Study this tool closely, because it can help you spot problems and make significant performance improvements. Some of the steps you can take to optimize your game include the following:

Reducing draw calls

A draw call is a request made by the engine to the graphics API to draw objects on the screen. The more draw calls your game is making, the more time it takes to render each frame. The more objects your scene has, the more draw calls the engine has to make to render each frame. Luckily, there's a way to reduce the number of draw calls being made in each frame.

It’s called batching.

Batching is simply combining objects into groups so multiple objects can be rendered with a single draw call. Your engine accomplishes this in two ways:

Dynamic batching
The engine will draw multiple objects with a single draw call if objects share the same material and textures. It’s in your best interest to limit the number of materials used in the scene. You should keep this in mind from the beginning of development.

Static batching
The engine will combine objects into a single mesh if they remain immovable during the gameplay. For static batching to work, you need to mark immovable objects as static. While static batching significantly decreases the time needed to render each frame, it comes at the cost of using more memory.

Reducing real-time lighting and shadows
Real-time lighting is heavy on the GPU and, in some cases, CPU. Reducing the amount of real-time light sources in your scene will naturally improve the performance of your game. If you need to use a lot of real-time light sources in your scene, think of a clever way to disable those that are invisible within the camera view.

Baking lightmaps
Replacing real-time lighting with baked lightmaps will save on GPU and CPU usage. However, since lightmaps are essentially textures, they will have to be stored in memory, so the boost in performance will come at a cost. Still, baked lighting is almost always preferable performance-wise, as long as your game allows for it. If your game features a lot of moving objects, baked lighting won’t be an option. If you can, however, you should use it as much as possible.

Reducing code that runs every frame
New developers tend to forget that code that runs every frame takes a lot of

processing power. There are cases in which it's perfectly justified, like listening for input. However, most tasks should not be performed every single frame. Think of ways you can prevent executing complicated calculations in every frame. Utilizing events and coroutines might be the solution you're looking for. Make sure to study those in-depth.

Implementing frustum culling

Frustum culling is the process of disabling objects that aren't visible within the view cone. Open-world games use it all the time. If your game is 3D and allows for free camera movement, using frustum culling can increase your framerate.

Using LODs

LOD stands for *level of detail*. It's often utilized in 3D games where you can view objects from various distances. Your engine already comes with LOD functionality. It allows for switching between meshes of different levels of detail, depending on the distance from the camera. All you need to do is provide meshes of various complexity. There are tools that generate LODs for meshes automatically. They usually work pretty well, although manual LOD creation is guaranteed to produce better results.

Optimizing raycasting

A raycast can be understood as an invisible laser—when pointed in a particular direction, it tells you if there's something in its way. If the laser ends up hitting an object, you get a lot of information about the hit. It can tell you how far the object is, at what angle it was hit, what the name of the object is, and much more useful information. Raycasting is often used to check if the character is grounded, if a bullet hits the enemy, if the player is near an obstacle, etc. Raycasting is a useful yet potentially performance-heavy operation. It can significantly decrease your game's performance if it's done wrong. Here are a few tips on how to optimize it:

Avoid using raycast every frame

If you have to check if the character is grounded, it may seem that shooting a raycast every frame is necessary, but doing so every second frame or every third frame will technically be just as accurate. Even more so, try raycasting in the physics update, which runs independently

from your frame rate. Your screen may be able only to display 60 frames per second, while your game might be running at 600 frames per second. By shooting a raycast each frame, you're basically wasting resources that could be spent on more important tasks.

Limit the length of your raycasts
Don't make your raycasts longer than they need to be. The longer they are, the more processing is required. For example, checking if your character is grounded might require shooting just a tiny raycast from the character's legs toward the ground.

Set a layer mask for your raycast
Consider what game objects should be excluded from the raycast. Coming back to our ground check example: make your raycast ignore all objects that are not on the "ground" layer. The fewer objects there are that a raycast can hit, the less performance-heavy it will be.

Utilizing Object Pooling
Object pooling refers to reusing preloaded game objects instead of creating and destroying them during gameplay. Instantiating and destroying game objects at runtime comes at a performance cost. Object pooling is a popular solution to this problem.

For example, if your game's character is an archer, you could preload ten arrow objects into memory before the game starts. Each time the player shoots an arrow, it will come from a pool of preloaded objects. Instead of spawning a new object when the pool runs out of arrows, the first arrow will be returned to the pool and reused during the next shot. This way, the pool will never run out of arrows, and the cycle will continue indefinitely.

To summarize, game optimization is an important topic, and it's something that you should research early on. While numerous optimization steps can be taken right before the launch, some of them should become a part of your pipeline as soon as you begin working on your game. For example, making your game objects share materials to allow for dynamic batching is not something you can do last minute. Also, keep in mind that

performance issues may not be noticeable on your computer, but they may become apparent on slower machines.

Last 10%

When your game begins to feel finished and stable, don’t rush to release it just yet. You might be eager to get it out there as soon as possible, but it’s still too early. Take some time to polish your creation as much as possible. Little improvements can make a big difference, both for your players as well as for the press. Too many games are released in the “minimum viable product” state. Adding extra polish and showing that you care is your chance to stand out.

Here are some ideas on how you can improve your game:

1) **Controls**
   Sluggish controls can feel extremely annoying to players. You might have gotten used to how your game responds to input, but consider making your controls feel tighter and more responsive. Also, if your game uses a keyboard, try implementing key remapping.

2) **Animation**
   Think about speeding up some of your animations. Increasing the speed of animations usually improves the overall feel of the game. It might not apply to every game genre, but if your game relies on timing in any way, you want to reduce the time it takes for the player to perform certain actions. For example, it's often a good idea to make your attack animations almost immediate. Pressing a button should result in an immediate response from the character. Immediate feedback feels responsive and satisfying.

3) **Camera**
   Whatever the genre is, the camera plays a significant role in making you feel a certain way. Incorporating camera shakes when appropriate can make the game feel more immersive. Also, adding water droplets during rain scenes, using dust overlays during impacts, and utilizing depth of field, motion blur, chromatic aberration, bloom, color correction, and other post-processing effects can make the camera feel more realistic.

4) **Pause Menu**
   You may assume your game is well configured, and nobody would like to change anything about it, but players expect to have access to a variety of settings in the pause menu. Fair warning: making a good pause menu with lots of settings takes a lot of time, but it's well worth it.

Consider implementing the following settings:

- **Graphics**
  Things like texture quality, shadows, anti-aliasing type, resolution, v-sync.
- **Audio**
  Master volume and separate faders for sound effects, music, and dialog.
- **Controls**
  Apart from remapping the keyboard keys, consider adding a mouse sensitivity slider and the ability to invert X and Y axes.
- **Gameplay**
  If applicable, add difficulty levels and various accessibility settings here.

5) Sound

Sound design is such an important part of the experience. If done right, it's rarely noticed. However, if your sound effects are inconsistent, poorly mixed, or too repetitive, they can quickly become annoying and ruin the player's immersion. Whether you're taking care of sound design by yourself or delegating it to a professional, now is the time to revisit the mix and to reexamine attenuation, reverb, and other parameters.

6) Details

You have a working game at this point, but what if you add things to it that are not essential? It would show that you care. Make a number of assets and place them in the world appropriately. Adding details that enhance the look of your scenes can improve the overall look of your game. Players might not notice all this extra work, but they will likely feel the difference.

Key Takeaways

Game design is the process in which both artistic and technical elements of the game come together. There are game design guidelines that you should be aware of. You don’t have to follow them, but know what they are before you decide to innovate.

The best way to learn game development is simply by doing it and having fun with it. There are plenty of free tutorials on any topic available online. The only missing component is hard work.

Iterative design is an effective way of making video games. It assumes the cyclic process of prototyping, testing, analyzing, and refining the project instead of trying to create a complete version of it from scratch.

Do your best to implement accessibility features. In most cases, it doesn’t take much time. There’s no excuse not to do it.

A vertical slice is a demo not available to the public. It’s used to showcase your game to investors and publishers.

Quality assurance and optimization are essential steps on the way to a successful release.

Once you have a finished game, take additional time to add details and extra polish. It can make a big difference.

• CHAPTER 5 •

Business and Law

While making a game is hard work, it is also fun. The business side is also hard work, but isn't much fun at all. That said, doing anything well commercially means you need to roll up your sleeves and get a little wise. Don't leave the details up to your lawyer or advisor. Take care and time to understand them yourself. Nobody is going to look out for your game's success or your new company as much as you. Making your first game is an accomplishment. Making your second depends entirely on how well you master the business and sales aspects of your first!

Mark Kern | @Grummz
Team lead for *World of Warcraft*, Producer of *Diablo II* and *Starcraft*.

Lacking a basic understanding of how to run business puts you at a severe disadvantage, which can potentially lead to terrible legal repercussions. Your goal should be to fill that gap in knowledge as soon as possible.

Please don’t mistake my words for legal advice. My company happens to be registered in the U.S. in the state of California. The information outlined in this chapter may not apply to you. Laws that impact the video game industry vary from state to state and from country to country. Therefore, my main advice is to do your own research and consult with an attorney. I hope that by going over some basic principles of business and law, you’ll be better equipped to research the relevant topics and ask the right questions when meeting with a professional.

Starting a Company

You may not need to register a company in the early stages of your gamedev journey. Before you consider forming a company, make sure that you treat game development very seriously and it’s not just a hobby. If that’s the case, there will come a point when it will be the right thing to do.

So, why would you want to register a company? Isn’t working as a sole proprietor just as good?

Here are the reasons why doing so might be a good idea.

Protect your personal assets

The main reason you want to register a company is to separate your personal assets from your business assets. Whatever company type you own, it offers you a degree of liability protection. For example, let’s imagine the following scenario: your game features jumpscares, and one of your customers with a pre-existing heart condition ends up having a heart attack while playing it. Your end-user license agreement fails to mention content warnings and is missing a limitation of liability clause. As a result, you get sued and lose. As a sole proprietor, all your savings

and possessions are on the line. You risk losing your personal assets, such as your car and your house. As the owner of an incorporated business, only your company's assets would be at risk. Your personal assets would stay protected. At least that would be the case if you run your business properly (more on that later).

Get a business bank account & credit card

If you own a company, you can open a business checking account and get a credit card for your business expenses. Your business account should be used for business purposes only: paying your contractors and employees, as well as receiving investments, funding, grants, and revenue from various retailers that sell your games. It's *essential* that you don't mix your personal and business expenses. Again, if you get sued, you don't want the court to discover that you've been operating your business as a sole proprietorship. Even though you represent a company, the court might still treat it as a sole proprietorship. Putting aside limited liability protection and holding the company's owner personally liable for the company's actions is known as "piercing the corporate veil."

Establish business credit history

By using your business card and paying your bills on time, you help establish your business credit history. It works similarly to your personal credit history, except that it affects the credibility of your company. Having a good business credit score can allow you to take out business loans at a bank at lower interest rates and give you leverage when negotiating with angel investors, venture capitalists, insurance companies, and potential business partners.

Prevent misunderstanding with co-founders

If you're not the sole owner of your studio, you want to avoid potential arguments with your co-founders and business partners. Don't enter partnerships that aren't clearly defined in writing. Verbal agreements might seem sufficient when a group of friends start working on a project, but that rarely works out well. If your game ends up becoming a hit, you don't want to get sued by your co-founders over various aspects of intellectual property or other details that were never defined in

writing. As an owner, solo or otherwise, you should prepare and sign an operating agreement. It’s an important document that outlines the internal operations of your business and establishes the relationships and roles of its members.

Get funding

If you're looking for equity investors, you want to show that your studio has liability protection. As a sole proprietor, you’re technically not investable. If you operate your business as a registered company, it’s considered a separate entity that can have multiple owners and shareholders. It opens up a doorway to getting more funding down the road, selling shares, going public, offering shares to your contractors instead of monetary compensation, etc.

Sell your business

As your business continues to grow, so does its value. While there are various ways of estimating what a business is worth, be aware that your revenue and ownership of intellectual property, such as copyright-protected assets and registered trademarks, add to the value of your business. One day, you may decide to sell your entire company. It’s not uncommon for the buyer to pay three or four times the annual revenue if your business has an established customer base, good cash flow, strong growth, and is well managed.

Add legitimacy to your projects

Consumers, publishers, investors, and the press will take you more seriously if you simply appear more professional. Adding *Inc* or *LLC* in your email signature (or whatever equivalents exist in your country) creates more trust. Signing emails with your name without specifying what company it’s associated with gives an unprofessional look when reaching out to important industry people.

Get more flexibility when paying taxes

As a company owner, you have much more flexibility when paying taxes. For example, as an LLC owner in the United States, you can elect your company to be taxed as a corporation, which can be beneficial under certain circumstances. In some cases, corporate tax rates would be

lower than individual tax rates.

Have more flexibility when hiring contractors

It won't be easy to gain trust from potential contractors as a sole proprietor, especially if you've never released a game before. Would you rather work for a company or a random person on the internet? If you can't afford to pay a contractor, consider asking them if they would be OK with working for you in exchange for a cut of your future revenue. As a company, you're likely to be seen as a legitimate and trustworthy organization. And hopefully, you are one. Never take advantage of your contractors; make sure you treat them fairly, so they love being a part of your project. By believing in your game early on, they become your first supporters. Your team is the most valuable asset of your business, so treat them with respect, and most importantly, pay your contractors on time.

Enjoy privacy

Separating your private address from your business address allows you to have a degree of privacy. If you ever have to deal with a backlash or online harassment, the last thing you want is to have angry customers showing up at your doorstep. As a company owner, you have the ability to hide your private information and list your business address instead.

Apart from the benefits of registering a company, there are also some disadvantages. Mostly, filing and recurring fees, as well as quite a lot of paperwork. You can hire an attorney to do most of the work for you—that would be the preferable way, although it would make the whole process rather expensive. Alternatively, you could use one of the web services that do all the paperwork for you. This may be an affordable alternative. Finally, you can study the subject and register a company entirely on your own. It's the cheapest way to get it done.

Since there are recurring costs involved in running a company, you don't want to incorporate too early. Here are some thoughts that might help you choose the right moment.

You've decided to make game development your profession
You're no longer testing the waters. You're now fully committed to finishing your game; you have a plan, and are in the process of executing it. Additionally, you have some good indicators that your game has the potential to become successful. Such indicators may include rapid growth in wishlists and social media following, frequent press coverage, offers from publishers, having your GIFs go viral, etc.

You're ready to hire contractors or employees
Your contractors will need to either sell you a license to use their work or transfer copyrights to you. In either case, you want those rights to be transferred to your company, not to you directly.

You're about to form a partnership in which you won't own the entirety of your game
If you're teaming up with your friends to make a game, you need to register a company as soon as possible. In the end, that's what a company is: a group of people working together towards one goal. Once your company is registered, you and your business partners should sign the operating agreement. The agreement will define equity ownership, management, voting rights, and other details.

You're about to receive funding from an investor or a publisher
If you're expecting a cash infusion, you don't want it anywhere near your personal bank account. Register a company so you can take advantage of the limited liability protection. Remember, you shouldn't mix your business assets with your private ones.

You have substantial business expenses
Start thinking about taxes as soon as you start working on your game. What can become a deductible? Many of your business expenses qualify as write-offs. Having a separate bank account strictly for business expenses will make it easier to track your business expenses.

Intellectual Property

Generally speaking, any creation of human intellect can be categorized as intellectual property (IP). By default, intellectual property is protected by law, preventing its unauthorized use by others. While intellectual property laws may vary from country to country, they remain pretty consistent throughout the U.S. and Europe. If you're located elsewhere, please double-check the accuracy of the information in this section.

In the video game industry, intellectual property is an umbrella term for three distinct categories:

Copyright

Essentially, copyright refers to the exclusive legal right given to a person to use their work or to authorize others to use it. There is no need to apply for copyright protection. The U.S. and Europe assign it by default once a work is created. So, what components of a video game qualify for copyright protection? Almost all of them: art, character design, animation, text, sound design, music, voice-over, and code.

Please note: since text is one of the protected categories, it should come as no surprise that even end-user license agreements and privacy policies are granted copyright protection. Don't ever consider copying any of those documents from other companies' websites.

What elements don't get copyright protection? Game ideas, game genres, and game mechanics.

Trademarks

A trademark is a sign, logo, symbol, word, or a series of words that distinguish your business or product from others. Trademarks play an important role in the video game industry, as they help protect both studio names and video game titles. Failing to trademark your game title could cause you much harm if your game becomes popular. Without a trademark, you won't be able to prevent other developers from selling clones of your game on other platforms using your title. Needless to say, it's a situation you want to avoid at all costs. You need to file for a

trademark for both your game title as well as your studio name. Before you settle for either, do extensive research. Check trademark databases before you file. Also, check if dot-com domains featuring your game title and studio name are available.

Filing for a trademark might seem like a straightforward process, but it's not. I had to find out the hard way. While you can do it all by yourself or use one of the web services that promise great results for low prices, you're probably going to get a rejection. The process itself can take from a couple of months to a half a year, so it's simply not worth it unless you really know what you're doing. It is one of those things for which you're going to want to use an attorney, preferably, a trademark attorney with experience in the video game industry. There are a few of them out there.

Patents

Patents are exclusive rights granted for an invention. It gives the patent owner the right to decide who gets to use his or her invention. In the video game industry, patents are rarely granted. It's hard to distinguish between a game mechanic, which is not protected by copyright law, and an actual invention. Some companies manage to patent their ideas, but it rarely happens. One famous example includes Sega, which successfully patented the floating green arrow in the game *Crazy Taxi* that was used to guide the player in the right direction.

Filing for patents is expensive, and the chance of getting them approved in the video game industry is low. As a first-time developer at the start of your career, you may find this type of intellectual property mostly irrelevant.

Contracts

As the owner of your newly established studio, sooner or later, you will want to hire contractors. So, what do you do when you find somebody you

want to work with? There are several ways to approach it:

Sign no contract at all
That’s the worst thing you can do. Not having a contract in place raises a lot of issues, most importantly, the question of ownership of the work delivered by the contractor. Once your game is out, don’t be surprised if you get sued for using assets that are technically owned by your contractor. It’s a recipe for disaster.

Use a template found on the internet
That’s also a terrible idea because chances are you won’t even fully understand the terms of the contract. Besides, the law in your country or state might contradict its terms. Such an agreement will not hold up in court should any party file a lawsuit. No template will ensure your collaboration goes according to your specific expectations. Every contract is unique and takes a lot of factors into consideration.

Hire an attorney
It will cost you, but you shouldn’t try to save money on this important step. Explain your specific situation to an attorney and have them write a custom agreement applicable in your case. Ask them to educate you about each paragraph, so you have a good understanding of how the contract works and how it protects you and your contractor if things go wrong.

Use the contract prepared by your attorney as a template
Chances are you can’t afford to pay for a professionally written contract every time you need one. If that's the case, ask your attorney to educate you on how to modify the contract so it can be reused for various contractors. Once you have a good understanding of how your contract works, you’ll be able to use it as a template for future collaborations.

When working with contractors, there will be two kinds of agreements that you will be using most often: work-for-hire agreements and non-disclosure agreement. Ideally, you want to have both prepared by your attorney. When it comes to work-for-hire agreements, you need to watch out for a few things:

Description of services
Make sure the agreement specifies what services are expected of the contractor. Be as detailed as possible because the agreement will be your go-to document should any dispute arise.

Relationship of parties
The agreement needs to specify that you're hiring a contractor, not an employee. The services performed by the contractor need to be defined as work for hire (contracted work). Clarify that you will not provide fringe benefits, including health insurance, paid vacation, or any other employee benefits.

Payment for services
Naturally, this is something you should negotiate with the contractor beforehand. Will it be an hourly rate or a flat fee? Will the payment be split into several installments? Will there be a prepayment? What method of payment will be used to transfer the funds? How will the payment be handled if either party terminates the agreement? What are the consequences of missing a deadline? There's a lot to consider here. Make sure the agreement takes all these factors into account and describes them in detail.

Copyright assignment (or licensing)
This one is very important. New works are automatically protected by copyright laws both in the U.S. and Europe. The creator of a work, in this case, a contractor, is the owner of their work by default. It means that if you hire a contractor to create something for you, technically, it remains their intellectual property whether you pay for it or not. Since your contractor's work is protected by copyright law, you have no right to use it in your game, unless you include the copyright assignment clause in your agreement. Copyright assignment is the transfer of the owner's rights to your company. The language required to make this hold up in court may vary from country to country, so make sure to bring this up when you meet with an attorney. While copyright assignment is the preferred way of using the contracted work, there's also an alternative available: licensing. In this scenario, the contractor

creates new work and grants you either a time-limited or perpetual license to use it in your game. Please note: going for a non-exclusive license will allow the contractor to reuse and resell their work to other developers or asset stores. Consider the implications of such an arrangement before you agree to it.

Copyright protection
It is important to have the contractor warrant that the work he or she is about to produce will be original and capable of copyright protection. In other words, this clause ensures that the work will not be plagiarized and will not violate or infringe upon any right owned by any other party.

Liability indemnification
This clause protects your company in the unlikely scenario in which your contractor ends up delivering a plagiarized work that infringes upon somebody else's intellectual property rights. Profiting from the work that you don't own puts you at risk of getting sued by the actual copyright owner. In this unfortunate scenario, it should be the contractor's responsibility to deal with any potential losses, costs, liabilities, claims, damages, or legal expenses.

Applicable law
Whether you're working with a local contractor or somebody located overseas, your agreement needs to specify what country's law comes into play, should one party sue the other.

Confidentiality
Work-for-hire agreements usually include a confidentiality clause. Its purpose is to protect confidential information that you will inevitably share with your contractors from getting out to the public. The confidentiality clause may be seen as a simplified version of a non-disclosure agreement.

When it comes to non-disclosure agreements (NDAs), those are usually used when you want to share a piece of confidential information with someone who is not a member of your team. Typically, an NDA defines you

as the disclosing party, and the person or a company who is receiving the confidential information as the receiving party. You might want to consider signing an NDA before you share an early build of your game with an investor or a publisher. It should also be signed before you share your game design document or other confidential information with your potential contractors or employees.

Here are a few things to watch out for when using non-disclosure agreements:

Confidential information definition
The receiving party cannot comply with the terms of the NDA unless you clearly define what constitutes the confidential information they're not allowed to disclose.

Exclusions
There are certain circumstances under which disclosing of confidential information should not be considered a breach. This section would define such circumstances.

Time period
This section should describe how long the agreement should remain in effect. NDAs can remain in effect for a certain period of time, until certain conditions are met, or in perpetuity.

Breach remedy
This section is meant to motivate the receiving party to protect the confidential information to the best of their ability. In the event of a breach, the receiving party should be responsible for damages resulting from unauthorized disclosure of the confidential information. It could be a commitment to cover court and attorney fees, loss of employment, or an injunction, which is a court order preventing the receiving party from continuing to disclose the confidential information to minimize the losses of the disclosing party.

Applicable law
Similarly to the work-for-hire agreement, NDAs also need to specify which country's laws regulate the terms of the agreement.

Work-for-hire and non-disclosure agreements should be written by your attorney and, ideally, should protect both your company and the other party. Apart from the contracts written by your attorney, you may come across other kinds of agreements that are offered to *you*, written by the legal teams of their respective businesses. Be aware that such contracts may prioritize protecting *their* businesses, not yours. Before you sign a contract with a publisher, an investor, a retail store, a distributor, a potential business partner, a marketing or PR agency, or any other business for that matter, consult with your attorney. A certain amount of negotiation might be required if your attorney determines that the terms of the agreement only protect the other party. Since attorney fees tend to be high, you might be tempted to skip on their services. Do *not* do that. If you're considering entering into a partnership with another business that has the potential to have a major impact on the future of your studio, having an experienced attorney on your side is a must. An experienced attorney can turn a good deal into a great deal. More importantly, they can prevent you from getting yourself trapped into a terrible deal that could potentially ruin your studio's future.

End-User License Agreement

Also known as EULA, this is a legal contract between your studio and your customer. Usually, a customer is required to agree to the terms of the end-user license agreement before he or she is allowed to play the game. EULAs tend to be quite detailed documents that specify the rights and restrictions applicable to the use of the game.

As you're finalizing your project and preparing for launch, an EULA should be at the top of your to-do list. If you release your game without having an EULA in place, you're taking a giant risk and exposing yourself to all kinds of lawsuits. If you're considering putting one together on your own simply by copying parts of EULAs from other games, please reconsider. As previously mentioned, EULAs, like any other written content, are protected under copyright law. If you're stealing intellectual property from other studios, you're simply playing with fire. There are various websites that allow you to generate an EULA based on data you provide, but such services are also costly, and for the most part, useless. Spend your money wisely and hire an attorney.

> *Legal help is necessary, but you don't need to seek out the biggest and most expensive law firms in order to get top quality advice. Instead, seek out one of the many attorneys who specialize nationally in video game law. You will likely pay half as much as you would at a big law firm, and you will get specialized advice from an expert in the intersection of games and law. It is also important to hire a lawyer as early in the process as you can. It is much cheaper to do it right the first time than to pay to correct mistakes later on.*
>
> **Stephen McArthur** | The Video Game Lawyer®

End-user license agreements are there to protect you and your business. Depending on the game you're making, your EULA will contain specific terms and conditions applicable to your game, and your game only. Again, this is one of those things that you don't want to approach as an amateur. Unless your game shows no signs of popularity and you don't expect it to sell more than a few copies, you should have an attorney write an EULA.

Privacy Policy

Similarly to EULA, a privacy policy is an essential legal document that you need to have before releasing your game. Ideally, you want to have it written as soon as you have a website and a newsletter. A privacy policy discloses the ways your company gathers, uses, discloses, and manages your customers' data. The document primarily applies to your game, but there might be other ways your company is collecting customers' data. As the concerns for privacy rights increase every year, the law becomes stricter and more unforgiving in this area. Lacking this important document can put your business at risk.

Writing this document is also one of those tasks that you want to delegate to an attorney. There's no template that you can use—privacy policy documents are custom-made for a reason. The way you collect, store, process, and share customers' data is specific to your particular company. The goal of having a privacy policy is to educate your customer as well as to protect your business from a potential lawsuit.

What if your game doesn't collect any data? If you're not doing it intentionally, your game engine might be doing it for you. Research this topic and try to find out if the engine you're using collects any data when a customer is running a compiled build of your game. For example, builds compiled in Unity Engine collect a certain amount of data about the player's hardware and the game's performance. The data collection is completely anonymous and complies with GDPR (general data protection regulation). Still, as a developer, you are responsible for disclosing what data gets collected and how it is used. Therefore, you need to do research and make sure you have a complete understanding of how your engine operates.

If you have a mailing list or offer customer support through email, your privacy policy needs to disclose how you handle your customers' data. It will also require some research on your part: you need to find out how the customers' data is handled by the third-party tools you may be using, like your newsletter service and email provider.

If your website uses any tracking tools, like Facebook Pixel, Google Analytics, or cookies, you will need to disclose that as well. Speaking of which, your website is a great place to host your privacy policy. You can link to it in your end-user license agreement.

Do your research and share what you've learned with your attorney: how your build might be collecting certain data, how the data is used, how your newsletter service handles user names and email addresses, etc. It will allow your attorney to create a comprehensive privacy policy that accurately reflects how your business handles customers' information.

Key Takeaways

As a new developer, you may not know much about the video game business. You need to educate yourself on this topic as soon as possible, because the lack of knowledge in this area might become detrimental to your success.

You may not need to register a company when you start working on your game. However, eventually it will be the right thing to do.

Intellectual property is an umbrella term for copyright, trademarks, and patents. While patents might not apply to your situation, you should educate yourself on copyright and trademark law. Be sure to consult a professional attorney when in doubt.

As a business owner, you need to understand how contracts work. The ones you will need frequently are work-for-hire and non-disclosure agreements.

Don’t forget to have your attorney draft an end-user license agreement and a privacy policy before you release your game.

• CHAPTER 6 •

Marketing and PR

As the market becomes more and more saturated, it's no longer enough to make a good game. You also have to be able to market it effectively. If you underestimate the importance of marketing and public relations (PR), your game will likely get lost among thousands of titles released every day. That's the primary reason most indie games fail. So, what exactly is marketing, and how does it differ from PR?

Generally speaking, marketing involves engaging in activities that seek to generate direct sales. The goal of marketing is to reach your potential customers who might be interested in your game and to convince them to take action, such as purchasing or wishlisting your title. Marketing campaigns have a short-term outlook and aim to achieve immediate results. Public relations refers to the image and reputation of your company.

The goal of PR is to maintain a positive perception of your studio by communicating your values to your audience. PR strategies involve ongoing efforts, not short-term activities. Therefore, they show results over a long period of time.

When working on *DARQ*, I prioritized PR over marketing. Whenever I had the chance, I would try to communicate my studio's core values: a customer-first approach, transparency, gratitude, fast customer support, immediate patching, direct and honest communication, and, most importantly, keeping promises given to customers. I did my best to communicate my core values through actions, not just words. Building a positive relationship with my customer base has been much more important to me than focusing on short-term marketing activities.

I choose to see marketing as day trading and PR as long-term investing. In order to launch your first game successfully and to continue to grow as a studio, you need to be engaged in both. Chances are, you won't be able to afford to work with marketing and PR agencies; therefore, learning how to do it yourself is one of the most important things you can do as a first-time developer.

Mission Statement

In essence, a mission statement is a summary of the goals and values that your company stands for. You can't fake a mission statement, and there's no need to—you don't have to reveal it every time you meet a potential customer. Defining one is important, though; it can inspire your team, help guide your company through hardships, and help specify your company's long-term PR strategy.

A good mission statement is short and concise. Try to write it in a few sentences, so that it can be easily memorized. Eventually, it will become your company's identity and will determine the marketing and PR choices you will be making. If your goals and values change over time, don't hesitate to revise your mission statement.

Website

Before you have enough marketing materials to create a good-looking Steam page, your website will serve as the main resource where your audience can learn more about your upcoming game. There are two ways you can approach this:

1) You can have a website for your studio, which will contain information about all your games.
2) You can have separate websites for each game and either completely skip making your studio website or postpone it until later.

Since you're a first-time developer, your studio name won't mean much to your potential customers. Nobody is going to search for it. Also, having a dedicated website for your studio might look a bit rash if all you have to show is one project. Your studio and its brand will become more relevant when you launch your game, but for now, spreading awareness about your game is more important. Once you have a portfolio of games, it will look great if you showcase them all in one place. For now, I'd recommend focusing on building a website for your first game.

Having said that, you should consider buying a domain name for your studio ahead of time. You want to avoid becoming a victim of cybersquatting. Cyber squatters are known for registering company names and brands as internet domains before a legitimate trademark owner gets to do it. Their main goal is monetizing search engine traffic or reselling the domain to the trademark owner at a premium price. Cybersquatting is illegal, but it's widely practiced regardless. After all, what are the chances you'd pursue a lawsuit instead of paying a premium price for a domain? The latter would be faster and cheaper.

Speaking of domain names, try to get one for your game as soon as you can. Just make sure your game title is not owned by another studio. Let your attorney conduct a search to determine that your trademark application has a high chance of getting approved. A good time to buy a domain for your game is right after your attorney files for a trademark.

In the early stages of development, all you need is a landing page with a call-to-action button. What should your call to action be? It's up to you to decide. If a visitor leaves without taking any action, they will probably never think about your game again. So you should aim to convert your visitors into newsletter subscribers or social media followers. At least that's what matters the most in the early days of your gamedev journey. Later, your priorities will shift, and all your marketing efforts will be focused on collecting Steam wishlists. When it comes to making a website, you have a number of options:

1) If you're an experienced web designer, make one yourself from scratch. You'll save quite a bit of money, but it will take a lot of time. I wouldn't advise doing that since your time is better spent developing the game.

2) If you don't know anything about making websites, you may want to commission someone to make it for you. It could be worth the money if you already have some impressive-looking assets, like a good-looking logo, eye-catching concept art, a trailer, etc. The problem with this approach is that your website will need to be updated regularly, so you will probably need to spend a lot of money over time having to rely on a contractor to do all that work for you.

3) If you're planning on having a lot of content on your website, such as regular updates or a devlog, you could try using a content managing system, like **WordPress**. It's free, but you'll likely have to pay for some decent template to make it look good. **WordPress** takes a while to set up, but once you figure it out, you'll be able to update your website by yourself.

4) If you know some very basic HTML, you can purchase an HTML website template. **ThemeForest.com** is one of the largest marketplaces you may want to explore. With basic HTML knowledge, you'll be able to easily edit the template by yourself and fill it out with content. HTML templates tend to be cheap, so it's a good way to save some money.

5) If you don't know HTML and lack general knowledge about how hosting and domain registration work, you could try services like **Squarespace.com** or **Wix.com.** These services allow you to get a working website up and running relatively fast without knowing anything about web design. You'll be able to update it by yourself and modify it as you see fit. They have a lot of templates to choose from and offer both domain registration and hosting services. The good thing about this approach is that the whole process is very user-friendly. The main disadvantage is that these services tend to be expensive, and the templates themselves look a bit generic.

Press kit

Journalists in the video game industry live busy lives. They receive hundreds of emails a day, so it's not easy to get their attention. If you're a first-time developer, it's going to be an uphill battle. If a journalist stumbles upon your email and it grabs their attention, they will likely proceed to look through your press kit to determine whether or not they're interested in covering your game. Therefore, creating a good press kit is one of the most important steps you can take if you want your game to be covered by the press.

A press kit is usually a single web page that contains information and marketing materials that a journalist might be looking for. It saves a journalist a lot of time to have all the information about your game in one place. You should link to your press kit from your website as well as have it in your email signature.

In a way, a press kit works like a business card for your game. It should be easily accessible by anyone who is looking for a concise representation of your project. Here's what a press kit should contain:

1) **Game title & description**
   This is your chance to grab someone's attention. Start with your game's title and keep the description short and engaging. One or two sentences is all you need.

2) **Fact sheet**
   This is usually a bullet point list that outlines the most important information about your game and your studio. It should be placed near the top of the page and contain the following information: your studio name and location, release date, platforms, price, languages, ESBR rating, your website and storefront links. Some of these might not apply to you at this stage.

3) **Banner image**
   Ideally, the banner image should be the artwork you use for your game's cover. Make sure it looks as professional as possible.

4) **Your studio**
   Include a short paragraph about your studio, its key members, etc. This

is also a good place to write about your mission statement.

5) Long description

Now that you've got your reader interested, you can afford to tell them about your game in more detail. Describe your game in a paragraph or two and provide any relevant information that makes your game newsworthy. For example, you could mention that your game won awards, features a well-known voice-over actor, went viral on twitter, or anything that could add legitimacy to your project.

6) Quotes

If your game was covered by the press, you could provide a list of quotes along with the article links.

7) Screenshots

Display 6-10 good looking screenshots as thumbnails and allow them to be viewed in full size. Choose the best-looking scenes from your game.

8) Logos, icons, and cover art

Similarly, display several variations of the logo, icons, and cover art. Make sure your logo is in PNG format and has a transparent background.

9) Thumbnail art

This is entirely optional, but if you add this, the YouTube community will love you for it. The idea is to provide several character renders on a transparent background. PNG files work great for this purpose. This allows YouTubers to create eye-catching thumbnails.

10) Download link

Provide a link that allows the press to download all your graphical assets in one archive. It makes their lives a lot easier. The archive should contain screenshots, logo, and cover art in high resolution. Preferably 4K or larger, on the off chance that your game makes it to the print media.

11) **Video**

This is the place to display your game trailers and teasers. Consider adding a download link for these too.

12) **Links**

Any relative links that help tell your story. These could be links to viral posts about your game, interviews, or public speaking engagements.

13) **Contact Info**

If a journalist is still here, they're definitely interested in your game. Just in case they want to interview you or gather more information for an article, provide email and social media links.

Look up press kits of your favorite indie games. You will see that most of them follow this structure.

Community Management

The importance of building an active community around your game cannot be overstated. People who become members of your community will motivate you to move forward, provide much-needed feedback, and give you the reason to persevere when things get challenging. They will also be the first to tell their friends about your game, retweet your content, pledge to your Kickstarter, and buy your game on day one. Therefore, honor and respect your community above all else. Be thankful and loyal to them. They're both the core of your business and your biggest supporters.

Building a community takes both skills and time. Some people have traits that make them naturally more effective at engaging with communities. You can choose to do it yourself or hire somebody to do it for you. Whatever you decide, make sure your community is being managed by somebody who displays the following traits and skills:

1) **Positivity and empathy**

Hopefully, your community will always be supportive and positive.

But what if something goes wrong? What if you need to delay the release of your game? What if your game ends up being buggy when it launches? Your community manager needs to have the ability to remain positive in every situation. Nothing good will come from reacting aggressively to negative comments. If there's much anger surrounding your game, there's probably a good reason for it. Listen to your community and consider engaging in an honest dialog. As an indie, you can skip the corporate tone and just be yourself. People will appreciate that. If you've done something wrong, apologize and think of ways to fix the problem as quickly as possible. Your ego has no place here—try to please your customers to the best of your ability. In essence, that's your only job.

I don't believe this trait can be faked. You're either a positive and compassionate individual, or you're not. You either respect your customers, or you don't. Putting a fake smile on your face when facing a backlash will only make things worse. If you're not naturally empathetic, find someone who is. Let them take over community management as you focus on other tasks.

2) **Communication and transparency**

A good community manager communicates effectively. Your community can smell a dishonest tone from a mile away. Just be open and honest about the state of your project. Transparency is a rare commodity in business. It's better to share bad news that will potentially disappoint your audience than to risk the long-term consequences of being caught in a lie. Trust is earned slowly, but can be lost instantly.

3) **Leadership and creativity**

One of the tasks of a community manager is making the community grow. Whether it's through organizing contests, scheduling weekly events, or constantly finding new ways of meaningful interaction, it requires both leadership skills and creativity.

Once you realize how much your studio depends on your community, you'll want to grow it as fast as possible. However, it's highly unrealistic to

expect explosive growth over a short period of time. Building a community usually ends up having a snowball effect. It's very hard to get ten followers when you're just starting out. It can be discouraging at first, but you should stay patient. As you continue to be active on your social media channels and try to engage with the community, your following will grow exponentially. This process takes years, so have realistic expectations. For example, in 2018, my Instagram account had 1,000 followers. It took a lot of work to get to that number. However, two years later, it got to 11,000. It would have been impossible to predict such growth at the time.

When it comes to platforms, it's up to you how you want to approach this. As of writing this book, Discord is the most popular tool for community management. Having the ability to chat with your audience in real-time feels very engaging and interactive. Also, it's great to have your fans chat with each other while you're offline. Twitter is a fantastic communication tool as well, although the communication here is more one-sided and doesn't feel as personal.

Social Media Guidelines

Minecraft is the best-selling video game of all time. It was developed mostly by a solo developer, Markus Persson, also known as Notch. As far as I'm aware, Markus didn't spend a dime on ads. Instead, he used YouTube, one of the most popular social media platforms, to reach his target audience organically. Thanks to the YouTube community, *Minecraft* videos generated over 100 billion views as of 2020. No triple-A studio has been able to replicate this level of success, no matter how big their marketing budgets have been.

The beauty of social media is that it gives all of us an equal opportunity to succeed. Is it hard? Yes. But it's possible. Therefore you need to make every effort to become adept at using social media.

There are lots of platforms to choose from: YouTube, Twitter, Facebook, Instagram, TikTok, LinkedIn, Imgur, Reddit, Tumblr, Snapchat, Pinterest, and many more. They're all different, so posting the same content on every platform simply won't work. As a developer, you can't possibly have enough time to use them all. I suggest picking one that will become your

major platform in terms of growing your following. Also, add one or two more platforms that would be treated as secondary channels of communication. Here's what you should consider when making your pick:

Which platforms are used by your target audience?
Knowing where your target audience hangs out should be the main determining factor. After all, you're a business, and your use of the platform comes with a goal in mind: growing your community, increasing your brand awareness, and generating sales. It doesn't mean you can't make genuine human connections while pursuing those goals. In fact, you should. But it's important that you're a part of a social network that finds your content interesting.

Which platforms do *you* like?
Many businesses misunderstand how social media platforms work. I keep seeing companies that post links to their products and say: "Buy this now!" It comes off as spammy salesmanship and has little chance to generate much attention. What really matters is that you engage with your community and interact with real people. You can't fake it—you have to actually enjoy it. Therefore, it's important to use platforms that you like so your interactions are genuine.

Which platforms give you a high chance of going viral?
Consider this as a factor, since one of the advantages of using social media is getting a shot at massive exposure. It's impossible to plan for going viral, but when it happens, it feels great. If one of your posts ends up hitting it big, you can potentially get millions of eyes looking at your game, completely for free. Not all platforms are equal when it comes to organic reach. Many platforms limit organic reach to force you to buy ads. Thankfully, there are still some that give you an opportunity to go viral: YouTube, Twitter, Reddit, Imgur, Instagram, and TikTok are among those platforms.

In my case, Twitter has become the main platform I interact with daily. I didn't like it at first, but I started enjoying it more and more after I saw its potential. Going viral on Twitter is incredibly exciting. It's a fun and interactive experience that can last multiple days. Discord and Instagram

are my secondary platforms. I use them once in a while to make important announcements, as well as to answer comments and direct messages.

I've had a fair amount of success on Reddit as well, although this isn't a platform that generally welcomes self-promotion. If you're not an active Redditor who adds value to the community, your marketing attempts will fail. I happen to like Reddit a lot from the consumer's point of view and use it primarily as a means to contribute to the gamedev community. I also follow various subreddits to keep an eye on industry news. Posting about your game once in a while is acceptable, but overdoing it will do you more harm than good.

All my social media accounts have the same username: *UnfoldGames*—the name of my studio. Similarly to the website dilemma, you should decide whether to make separate accounts for all your games or post about your games from your studio account. In my opinion, it doesn't make much sense to split your following into separate accounts dedicated to individual games. Triple-A studios do that, but they deal with very different marketing budgets than indie developers. Social media following tends to grow slowly over time, at least in most cases. The more people follow you and know who you are, the easier it will be for your game to become successful. When you make more games, you won't have to build a new community from scratch. You'll get to market your new products to your pre-existing customer base. That's the main advantage of running your social media accounts as a studio, not as a fan page for a specific title.

While each social media platform has its own set of rules and best practices, I'd like to give you 10 general guidelines that will help you get started:

Know your first follower

As you start a new social media account, your first "follower" will be there waiting for you. It's your *reputation*. It will literally *follow* you everywhere you go, for years to come. Respect your audience. Never lie. Treat them fairly. Offer them good deals. Do your best to be useful and provide value.

Start early

Growing your following will take a long time, so it's a good idea to start as early as possible. You don't have to post much content if you don't

have anything to show. However, even at this early stage, there is a lot you can do. You can follow other people, get to know them, engage with their content, get their attention, etc.

Provide value

By now, you should have a good understanding of the demographics associated with your chosen platforms. What kind of content do they like to consume? What do they appreciate the most? Keep in mind that if you turn your account into an advertising machine spewing calls to action, nobody will pay attention. You need to constantly think of ways to enrich other people's lives. Why else would anybody want to become your follower?

Build real relationships

Social media platforms assume that you are a *social* individual. It's hard to grow your following if you don't engage with other users. Unless you're a celebrity or a big brand, you can't afford one-sided communication. In other words, you can't limit your interaction with your audience to simply posting content. You also have to answer comments and direct messages. Apart from *reacting*, you should also be *proactive*. Try to engage with people you find interesting and interact with their content. Again, don't try to fake interest in what others are posting. There are plenty of like-minded people out there who could become your real friends, or potentially, business partners.

Be consistent

Keep in mind that your followers also follow lots of other accounts, so if you disappear for a while, you'll be forgotten. The only way you can stay relevant is to keep posting content regularly. Each platform has its own guidelines when it comes to acceptable posting frequency. Make sure you stay consistent in the way that aligns with the platform's best practices. For example, while posting multiple times a day on Twitter is perfectly acceptable, it would look spammy on other platforms, like YouTube. Whatever the acceptable norm is, don't disappear for weeks. People need to remember who you are and expect to see your posts regularly.

Don’t change your profile picture
Most social media platforms are crowded places. Once you gain some traction, people will associate your account with your profile picture. If you change it unexpectedly, many will be confused as to who you are and why they follow you. Try to keep your profile picture unchanged for as long as possible.

Post high-quality content
It’s nearly impossible to come up with high-quality content on a daily basis. Luckily, you don’t have to post original content all the time. Many social media platforms allow you to share other people’s content. Not only will it make you look good to share something amazing, but it will also be appreciated by the original poster. If you do that often enough, they may even return the favor and repost some of your content. When it comes to posting original content, make sure yours is of high quality. In other words:

Your content needs to bring people value.
Value doesn’t have to mean useful information. Some posts bring entertainment value, while others simply serve as an enjoyable distraction. Whatever you post, it needs to be perceived as valuable.

Your content needs to be engaging.
Your posts have to encourage people to interact with them: click, read, comment, like, share, etc.

Collaborate with other developers
You can create a partnership with other developers and commit to supporting each other in pursuit of growing your audiences. You can exchange shoutouts, engage in collaborations, post shares, or retweets—whatever applies to your respective platform.

Be creative
Your followers are scrolling through their feeds in search of exciting content. It’s your job to create that excitement. Think of ways you can attract your followers’ attention. Consider organizing contests, mini-

games, giveaways, etc. As a developer, you can ask your followers to help you decide on your character's name, help you choose between two versions of your game's logo, etc. Your fan base will appreciate you for treating them with respect and asking for their feedback.

Choose video over text

Whatever platform you choose, visual content is much more engaging than text. Try posting video content whenever possible. It's the best way to showcase your game anyway. We've already spoken about the importance of GIF's. They are especially effective on Twitter. If you can't post videos, try to post eye-catching images. Text posts can be engaging too, but it's easier to grab attention with visuals.

Social media algorithms, as well as the popularity of various platforms, change frequently. Chances are that social media platforms will evolve significantly by the time you finish your game. What works now won't work in a year or two. Therefore, I want you to stay alert when it comes to algorithm changes within the platforms that are popular now, as well as keep an eye on new platforms that have the potential to become disruptive. Joining those early, while the competition is low, can give you a massive advantage. Having said that, I'd like to give you a few words of advice regarding the three major platforms: Twitter, Facebook, and Instagram. The advice is tried and true as of writing this book. I hope it will stay relevant for a long time.

1) **Twitter**

Twitter is very popular among the gamedev community, gaming press, content creators, and streamers. I don't see a reason why you would want to stay away from it. I think it would be accurate to say that at least half of *DARQ's* press coverage came from journalists discovering the game on Twitter.

When starting a new account, use hashtags to attract attention. Popular hashtags for an indie studio are **#GameDev** and **#IndieDev**. You can also use engine-specific hashtags: **#MadeWithUnity** or **#MadeWithUnreal**. The official accounts of Unity and Unreal Engine

might retweet your GIFs or screenshots, which usually results in a noticeable exposure boost. There are also day-specific hashtags that are very popular: **#ScreenshotSaturday** has a big audience from Friday night until Saturday night. This is when a lot of industry people are looking for new interesting games. Participating in the Screenshot Saturday showcase can help you get the attention of journalists, publishers, investors, and other industry professionals. Use this hashtag to showcase your work in progress. Another popular one is **#IndieDevHour**—it's active for one hour starting on Wednesday at 7 pm UK time.

Once your account grows, you may want to stop using hashtags and rely on retweets to reach a broader audience. Using a lot of hashtags may look a bit spammy if you have an established account.

Don't use Twitter ads. They're overpriced and don't produce good results. I've yet to meet a developer who has anything good to say about them.

Don't follow too many people. If you follow more people than follow you, your account might come off as spammy. Don't follow people just so you can unfollow them as soon as they follow you back. Besides, the followers you acquire this way don't really care about your game anyway. It's a waste of time. Also, don't even think about buying followers. People will see right through this lie when they notice no engagement on your posts. You're not fooling anyone with fake numbers. Building your brand on an obvious lie is a terrible idea. Instead, use every opportunity to show people that your brand is trustworthy. Even if your following is small, be proud of it. Your reputation will follow you everywhere, so always prioritize honesty and transparency when interacting with your community.

2) Facebook

Just a few years ago, Facebook Pages were an effective way to reach your target audience organically. Sadly, in 2018, Facebook made significant changes to its algorithms. Since then, organic reach is almost non-existent, so technically, it doesn't matter how many likes

your page has. You won't be able to reach the majority of your audience organically anymore.

On the other hand, Facebook remains the best platform when it comes to highly targeted ads. You can reach people who are your target audience with a few clicks of a button—that's an amazing opportunity. The ads are reasonably priced, so if you have a marketing budget, you should consider using them.
Additionally, you should join a number of Facebook groups. ***Indie Game Developers*** group is a general community of indies just like you. While developers might not be your target audience, it's motivating to be a part of an active and supportive community like this. You're free to showcase your game here, ask questions, and receive feedback. If you're looking for contractors, a good group to join is ***Gaming Industry Talent***. You can find all kinds of professionals here who would be happy to work for you. There are also software-specific groups, like ***Blender*** or ***Pixologic ZBrush***.

Finally, consider creating a group of your own. Make it private and use it for communication with the members of your team. That's what I did when I started working on *DARQ*, and it proved to be an effective way of keeping everyone updated on the development progress.

3) Instagram

The algorithm of the feed is constantly changing, which often affects the visibility of your posts. As of writing this book, Instagram feed remains sorted non-chronologically. The posts that make it to the top of the feed are from people you interact with the most. In other words, if you want to increase the organic reach of your posts, you should become an active member of the community.

The key to making your posts do well on Instagram is to use relevant hashtags. Similarly to Twitter, hashtags allow your post to reach a wide audience—people who are not following you. If your post receives a lot of likes and comments, it will climb the ranks within a given hashtag. Depending on the competition within a given hashtag, your post can make it to the top of the list. This is a great opportunity to

grow your following organically.

When I started my account, it was relatively easy to reach some level of virality. Many of my early posts had over one hundred thousand views, while my number of followers was around 1,000. It became more difficult as the algorithm kept changing, but I believe it's still possible. I see Instagram posts go viral all the time.

Currently, Instagram stories seem to generate more engagement than posts themselves. Take advantage of this and post stories as often as possible. This way, you can remain in the spotlight on a daily basis. If you manage to reach ten thousand followers, you'll be able to add links to your stories. That's when the stories can become an effective marketing tool. You can send traffic from your stories straight to your storefront, official website, other social media accounts, etc. Be careful, though: if you overdo it, your followers will see it as spam.

If you want to show up in front of other stories, consider going live. You can livestream your development process, run Q&As, or showcase your daily life so your followers can get to know you better.

Marketing Materials

As you develop your game, you should continue to expand your arsenal of marketing materials. In essence, they are items designed to show off your game to potential customers, publishers, investors, journalists, and content creators. Ideally, you want your marketing materials to generate excitement, curiosity, and word of mouth. If people feel the need to tell their friends about your game after they watch your trailer, you've done your job right.

There are several kinds of marketing materials you'll need:

1) **An elevator pitch**

   This is the most important piece of text you will ever write about

your game. An elevator pitch is a brief description of your game that is designed to spark interest. It should be as brief as possible, lasting no longer than an elevator ride—20 seconds or less. Ideally, you want to be able to describe your game in one or two sentences. Make every word count. Writing a good elevator pitch can take some time, but it's worth the effort.

Why do you need an elevator pitch in the first place? It's only a matter of time before somebody asks you to describe your game. If you ever decide to showcase your game at a convention, you'll be asked to describe your game a thousand times. You'll need to have your answer loaded and ready whenever somebody says: "So, tell me about your game." You'll have to talk about your game to the general public, industry professionals, publishers, investors, and journalists. If they like your elevator pitch, they might take further steps that may lead to press coverage, business opportunities, funding offers, or publishing deals. If you plan to present your game in public, make sure to rehearse your elevator pitch. It should be deeply ingrained in your mind by the time you need to use it.

Apart from public events, you'll be asked to describe your game on social media or via chat on multiple occasions. Also, storefronts like Steam and GOG will ask you to provide a short description of your game. There will be plenty of opportunities to use your elevator pitch, so figure it out as soon as possible.

Here are some things to consider when writing an elevator pitch:

- Avoid using video game slang and technical language so a business person can understand what you're saying.
- Avoid referencing other games, on the off chance that your listener is either unfamiliar with it or dislikes it for whatever reason.
- State your game's genre and explain what makes it stand out from other titles in that category.

- Use attention-grabbing words and try to appear confident when talking about your game.

2) **Trailers**

It should come as no surprise that trailers are among the most important marketing assets your game will ever have. Trailer releases are considered newsworthy events, so you should try to get some press coverage when you release yours. A good time to release an announcement trailer is when you have a vertical slice of the game ready. Your announcement trailer should look as polished as possible and be under two minutes in length.

Following the announcement trailer, you should release a few teaser trailers. These types of trailers are shorter, typically one minute long or shorter. They're used to heighten the excitement about your game while it's still in development. These should be relatively easy to make, given that by that time, you'll have plenty of interesting footage to show.

And finally, when you're ready to release your game, you should create the launch trailer. It will become your main marketing asset before the release. This trailer will also be used on your storefront pages, so make it look as good as possible. Most customers never bother to read the game's description or look through the screenshots. They mostly make a purchase decision based on the trailer. If a customer doesn't like the trailer, they won't spend time reading reviews and will simply move on to another game. That's why having an attention-grabbing trailer is essential, both before and after your game's launch.

If you don't feel comfortable with the idea of learning video editing overnight, consider hiring a video editor. Learning video editing is not as difficult as it may seem, so if you don't have the budget, consider doing it yourself. All *DARQ* trailers were edited by me, without any prior editing experience. It just took a few days to learn the editing software. **Adobe Premiere Pro** was quite intuitive, and I managed to achieve decent results with a fair amount of hard work.

If you're a Mac user, you will probably need **Final Cut Pro**. When it comes to screen capturing, I recommend **OBS**—it works great, and it's free.

Whether you're going to be editing the trailer by yourself or not, here are some tips to help you achieve good results:

- **Avoid non-gameplay footage**
  Unless your game is a highly anticipated triple-A sequel, you can't afford to have a cinematic trailer that doesn't show gameplay. Your potential customers want to see gameplay, so show it immediately. Don't have them wait for it, because they won't. From the time they hit the play button, you have 3 seconds to capture their attention. I suggest that you start your trailer with "the hook" of the game—your core mechanic.

- **Don't include your company branding**
  If you're a first-time developer, nobody cares about your studio logo. Don't try to copy the format of triple-A trailers. They put their logo in front of trailers because it brings a lot of value. Your logo brings no value at this point and only wastes the precious seconds of the trailer's opening. If you have to include your branding, do it at the end of the trailer, not at the beginning.

- **End the trailer with a call to action**
  Depending on the kind of trailer you're making, your call to action could be: "Wishlist on Steam," "Follow on Twitter," or "Get it Now." Not many people will watch your trailer all the way to the end. Those who will are obviously interested in your game, so tell them what they should do next to support your project.

- **Capture gameplay footage with sound but no music**
  When you're ready to screen-capture the footage for your trailer,

don’t forget to mute the music in your game.

- **Commission a composer to write music**
  Having custom music written especially for your trailer can make a big difference. Ask a composer to write a trailer track and use it when editing. Try to line up your video clips with the rhythm of the music.

- **Show variety**
  Show off contrasting scenes, so the viewers can see that your game has a lot to offer.

- **Don’t include text**
  If you have to include text, at least be aware of the consequences. A big portion of the PC market doesn’t speak English. You don’t want to alienate a large portion of your audience. In most cases, text is unnecessary. Try to *show* your game’s features instead of writing or talking about them.

- **Create an end card**
  Your trailer’s end card should contain your game’s title and cover art, as well as platforms, release date, social media handles, and official website.

- **If you’re uploading to YouTube, export in 4K**
  Even if your footage was captured in 2K, try exporting the trailer in 4K. This will stretch your footage, but it won’t be noticeable when you upload it on YouTube. The advantage of uploading in 4K is significant: the video and the audio won’t get compressed as much by YouTube compression algorithms.

3) **GIFs**

While trailers take a lot of work and effort to produce, animated GIFs are relatively easy to make. You only need to showcase a couple of seconds of gameplay in a GIF. This allows you to post them on social media quite regularly, which will keep your audience engaged while you’re busy working on the game. GIFs do

particularly well on Twitter. If you get lucky, your tweet will go viral and give you an incredible amount of exposure.

4) **Screenshots**

While screenshots are not as engaging as trailers or GIFs, you should aim to have at least six or more high-quality screenshots. Your website and press kit should list them all. You'll also need them for your store submissions.

Here are a few tips that will help you take better screenshots:

- Take a large number of screenshots, so you have a lot to choose from.
- Try to show a lot of variety in your screenshots: different characters, contrasting environments, etc.
- If your game features a lot of user interface text that is not essential to understanding your gameplay, consider hiding it to achieve clean looking images.
- If your game relies on user interface elements that are essential to understanding your gameplay, you should keep them in your screenshots.
- Consider writing a camera script that would allow you to move it freely in your game. This way, you'll have much more control when it comes to angles and composition.
- If your game is fast-paced, consider writing a script that would allow you to control the game's speed. Slowing down or pausing your game will enable you to capture great scenes at the right time.

Press Coverage

When you're just starting out as a game developer, it's not easy to get your game noticed by the press. This is because there's a conflict of interest

at play. You're looking to have your game covered so it can grow in popularity, while journalists are looking to write about games that are *already* popular. If that feels unfair to you, please reconsider. The business model of media outlets is largely based on advertisement. Would an article about your game generate as many clicks as a news story about one of the most anticipated triple-A titles?

Having said that, you shouldn't give up on pursuing press coverage for your game. I've been fortunate to have *DARQ* featured in hundreds of media outlets, including *IGN*, *Kotaku*, *PC Gamer, GameSpot,* and *Forbes*. It's possible for a first-time indie, and I'm proof of that. I'm going to share with you what I've learned over the years:

1) **Timing is everything**

   In every game's development cycle, there are several milestones that are considered newsworthy. You should contact the press when there's something important to write about, so timing is very important. Here are some of the most important milestones that are likely to attract the interest of the press.

   - **Game announcement**
     This is one of the biggest newsworthy events your game will ever have. Ideally, you want to have the announcement trailer ready by the time you contact the press.

   - **Demo release**
     If you're planning on releasing a demo, consider notifying the press before that happens.

   - **New trailer**
     Releasing a new trailer along with an update on the development progress is an opportunity for more coverage and possibly interviews.

   - **Early access release**
     If you're planning to go through Early Access on Steam, this is a newsworthy announcement. The press should cover your release date, and possibly review your game at this point.

- **Launch**
  The biggest news of all, deserving of a detailed review of your game.

2) Have realistic expectations

I want you to know what to expect when it comes to the impact press coverage will have on your game's popularity. Since your game will not be widely known at first, press coverage might not immediately result in the increased popularity of your title. Articles that feature indie games from unknown studios don't get that much traffic, so you're likely not going to get as much exposure as you're hoping for. Regardless, getting your game covered in major media outlets has a number of advantages that you should consider:

- **Legitimacy**
  As a new developer, you need to convince your audience that your game is not an amateur project that will never be released. Getting your game featured in articles across the web adds legitimacy, both to your game and your studio.

- **Quotes**
  Complimentary excerpts from articles can be used for marketing purposes. They can be shared on your store pages, social media platforms, as well as in your press kit. In a way, positive quotes serve as a stamp of approval for your game.

- **Video playthroughs**
  Content creators look through press headlines regularly to keep up with current trends. If your game grabs their attention, they might request review keys and showcase it on their channels.

- **Metacritic score**
  Once you're close to the launch, you should consider the

importance of the Metacritic score. If you're not familiar with **Metacritic.com,** it's a website that aggregates review scores for video games and other media. While most of your target audience will probably skip on reading reviews of your game, they might check out your Metacritic score. In other words, the more media outlets review your game, the more accurate your Metacritic score will be. We'll speak more about this in Chapter 8.

3) **Follow emailing guidelines**

If you can't afford to hire a marketing and PR agency, you will have to do a lot of emailing by yourself. Here's how to approach it:

- **Build a list of contacts**
  Browse through the media outlets you're targeting and read the most recent news and reviews featuring games in your genre. Take note of who the author is. Try to find their email address. Often, it will be listed on the site somewhere. In other words, try to compile a contact list of journalists who are interested in covering your genre.

- **Choose the right time**
  Journalists receive a lot of emails, so it's better to contact them on less busy days. Fridays are known to be the least productive days in every office environment. Some editors might finish early, while others might be reluctant to start on a new writing task as the countdown to the weekend begins. Mondays are probably the worst days to contact the press, as they're dealing with the backlog of emails sent to them over the weekend. That leaves Tuesday, Wednesday, and Thursday. In addition, avoid contacting journalists during big industry events, like E3, PAX, GDC, and other conventions. Chances are, they're not at the office and don't have time to read through their emails.

- **Personalize your pitch**
  I know it takes a lot of time, but it's simply a way to show your respect. How would you feel if you kept receiving hundreds of emails a day that start with a generic greeting without even mentioning your name? It comes off as spam. At the very least, address your recipient by name.

- **Keep it short**
  When writing your email, keep it as short as possible. Nobody has time to read through long paragraphs. Providing a minimum of information, like your game title, elevator pitch, an attention-grabbing screenshot, or a GIF, is all you need. Be sure to include a link to your press kit, so a journalist can easily access more information about your game.

- **Don't expect a reply**
  Don't get discouraged if you don't hear back. Journalists usually don't have time to reply, even if they are in the process of writing an article about your game.

- **Set up Google alerts**
  Using Google alerts allows you to monitor the results of your marketing efforts. If the press covers your game, you will receive an email notification. It's a good way to track the effectiveness of your email pitches.

- **Follow up**
  On the off chance that your email got lost among hundreds of others, consider following up in a few days. Be polite and don't follow up more than once—you don't want to get your email flagged as spam.

YouTubers and Streamers

These are the taste-makers of the industry. As an indie developer, you can benefit greatly from collaborating with content creators and streamers. After all, that's how *Minecraft* became the most popular game in the world. A part of your marketing campaign should be focused on sending your game to content creators and streamers. Be selective about it and make sure the person you're emailing has a substantial viewership and covers games in your genre. Try to write personalized emails.

Additionally, you can increase your reach by using the key distribution services like **KeyMailer** or **Woovit**. They allow content creators and streamers to request keys for your game ahead of its release. Eventually, you'll be able to send hundreds of keys with a press of a button. The good thing about these services is that all participating YouTubers and streamers are verified—they are the legitimate owners of their respective channels. However, not all of them have good intentions. Some of them simply collect free keys without ever planning to cover your game. The keys are later resold at a discount on various websites. Key distribution services usually require a subscription fee to unlock features that can help you separate the legitimate requests from the fake ones. Still, it might be worth it.

Key Takeaways

Marketing campaigns aim to reach your target audience and generate sales in the short-term. The goal of PR is to maintain a positive perception of your studio over the long-term. In order to succeed as an indie developer, you need to master both.

You need to create a set of marketing materials to promote your game. Apart from trailers and screenshots, you will need a good elevator pitch, a website, and a press kit.

Your success as an indie developer is mostly dependent on your ability to grow a community around your game. If your skillset makes it hard for you to be a good community manager, consider delegating it to someone else.

Learn how to pitch your game to the press. Timing is everything, so contact journalists when you have a newsworthy story, like the reveal of your announcement trailer, or a demo release.

Depending on your game's genre, YouTubers and streamers can play a major role in your game's success. After all, that's how *Minecraft* became the most popular game in the world.

• CHAPTER 7 •

Publishing and Distribution

You may already know that I chose to self-publish *DARQ* on Steam and GOG. In my case, it turned out to be the right thing to do. I'm not advising you to follow in my footsteps though, as your circumstances might be completely different. Signing with a publisher could potentially solve a lot of your problems and ultimately become your path to success.

First-time developers often have impostor syndrome. Lack of experience combined with the fear of failure often results in rushing to sign publishing deals without considering all the pros and cons. Understandably, the idea of having an experienced business partner come in and take all the risk while you're given money to focus on making your game sounds like every developer's dream. And it is! Depending on your particular circumstances, and most importantly, the terms of the deal, it may or may not be worth it.

Publishing vs. Self-Publishing

We've already briefly discussed what you may be asked to give up in exchange for publishing services. Depending on your particular situation, it

could be:

1) **A percentage of future revenue**
   In most cases, publishers will offer you a cut of the net revenue, not the gross revenue. To clarify, net revenue is your game's sales, minus the cut taken by the storefronts (which is usually 30%), minus refunds and chargebacks, minus the expenses incurred by your publisher in regard to your game. Publishing expenses might include funding, advances, and other services that you may require, such as marketing, localization, quality assurance, etc. Since you can't control your publisher's expenses, agreeing to a cut of the net revenue could end badly for you. Depending on the specifics of the agreement, it could be something you may want to avoid . Consult an attorney when in doubt.

2) **Intellectual property ownership**
   You might be asked to give up the ownership of your IP. In most cases, it's a red flag. This changes the dynamics of your partnership and essentially turns you into a contractor—well, kind of. Now that you have milestone deadlines and other contractual obligations, how else would you define your role on a project that is no longer yours? Think twice before giving up the ownership of your IP. Don't do it, unless you're offered a big pile of money.

To be fair, not all publishers will offer you such unfavorable terms. There are genuinely good companies out there that are run by good people. I'm mentioning these terms because that's what I was offered in the vast majority of cases. I also had the pleasure of meeting publishers who were fair, transparent, and passionate about games. Scott Millard, the managing director of *Feardemic* was one of them. That's why I asked him to be a part of this book.

> *When looking for a partner in a publisher it's important to evaluate how your game compliments their current portfolio and how their current portfolio compliments your game. Try not to look at it from the perspective of competition. Instead, try*

and look through the lens of the consumer. Many gamers buy games that are similar or are from the same genre. If your game is a strategy game, find a publisher who has had success with a similar type of game.

Scott Millard | Managing Director at *Feardemic*

Let's go through some of the services that a publisher can provide. We'll also consider the solutions that are available to you if you choose to go the self-published route.

1) **Funding**

   A publisher can provide you with money needed to develop your game. It's a great solution if you're struggling to pay your contractors or your own bills. Getting a big chunk of cash up-front can also help you quit your job and concentrate on finishing your game full time.

   Alternatively, you can try to fund your game in other ways. While it's not easy, there are other options available to you, such as lenders, investors, crowdfunding, Patreon, and grants.

2) **Feedback and guidance**

   An experienced publisher can guide you through the process of the development and offer you valuable advice that can save you from making mistakes caused by your lack of experience.

   As an alternative, you can ask for feedback from fellow indie developers. They are easy to meet on Twitter, and most of them are generous with their time if you have specific questions. Also, keep in mind that most publishers are business-oriented people and might not have much experience when it comes to game design, coding, optimization, etc. While publishing companies provide those services, they're usually delegated to other companies or handled by their in-house teams. Your indie developer friends might be better equipped to give you advice and feedback when it comes to the specifics of the

game development.

3) **Quality assurance**

Some publishers might have an in-house quality assurance team, which would make your life a lot easier. After all, it's important to make sure your game ships with as few bugs as possible. Other publishers who cannot afford a full-time QA team delegate testing to other companies they're partnered with.

As a self-published developer, you can choose to do the same: partner with a company that provides quality assurance services. If professional QA services are out of your budget, you can delegate testing to the dedicated members of your community. Even though your community would be happy to help test your game for free, consider providing something in return. If you can't afford a payment, at least offer merchandise, in-game credit, or a free copy of the game when it ships. Profiting from unpaid work is immoral, and it won't help your studio's reputation.

4) **Marketing & PR**

Similarly to quality assurance, a publisher can assist you with marketing and PR. Most likely, your publisher will delegate this task to a marketing agency.

If you don't have a publisher and want to use a marketing agency, you can delegate this dreaded task without the need for an intermediary. Some marketing agencies will agree to work on revenue share basis, but most will expect an upfront payment. Marketing and PR agencies might be a necessary expense for large studios because their fanbases are too large for them to handle. However, for an indie studio with a niche audience, it might be an overkill. In fact, you might be able to market your game more effectively than a marketing agency. Why? Mostly, because you simply *care* more than they do.

I've been approached by several marketing agencies during *DARQ's* development. Some wanted a 30% revenue cut and no upfront

payment. Others offered pre-launch marketing campaigns in the range of 20,000 dollars. While I've never used the services of marketing agencies, I was not impressed with their pitches. In retrospect, some of the games listed in their resumes did much worse than *DARQ*.

Understandably, a marketing agency can only do so much if a game is simply not very good, but that also shows you that delegating marketing doesn't guarantee good results. If you were to spend about 20,000 dollars on a marketing campaign, you would become one of many clients of the agency. Their employees work from nine to five, which is probably less than you do. Your project would be assigned to a marketing specialist who's dealing with a lot of other projects at the same time. They won't think about your game when they go to sleep. They won't spend countless hours trying to invent new ways of reaching your target audience. Sure, they can pitch your game to journalists, streamers, and influencers, but so can you. After your agent's salary is paid using your money, they might spend what's left on Facebook and Instagram ads. While they might have some personal connections with journalists and influencers, the point still stands: they can't guarantee results either way. If you or your publisher were to pay this much money for a marketing campaign, wouldn't you be better off just buying 20,000 dollars' worth of ads on Facebook and Instagram?

If you can't afford to hire a marketing team, there's no need to worry. Many indie studios manage to market their games effectively without spending a dollar. If you continue to educate yourself in this area, and your marketing efforts are consistent, you may not need to spend money on marketing. For example, *DARQ* had a marketing budget of zero dollars, yet it ended up in the top 50 of the most shared games of 2019, according to Metacritic. As a first-time developer, your main goal should be to minimize your expenses. While letting someone else take over marketing might feel like a relief and save you time, it won't guarantee your game's success. If you were to sign with a publisher, try to retain control over the marketing budget. It's easy to pour thousands of dollars into services that ultimately won't have much effect on your sales. In the end, it's you who will have to pay

for them.

5) **Video Game Conventions**

A publisher can fund showcasing your game at video game conventions, like E3, PAX, and GDC. While it's a pricey way to market a game, a publisher might do it more efficiently than if you were to do it by yourself. They may be able to rent a large, attention-grabbing booth to showcase all of their upcoming games in one place. A publisher can also help schedule press tours to make sure that key journalists get to see their titles.

As an indie developer, you may not be able to afford to showcase your game at a convention independently. But it's not essential to your game's success at all. Conventions generate a lot of attention from the press, but it's mostly triple-A games that benefit from it. I've seen many indie developers paying premium prices for tiny booths, electricity, wi-fi, rented equipment, accommodation, and travel. All these expenses can easily add up to around 10,000 dollars per convention. Sadly, such investment in no way guarantees press coverage or a bump in sales.

Having a tiny booth on a large floor doesn't attract much attention. The press is there to see the upcoming titles of triple-A studios, and so are other attendees. Not a lot of people will pay attention to a small booth crammed in among a hundred others. I have no doubt that conventions have the potential to provide an exposure boost to indie developers when done right. There are other opportunities that might come with it: networking with industry professionals, getting player feedback, maybe even winning awards. But is it worth the price? Think about how much more exposure you could get if you were to spend your convention budget on highly targeted YouTube ads. With the average price of 18 cents per view, 10,000 dollars would let you reach over 55,000 potential customers this way. They would get to see your game in the comfort of their home, in a distraction-free environment, just a click away from wishlisting it. How many people will get to see your game in a crowded convention center dominated by triple-A studios? Most importantly, how many of them will

remember its title so they can wishlist it a few days later when they get back home?

6) Customer base

One of the main advantages of signing with a publisher is the ability to utilize their pre-existing customer base. Needless to say, this is a major asset, and depending on the publisher's notoriety, it should be seriously considered.

The real value a publisher can bring is that it has created a closed loop community around its portfolio. The value of having your game introduced into this community can be the difference between obscurity and success. It's of course possible to buy engagement, but introduction to an active community with genuine interest in your game type is an invaluable ticket in an increasingly noisy market.

Scott Millard | Managing Director at *Feardemic*

If you choose to go solo, you won't have a pre-existing customer base. But you can start building it, so you're not dependent on a publisher each time you want to make a game. There's one more thing you can do: partner with another developer who is working on a game in your genre. Discuss how you can help each other thrive and expand your communities through cross-promotion and other forms of collaboration. If you can't find developers who are open to collaboration, reach out to me directly (@UnfoldGames)—I'm always happy to discuss potential partnerships if they were to be mutually beneficial.

7) Distribution

One of the responsibilities of a publisher is to distribute your game. Since most indie games are distributed digitally, it's relatively easy to get it done within the PC market. Console distribution is a more involved process, so having a publisher on your side can make a big difference.

If you choose a self-published route, you will have no problem distributing your game on PC platforms, like Steam, GOG, and possibly Epic Game Store. As of writing this book, Epic Game Store is still in its early days and remains hand-curated. Hopefully, it won't be long before the store opens their doors to all developers. As far as physical distribution is concerned, it's something that would be very difficult to pull off without the help of a publisher.

8) Console porting

Publishers can provide assistance with porting your game to consoles. Again, porting might be handled by the in-house team or could be delegated to a port house—a company that specializes in porting games to consoles.

If you want to release your game on consoles and you're self-published, you're going to have to handle porting yourself. Porting to consoles is not a simple process and requires access to game development kits. Game development kits, or simply, dev kits, are a special kind of hardware provided by console manufacturers that allow developers to port their games to the manufacturer's platforms. Dev kits are not available to the general public. Getting one requires entering into an agreement with the manufacturer and paying a fee. Getting one can also take quite a long time. Whatever console you're porting to, expect it to be a lengthy process if you want to do it all by yourself.

Even though both Unity and Unreal Engine make it easy to export your game in the format required by your target platform, there's a lot of additional work and optimization that is often required to make everything work as it should. Since console manufacturers have full control over what game is allowed to be released on their platform, you will have to go through a process known as *certification*. This is a process in which your game is meticulously tested to ensure that it's bug-free and customer-friendly. The certification process on consoles tends to be slow, so a publisher's experience in this area would certainly be helpful.

Alternatively, you can delegate porting to a company that specializes in bringing PC games to the console market. If you're worried about the costs, don't be. First of all, you don't have to port your game to consoles before you release it on PC. Arguably, you should first ensure that there's enough demand for it to justify the costs associated with preparing your title for console releases. If your PC launch goes well, you'll be able to afford to delegate console ports on a flat-fee basis.

Port house fees vary a lot depending on where the company is located. Companies operating in North America tend to be much more expensive than those located in Eastern Europe. Price is not the only factor you should take into account, though. If you were to work with a port-house, make sure they are a reputable company with a proven track record. After all, you want to feel safe sending your entire project to them. Preferably, you want to work with somebody local, who would be able to communicate well in your language. Also, in the unlikely case of litigation resulting from a dispute or copyright violation, a local port-house is much easier to pin down than a company located overseas.

If you're determined to release on all platforms simultaneously, or if you simply lack the capital to hire port houses after your PC launch, there's still a solution. Some port houses will agree to port your game in exchange for revenue share. You will retain full ownership of the game, and it will be registered using your credentials. Some of the offers that I came across involved thirty-percent revenue cuts per console.

Finally, if you're completely unfamiliar with this market, you can still team up with a publisher, but just for console releases. This way, you can focus on your next game while your publisher is dealing with porting and the pre-launch marketing campaign for your console ports.

9) Business partnerships

By signing with a publisher, you may gain access to their network of content creators, streamers, and business partners who may be useful in promoting your game.

If you decide to go solo, it's up to you to build your own network and enter into partnerships that you feel are right for your studio. For example, I chose to team up with *Intel* by becoming a part of their GameDev Boost program before releasing *DARQ*. Partnering with *Intel* increased the exposure and the credibility of my studio.

10) Localization

In most cases, localization is a complex process that involves much more than just translating text. Games that have a lot of text or spoken dialog require professional localization services that usually include several stages: translation, proofreading, regional and cultural adaptation, editing, integration, etc. There are companies that specialize in video game localization, and your publisher is most likely partnered with one of them.

As a self-published developer, your best course of action is to avoid making a game that has a lot of text or spoken dialog. The more your game relies on text, the more costly it will be to bring your game to other markets. If you can find a way to limit your text to user interface elements, you will be able to simplify the whole process and save a lot of money. Regardless of your specific needs, you can outsource localization to a translating agency, just like a publisher would. If your game has just a few lines of text, you can get away with much simpler solutions, such as hiring individual contractors or asking your community members to help you.

Also, remember what your target demographic is. If you're releasing your game on Steam, your game should be available in English, Simplified Chinese, Russian, German, Portuguese, and hopefully more. When I was working on *DARQ*, it was important to me to make it available in as many languages as possible. I tried my best to limit the use of text in the game, so in the end, there were just a few

hundred words to translate. That allowed for translating *DARQ* to 19 languages thanks to the active members of our community. Everybody who helped with localization was featured in the credit roll and was given a copy of the game.

Ultimately, it all comes down to the long-term vision for your company and the deal you're offered. Even though signing with a publisher can have a lot of advantages, here are my reasons for deciding to go solo:

- I wanted to acquire as many skills as possible, so I could publish *DARQ* and future projects with as much competence as a publisher would.
- I wanted to retain full ownership of my intellectual property.
- I wanted to have full creative control over *DARQ*.
- I wanted to have full control over expenses associated with publishing and marketing (which in my case, turned out to be close to zero).
- I wanted to focus on growing my own community instead of having to rely on the pre-existing customer base of a publisher. While it's definitely a longer and more difficult way to go, it's the only one that's sustainable and scalable in the long term.
- I wanted to be in control of the release date, price point, and discount schedule, so I could serve my customers to the best of my ability.

To summarize, there's no right or wrong way to go about it. There are simply too many factors to consider. Whether you pursue a publishing deal or not, make sure you're educated on the topic and understand the reasons behind your decision.

Video game content rating

Also known as maturity rating, content rating systems are used to assign video games into appropriate age categories based on their content. Getting

your game rated is voluntary and comes with a fee. While platforms like Steam and GOG don't require it, console and mobile manufacturers do.

Every region has its own content rating system. That's why maturity rating is considered a part of the localization process. The most common age-based rating systems include:

1) ESRB, used in the United States
2) PEGI, used in most European countries
3) USK, used in Germany
4) RARS, used in Russia

If you're focusing on the PC market, feel free to skip the maturity rating for now.

Digital Storefronts

Since physical retail for PC games is slowly fading away, we're going to focus on digital distribution only. Digital storefronts are the equivalent of physical retailers. They host your product, deal with customers, issue refunds, process payments, collect revenue, and pay you a royalty. They also let you set the price and give you the freedom to run promotional discounts or to participate in seasonal sales. While digital storefronts won't market your game, they will do their best to improve its visibility if it proves to be popular.

1) Steam

Most PC games are distributed by Steam—the largest online retail store owned by Valve. The store comes with its own launcher that offers a variety of features to its customers. As of writing this book, Steam is not curated so anybody can release a game on their platform. Listing a game on Steam requires paying a 100-dollar fee, filling out some forms, and going through a simple certification process.

Steam takes a 30% cut from the revenue generated from the sales of your game. If your revenue exceeds 10 million dollars, Steam's cut

will drop to 25%. If you happen to generate over 50 million in revenue, the cut will drop to 20%. Needless to say, if you're an indie developer, chances are you'll stay within the first tier. This revenue split model is meant to encourage triple-A developers to release their games on Steam instead of trying to build their own launchers, which has been a trend among large studios.

In addition to selling your game on Steam, you're also allowed to generate retail keys. Essentially, retail keys are twelve-digit codes that allow customers to activate your game on Steam. You can generate as many retail keys as you want. It's completely free, and you can sell them outside of Steam. In other words, if you want to keep 100% of your revenue, consider selling Steam keys directly on your website. You can also sell them in other stores, although each store will expect a revenue cut.

I also feel it's important to mention that Steam pays you your royalties every month, which is quite convenient. At the end of each month, you will receive your net revenue from the previous month. For example, your February revenue will be transferred to your bank account at the end of March. Also, Steam lets you monitor the sales of your game in real-time, which is a welcome feature.

The important part of the Steam platform is its digital rights management system (DRM). The goal of DRMs is to prevent unauthorized access to digital media and to limit consumer's ability to copy and illegally redistribute digital goods. Steam offers a light version of DRM through the use of their launcher. It's not meant to prevent piracy, although it helps to limit casual attempts at copying and redistributing game files. The main purpose of Steam's DRM is to verify the ownership of the game and to make sure all features offered by the launcher will work properly when a game is running. To clarify, if a game is activated through Steam, it will require the Steam launcher to run. In fact, the launcher functions as a "wrapper" for your game. In addition to Steam DRM, you have the freedom to incorporate third-party DRM solutions. Triple-A studios use much more aggressive forms of DRM that often require creating an account with a third-party

DRM provider, staying online when playing, and other inconveniences that make playing a legally purchased game unnecessarily difficult. Such forms of DRM are universally hated by the PC community, so I wouldn't recommend using them. No DRM can protect a game from being pirated—it can only delay it at best.

2) GOG

Formerly known as Good Old Games, it's a digital distribution platform owned by CD Projekt—one of the top game development companies known for its *Witcher* series as well as *Cyberpunk 2077*, which remains in development as of writing this book.

At first, GOG was a store that only offered classic games, but as the platform grew, they expanded their handpicked collection to include new releases as well. The market share of GOG is relatively small compared to Steam, but they have an active and loyal fan base. Some of your target audience will have a strong preference for GOG over Steam.

Apart from a generous refund policy and a customer-first approach, all games listed on GOG are DRM-free. While GOG provides its own launcher, using it is entirely optional. Players can simply download a game, install it, play it without ever using the launcher, copy it to a different computer, etc. The GOG launcher offers a lot of useful features, but the fact that it's optional is what made this platform popular and universally liked. Since all games are DRM-free, consumers feel like they actually own the games as opposed to being allowed to use them as a service.

GOG's team curates what games make it to their store. They might approach you if they find your game to be a good fit for their audience. If you don't hear from them, you can simply reach out and pitch your game. Similarly to Steam, GOG takes 30% of your game's revenue.

GOG has a different feel when it comes to interacting with the platform as a developer. Steam provides you with a lot of tools that

allow you to do everything by yourself. With GOG, you will be in touch with several representatives who will address your needs. That includes the creation of your game's page, making changes to it, putting your game on sale, etc. While the Steam approach feels more efficient, it's nice to deal with actual people who are there to assist you every step of the way. My experience with the GOG team has been great; it's been a pleasure to interact with their friendly and helpful staff.

Keep in mind that GOG pays out royalties four times a year. There's no way you can check your earnings before you receive your quarterly statement.

3) Epic Game Store

Epic Game Store (EGS) started as the launcher for Epic Game's hit title *Fortnite* and later expanded to become a digital store. As of writing this book, EGS is still in its early stages of growth and doesn't offer as many features and games as its competitors. While its market share is estimated to be relatively small, EGS is growing rapidly and expanding its fan base thanks to free games offered weekly and to the many titles that are exclusive to their platform.

EGS has a strong preference for exclusivity deals, especially when it comes to indie titles. I had the chance to learn that firsthand when an EGS representative reached out to me right before I was about to launch *DARQ* on Steam and GOG. I was offered an exclusivity deal, which I rejected because I had already promised to release *DARQ* on other platforms. I wanted to release *DARQ* on EGS in addition to Steam and GOG, but I was told it wasn't an option. This story made headlines, so if you're interested in learning more, read the post I wrote on **Medium.com**: *Why I turned down the exclusivity deal from the Epic Games Store*.

While EGS often becomes the subject of heated debates, it's a rapidly expanding platform that brings much-needed competition to the market of PC games. Their customer base isn't as big as Steam's, but they try to stay competitive by offering a more generous revenue split

to the developers. When a game is sold on the Epic Games Store, the developer gets to keep 88% of their profits, as opposed to 70%, which is the industry standard. Hopefully, more platforms will follow this trend.

4) **Itch.io**

This is a small yet unique digital storefront. It's a marketplace dedicated to indie games, where developers are in complete control of their business model. If you choose to list your game on Itch.io, customers will get to use the "pay what you want" option, as long as it's above the minimum price set by you. Many games listed on Itch.io are free, so customers can donate whatever they feel is appropriate for the game, or simply not donate at all. When it comes to revenue split, you're in full control here! It's the only platform that lets you decide how much revenue you are willing to share with the store. In other words, it's a special place full of unique indie creations.

You can also choose your preferred way of getting paid. You can either get paid directly by your customers or nominate Itch.io to process payments on your behalf. While both ways come with some pros and cons, it's the most developer-friendly system you'll see among digital retailers.

Because of a limited audience, this store alone won't guarantee your financial success. Regardless, it could be a good place to get your feet wet and gather players' feedback before you launch on other platforms. It's also a good opportunity for exposure since many content creators regularly check this platform looking to showcase new and unique games on their channels.

Key retailers

When it comes to key retailers, things work a bit differently. First of all, key retailers don't host your game. Instead, they will ask you to provide a batch of Steam keys so they can sell them at a lower price to their

customers. Such stores have full control over pricing, so they may choose to sell your game at a discount, make it a part of a larger promotional event or put it in a bundle. To clarify, a bundle is two or more titles sold together, usually at a significant discount. A key retailer would try to maximize their profits, so they won’t lower your game’s price more than necessary. But be aware that the core of their business model is offering better deals than the main storefronts.

Key retailers are good at marketing and have their own networks of influencers, content creators, and streamers. However, their main priority is to sell as many copies as possible, so be careful when you’re offered a deal with a key retailer. It may be a good idea to wait until your game gets older and your sales slow down. The worst thing you can do is to devalue your game too early. Once your game ends up in a bundle at a fraction of the original price, it will be hard to restore the perception of its original value.

The main key retailers include Humble, Fanatical, Green Man Gaming, and Chrono.gg. You may be approached by their representatives before or after your launch, or you can pitch your game to them when you feel the time is right.

Key Takeaways

Signing with a publisher can be a great move for your studio. It can also be a big mistake. There are many factors involved in making this decision, so be sure to educate yourself on this topic.

If you decide to partner up with a publisher, be sure to hire an attorney to negotiate on your behalf.

If you decide to self-publish your game, have a plan when it comes to funding, quality assurance, localization, porting, distribution, marketing, and other tasks that are usually handled by a publisher.

Learn how major digital retailers operate. Steam should be your main priority. As of writing this book, it has the largest share of the PC market. Also learn about GOG, Epic Game Store, and Itch.io.

Consider selling Steam keys on your website so you can pocket 100% of the revenue.

Once your game gets older, consider making deals with key retailers, like Humble, Fanatical, Green Man Gaming, and Chrono.gg.

• CHAPTER 8 •

Preparing for Launch

Preparing for launch includes a number of important tasks. Some, such as creating your store page, should happen relatively early. Others should be tackled much later when you're approaching your launch day. Let's go over the pre-launch checklist and discuss each task individually.

Store pages and wishlists

Steam will likely be responsible for the vast majority of your PC sales, so you should prioritize setting up your store page on their platform. When should you create a Steam page for your game? As soon as possible, as long as you have a good-looking trailer and some screenshots to show off.

The main reason for listing your game on Steam when it's still in development is to collect wishlists and to grow your following. When a Steam user adds your game to their wishlist, it means they are interested in it. Accumulating a large number of wishlists before launch is the single-most important thing you can do to increase your chances of success. Why? When your game launches, Steam sends email notifications to all users who wishlisted your game. The more sales you can generate on the launch day, the more visibility your game will receive within the store. As one would expect, Steam algorithms favor games that sell well. As a result, games that have pre-existing audiences receive more exposure on the platform. Titles

that launch with low wishlist numbers are unlikely to attract much attention. As a result, they don't receive much visibility within the platform.

Not all users who wishlisted your game will buy it on the launch date. Most of them won't. Some might buy it when your game goes on sale, months or years later. Be ready to see a large portion of your wishlists never convert to sales. The conversion rate of wishlists varies greatly among games. It depends on multiple factors, such as your price point, user reviews, the Metacritic score, and how long ago your game was wishlisted. If somebody wishlisted your game three years before launch, they might not remember your game when they receive an email notification. Furthermore, old wishlists might have a lower conversion rate because some of those users are no longer active on the platform. Steam users don't spend all their lives playing games. They also start families, get full-time jobs, or simply start new accounts on other platforms. In other words, it's not always the number of wishlists that matters—it's also the *quality* of them. For example, 10,000 wishlists collected a week before the launch is probably worth more than 30,000 accumulated gradually over several years.

So how many wishlists should you aim to have before you launch your game? While there's no way to answer this question accurately, multiple reports indicate that 20,000 is usually enough to get your game featured on the "New and Popular" tab on the front page of Steam. That's a good goal to have because being listed on the front page of Steam will give your game a much-needed boost on your launch day. Word on the street is that 50,000 wishlists or more, combined with a good user score, can get your game displayed in the "Featured and Recommended" section. It's the holy grail of exposure on Steam and the largest spot on the front page. DARQ had over 70,000 wishlists when it launched, and it was listed in both "New and Popular" as well as "Best Selling" tabs. A day later, it made it to the "Featured and Recommended" section.

The easiest way to accumulate wishlists is through organic traffic within Steam, so simply start this process as early as possible. While wishlists accumulated closer to launch will probably convert better, you want to have as many of them as possible. To increase your reach within Steam, make sure to mark your game with tags. If you're not sure what tags will work best, look up games that are similar to yours and pay attention to their user-defined tags. See if any of them apply to your game. Once your game has appropriate tags, it will start showing up on other game pages, in the "More

Like This” section. If you do it right, it will ensure a steady flow of traffic to your page. If your page looks good, you should see a good conversion of visitors to wishlists.

Apart from growing your wishlists organically, do your best to acquire as many as possible through your marketing efforts. “Add to your wishlist” should become your main call-to-action once you have a Steam page. Remind people to wishlist your game at the end of your trailer, or when you post your GIFs on social media. It’s a slow process, so don’t get discouraged if it takes a while. That’s why you want to start thinking about increasing your wishlist numbers as early as possible.

Apart from wishlists, focus on making your Steam page look as attractive as possible. Learn from successful titles and study their store pages. They tend to have little text and a lot of eye-catching images and GIFs. Speaking of text, make sure it’s translated to multiple languages, so you don’t alienate your potential customers.

Creating a Steam page and not updating it until the game’s release is a bad idea. You should try to avoid making it look abandoned. Update it frequently, check the discussion tab, answer questions, and keep your audience up to date by posting regular announcements. Don’t be discouraged by the lack of activity at first. It will become more active with time, as word about your game spreads all over the internet.

GOG also allows its users to wishlist games. If your game has been approved by GOG, try to get your page up and running as soon as possible. You won’t be able to edit it yourself, but GOG representatives will be happy to set it up for you if you provide all the marketing materials.

Release date

Be careful when announcing your release date—don’t do it too early. As a first-time developer, you may not realize what it takes to get your game integrated with platforms like Steam, GOG, and others. Implementing achievements, trading cards, testing, and the certification process take time. If you’ve never done this before, don’t announce your release date until everything is one hundred percent ready to go.

When choosing your release date, consider the following:

1) Avoid releasing your game close to big industry events, like E3, PAX, GDC, etc. All the attention of the press and content creators will be focused on the titles announced during these events.

2) Avoid releasing your game during major sale events. Both Steam and GOG have annual sales that offer great games at major discounts. Trying to sell your title at full price while a lot of high-quality games are offered at a 75% discount might result in a disastrous launch. Summer and Winter Sales tend to last about two weeks. While their dates change every year, they generally take place around the same time and get announced in advance. They attract a lot of customers who buy games in bulk. Releasing your game right after a big sale is a bad idea too, since you would be trying to sell a product to customers with empty wallets and backlogs of games waiting to be played. Apart from Summer and Winter Sales, keep an eye out on other sale events, which usually last a week or less. These include Halloween Sale, Black Friday Sale, and Lunar New Year Sale.

3) Avoid launching your game on the day when a highly anticipated triple-A title is being released. It's unlikely you'll get any attention during this time.

4) Similarly, don't release your game if another game in your genre has launched recently. Triple-A or indie, you want to have as little competition as possible within your niche when you launch.

Launch Discount

Steam allows you to release your game with a discount. Launch discount lasts seven days and is usually set to ten or fifteen percent. The decision is yours, as it's entirely optional. Keep in mind that if you decide to run a

launch discount, you will not be allowed to discount your game for two months after the launch discount expires.

On the one hand, a launch discount creates a sense of urgency. Steam customers are always on the lookout for a good deal, and knowing that the price will go up soon might motivate them to purchase your game. On the other hand, a launch discount can communicate the developer's lack of confidence in their creation. After all, games that are in high demand don't lower their prices and often openly communicate to their players that they won't go on sale anytime soon, if ever. Some games get away with permanently increasing their prices as they grow in content.

Early Access

Steam gives you the ability to sell your game as it's still being developed. While many players have been disappointed with Early Access releases, some titles manage to stand out from the crowd and make it their path to success. Not all games will benefit from being released in an unfinished state. You should consider Early Access release if:

You're looking for players' feedback as you continue to tweak your game and add content to it.

Your title is a sandbox game (it has a great replayability value and is potentially endless). Story-driven linear games obviously don't fit into the Early Access formula.

You're willing to make your community a part of your development team. Early Access suggests that you will listen to your players' feedback and respond accordingly. Players who purchase an unfinished game expect to be heard when they express their opinions.

You're running out of money and looking for a sustainable way to continue the development of your title. Early Access allows you to start generating revenue as you continue to develop your game.

If you think Early Access is a good fit for you, be aware of its potential pitfalls. Your unfinished game will be treated as your main release. It will be reviewed by journalists, content creators, streamers, and your customers. If you release a buggy and unplayable build, you will be judged accordingly. Also, some of the Early Access titles never get to the finished state, so understandably many customers see such games as a high-risk purchase.

As of writing this book, there are multiple titles that are in Early Access and doing great, such as *Factorio* and *Raft*. There's a lot you can learn from watching these games grow over the years. Their respective developers are doing a great job improving their games and engaging with their communities.

Press reviews & embargo date

Three weeks before your launch day, you should start sending review keys to journalists. Depending on the outlet, they may need three to five keys. Hopefully, by this point, your game will be well known, and it won't take much effort to get the attention of the press. Perhaps some media outlets will reach out to you before you get a chance to contact them. I can't tell you how thrilled I was when I received an email from IGN's executive editor who requested *DARQ's* review keys. It was a meaningful moment to me that made me reflect on the humble beginnings of my project.

When sending review keys, don't forget to state your embargo date. Essentially, embargo is a request not to reveal any materials associated with the game until the indicated date. In other words, it's the day you want the reviews to come out. Most commonly, it would be the launch day or the day before. Since you cannot demand the embargo date be respected, simply ask politely.

There are a lot of media outlets out there. Should you send review keys to all of them? Probably not. Cold emailing people rarely works. Your time would be better spent sending highly personalized emails to the journalists who hopefully have already seen your game on Twitter or elsewhere. While any press is better than no press, try to get your game reviewed by the most professional and respected media outlets. Why?

- They are more likely to respect your embargo date.
- They won’t try to steal and resell your keys.
- Their reviews will likely qualify to be included in your Metacritic score.
- Review quotes from reputable outlets give your game legitimacy. You can use them in your marketing materials. Using quotes from unknown sites makes your game look unprofessional.

Whomever you’re contacting, make sure the reviewer actually likes games in your genre. While journalists do their best to stay as objective and professional as possible, their personal taste will surely affect their opinion of your game. If you ask somebody who is a fan of family-friendly titles to review a violent horror game, don’t be surprised if they don’t have a good time.

Content creators and streamers

You will likely receive a lot of key requests from YouTubers and streamers as your game approaches its launch. Some will be legitimate, while others will be fake. You will have to become very good at spotting scammers who just want to get as many keys as possible to resell them illegally. Scammers will usually pretend to own a channel. They might even use an email that looks deceptively similar to the one listed on said channel. When in doubt, ask them to verify their request by messaging you from one of their official social media accounts.

Some popular red flags include:

1) **Claiming to be a popular YouTuber or streamer**
   Popular content creators are unlikely to contact you directly. If they like your game, they will buy it.

2) **Asking for more than one key**
   While media outlets usually need several keys, YouTubers and

streamers rarely need more than one. They might want to do a giveaway, which justifies asking for multiple keys. If you're okay with giving away keys, check if this is something they do on a regular basis.

3) **Having a lot of subscribers but not many views**
   Some content creators who contact you might link to their actual channels. You may successfully verify that the email they're using matches the one listed on YouTube. But what if their subscriber count doesn't match the activity on their channel? Having a million subscribers and two hundred views per video suggests that those aren't real subscribers.

4) **Covering games that don't match your genre**
   If you manage to verify that the request comes from somebody who actually owns the channel, you may want to check if their video content matches what your game has to offer. If it's a channel dedicated to *Minecraft* videos, it's suspicious if they want to review your horror game.

Apart from replying to emails, you should be actively pitching your game to content creators. If your game is story-driven, you should avoid sending review keys too early. You don't want to spoil the game before it's actually released. If your game is not linear and allows for countless hours of unique gameplay, YouTubers and streamers are your best allies. You should offer them review keys way before your release date.

Prepare additional marketing assets

If you followed the advice in this book, chances are the launch of your first game will be the most overwhelming day of your life. Once you press the "release app" button, there will be no time to prepare good-looking images and videos for your social media accounts. Make sure to have your launch marketing assets prepared in advance. Leave no stone unturned, because you won't have time to deal with any of that on the launch day.

Test patching your game

It’s not unlikely that a game-breaking bug finds its way into your final build. That would not be ideal, but don’t worry if that happens. Just make sure you can patch your game quickly, preferably the same day. In order to get it done efficiently, you need to know how patching works on Steam and other platforms. Storefronts like Steam make the whole process easy—you just need to test it before launch.

If you need to release a patch on your launch day, do it as soon as possible, but also don’t panic. Fix bugs carefully, so you don’t accidentally create new bugs. You need to test the patched build before releasing it to the public. The last thing you want is to ruin your reputation by releasing a patch that breaks the game.

Create a customer support email

While the Steam community discussion tab is often used to report bugs, it’s a good idea to handle customer’s issues via email. Besides, Steam requires you to have a customer support email address. You don’t want it to be your company’s main email account—it would be too confusing to mix customer and business communication within a single account. Your customer support email address should also be listed in your end-user license agreement and privacy policy.

Have a lot of extra keys ready

If you’re running low on review keys, make sure to have Steam approve a new batch before your release day. Chances are, you’ll be approached by YouTubers and streamers right after your game launches. You want to be ready to send them keys on a moment’s notice—just make sure the requests are legitimate. Ideally, you want to deal with content creators on social

media, where you can verify that they're messaging you from their verified accounts.

Expand your team

If you did your job right, your launch day might be too big for you to handle. Your email box and social media might explode with messages and comments. You will need help for at least a couple of days following your game's release. For example, while you handle patching the game, your team members could keep an eye on your email, social media, customer support, community forum, etc.

I didn't think of it when I launched *DARQ*. Things got overwhelming the minute the game went live. I couldn't possibly respond to the hundreds of emails, comments, and messages I received on the launch day and after. I needed a team to help me deal with social media, customer support, community forum, and emails, but it was too late to put it together. That's why I'm sharing this advice with you, hoping that you learn from my mistakes.

Prepare for Steam broadcast

Your store page can host a live broadcast, and you should definitely take advantage of this opportunity on the launch day. Your stream will be visible on the top of your page. With a bit of luck and a lot of viewers, it can even get featured on Steam's front page.

Hosting a live stream on your launch day can give you a nice visibility boost. You will have to do quite a bit of research if you've never live-streamed before. First, you will need to read Steamworks documentation on broadcasting. Additionally, you'll have to learn the basics of streaming software, like **OBS**. You may also choose to stream simultaneously on multiple platforms, which might dramatically increase your visibility. To do that, you will want to use **Restream**. It's a cloud-based multistreaming service that allows you to share your stream on various platforms, such as

Twitch, Facebook, YouTube, and LinkedIn. Make sure you test broadcasting across all your accounts, so you avoid technical problems on your launch day.

Write a Steam announcement draft

Write a draft in which you announce the game launch as well as inform your fans about your streaming schedule. Be sure to include eye-catching screenshots and other marketing assets, like quotes from the reviews or scores your game has received. Keep the draft unpublished until your launch day.

Write a newsletter draft

Similarly, write a draft of your newsletter update. Make sure to include the links to your store pages as well as information about your upcoming broadcast.

Get a lot of rest

Tomorrow's the big day—your game's launch! You've worked hard and sacrificed a lot to get to this point. Now it's time to rest. Make sure you get a good night's sleep before your launch day. You're going to need every bit of energy. Depending on how things go, you may have to stay up all night patching your game, or you might be simply too excited to fall asleep. When *DARQ* launched, I couldn't help but stay up all night watching playthrough videos and live streams, reading and answering emails, comments, and messages. In all honesty, the launch day was the most exciting day of my life.

Key Takeaways

Preparing for launch involves a number of tasks. Some need to be taken care of way in advance, like creating your store pages.

Once your store pages are up, all your marketing efforts should be focused on collecting wishlists.

Pick a release date. Try to release your game when there are no large triple-A releases, or industry events, like PAX, E3, and GDC. Also, avoid releasing during seasonal sales.

Don’t announce your release date until your game is 100% finished and integrated with the storefronts.

Send review keys to the press three weeks ahead of your launch date. State your embargo date.

Send review keys to YouTubers and streamers.

Prepare marketing materials for your launch day and write drafts of Steam and newsletter announcements.

Test patching your game, so you know how to do it fast if game-breaking bugs are discovered on your launch day.

Generate extra review keys for your launch day; you may need them.

Prepare for Steam broadcast. Test all the necessary software, so everything goes smoothly on your launch day.

Get a lot of rest the night before.

• CHAPTER 9 •

Launch!

It feels surreal when this day finally comes. I hope you get good rest and feel ready to face one of the wildest days of your life. If you followed the advice in this book, you already have an active community of people who can't wait to play your game. You also have a fair amount of wishlists to please the Steam algorithms, so don't worry—things will go well.

While the launch day is perhaps the most important day in your game's lifespan, remember that it's just the beginning. If it doesn't go as well as you hoped, it's not the end of the world.

While most games generate a large portion of their annual revenue on the launch day, others have a completely different trajectory of sales. If your launch day ends up feeling slow, your sales might accelerate gradually as your game starts spreading on YouTube and streaming platforms.

> *After four years of development, I felt pretty hopeless when One Hour One Life sold only 315 units on its 2018 Steam launch day, especially since "The Castle Doctrine" launched with 2,475 day-one units back in 2014. Ouch! But I was pleasantly shocked when those initially dismal numbers rose day after day over the following weeks and months. "One Hour One Life" did about 10x worse than "The Castle Doctrine" on launch day, but it ended up making about 10 times more money over the long haul. Long-term word of mouth has completely replaced launch-day*

press hype.

Jason Rohrer | @jasonrohrer | Developer
One Hour One Life, The Castle Doctrine

In other words, accept whatever happens today, and give it all you've got because there's still a lot to do. Are you ready to face one of the most exciting days of your life? Let's go!

Understand the goal

Now that the release of your game is minutes away, you can't just sit around and do nothing. Today is the day the Steam algorithms learn about your game, specifically about how popular it is and how customers respond to it. As in any relationship, first impressions matter. Your goal is to generate as many sales on day one as you possibly can, so Steam algorithms can feature your game on "New and Popular" and "Best Selling" tabs. Getting listed on the front page can help you a lot, so do your best to make it happen. While a big portion of sales will come from email notifications sent to users who wishlisted your game, you should do your best to spread the word in your community and beyond. Keep in mind that Steam tends to send email notifications at random times during the day. Not all users who wishlisted your game will get an email the minute your game goes live. That's why it's important to do everything in your power to bring as much external traffic to your store page as soon as your game launches.

Execute the plan

Once you release your game, have a step-by-step plan ready to execute as quickly as possible:

1) **Send out a newsletter**
   Hopefully, you have written a draft the day before, and now it's just a

click away from being sent. Tell your audience that your game has been released. Be sure to provide links to your store pages and don't forget to mention your stream schedule.

2) **Post on your social media**
   Notify your social media followers and use the marketing materials you have prepared in advance. The launch of your game should be announced with some eye-catching visuals. Also, don't forget to mention your live broadcast.

3) **Post a Steam announcement**
   Remember the draft you had written earlier? It's now time to post it and let your Steam followers know that your game is out and that you're about to start a live stream.

4) **Post a Steam event**
   As you're preparing to broadcast, post an event announcing your live stream. It will fire up an additional notification to your Steam followers.

5) **Start Steam broadcast**
   There's a lot you can do during your Steam broadcast. Apart from simply playing your game, you can introduce your team, talk about the development, mention fun behind-the-scenes facts, bring in guests, and give away copies of your game. You can use Twitch chat to engage with the community and answer their questions. Think of ways you can keep your viewers engaged.

6) **Delegate other tasks**
   If you followed my advice from the previous chapter, you have a few team members helping you with social media, customer support, Steam community forums, and emails. It will be an overwhelming time, so having some help while you stream is crucial. Instruct your team members on how to handle key requests, answer customer support emails, etc.

Engage with your community

Once you're done streaming, you should proceed to check your store page and community forum. What are the questions being asked? Are there any bugs reported? Are there any negative user reviews? Make sure you're as responsive as possible. Engage with your customers. Collect all the bug reports and store them in a spreadsheet. Promise to fix the bugs soon, and make sure you keep your promise. If there are any game-breaking bugs that are easy to fix, try to fix them *immediately*. Releasing a patch on a moment's notice is the right thing to do. After all, your customers spent their hard-earned money to enjoy your game. It's your duty to deliver a working product. While it's common to discover new bugs on the launch day, your ability to address them fast will speak volumes about you and your studio. It's an opportunity to show that you care about your customers, not just their wallets.

Check your social media comments, messages, and email. Keep an eye out for Steam key requests; if they look valid, send them. Your game needs as much attention as it can get at the moment.

Breathe

Allow yourself to take a break and celebrate. You've done something amazing—something that not many developers will ever get to experience. Bringing a project to the finish line is an achievement in itself. Now you know how difficult it *really* is, and it only makes you enjoy it more. Reward yourself. Have a glass of wine, or a cup of tea—whatever your ritual is, let yourself take pleasure in this moment. I don't think you'll be able to stop yourself from constantly checking how many copies have been sold. At least try not to overdo it. Relax and just breathe. If you're feeling too excited to sit still, head over to YouTube or Twitch. Enjoy watching people playing your game. It will feel extremely rewarding and give meaning to all the hard work and sacrifice you put into your game.

I hope you're not alone during this special, potentially life-changing day. Surround yourself with family, friends, or team members. Have a party and

celebrate your game’s release. It’s an interesting feeling, isn’t it? You’ve spent thousands of hours tweaking your creation in solitude. You might have lost track of time, and it might have felt like a never-ending process. Now your game is out of your control. Your audience will have unique experiences playing it. YouTubers and streamers will make videos of it that will forever exist on the internet. Chances are, you will never have enough time to watch all playthroughs of it.

Thanks to your self-discipline and dedication, you’ve created something bigger than you. Almost certainly, your creation will outlive you. It will exist indefinitely, in one form or another. You’ve created a piece of art, something that has value. While some people will never know about it, your game will touch the lives of many. As you ponder the implications of what’s just happened, I’m raising an imaginary glass in your honor. You’ve done it. Against all odds. Congratulations.

Key Takeaways

Accept whatever happens on your launch day.

Remember that selling a lot of copies on the launch day is not a prerequisite of long-term success.

Be quick to execute your launch plan using all the assets you prepared in advance.

Do everything in your power to bring external traffic to your store page to please Steam algorithms.

Broadcast your game and share your stream among multiple platforms.

Use the help of other team members to handle comments, forum activity, reviews, key requests, and customer support.

Celebrate this incredible achievement. You deserve to have a party. Surround yourself with your family, friends, and team members. Not many indie developers end up finishing and releasing their games. Be proud of what you’ve accomplished.

• CHAPTER 10 •

What Next?

Now that your game is released, you should think about your next steps. Regardless of how your launch went, there's so much you can do to continue to grow your game's popularity and extend its lifespan. It's a lot easier and less risky to capitalize on your existing IP than to create a new one from scratch. If you simply abandon your game, the momentum you managed to generate on your launch day will quickly fade away, and your title will get lost in the crowd of other releases. Eventually, moving on to developing your next game will be the next step in your career. Before you do, however, make sure you explore all opportunities that your first game has to offer.

> *The accepted wisdom was that your success or failure depended solely on your day one sales. Draw Distance is living proof that that's not always the case. "Serial Cleaner" was quickly written off shortly after sales stopped in the first few weeks after the release. Several months after, however, a trickle of sales started growing into some substantial numbers thanks to promotions and subscriptions. Here we are three years later celebrating a million users across all platforms.*
>
> **Michał Mielcarek** | CEO at Draw Distance
> *Serial Cleaner, Vampire: The Masquerade – Coteries of New York*

Discounts

Not many customers are willing to pay the full price for a game unless it's a highly popular title. You'll notice that your wishlist numbers will shoot up once your game is released. It indicates that a lot of people are waiting for it to go on sale. Participating in seasonal sales and discounting your game on your own can increase your earnings dramatically. Keep in mind that Steam allows you to put your game on sale every six weeks. Every promotion can last no longer than two weeks.

When it comes to discounting your game, it's important to follow one rule: there's a buyer at every price point. Some of your dedicated fans will be willing to buy your game at the full price. Others will consider buying it at a 10% discount. There are customers who never buy games unless they are 50% off or more. In other words, you don't want to discount your game too much too early. If you do, you will miss out on the opportunity to sell copies of your game at higher prices. Start with smaller discounts and gradually increase them over the years.

Don't hesitate to repeat the same discount twice or decrease the discount once in a while. If you have a lot of wishlists that haven't converted to sales yet, it doesn't necessarily mean that you need to lower the price. Users who wishlisted your game could have various reasons not to buy it, no matter what the price is. For example:

1) They might be traveling.
2) They might have a backlog of games they haven't started playing yet.
3) They might be too busy with other life activities.
4) They might be struggling financially.
5) They might have ignored the email notification about your sale.

That's why it could be a good idea to repeat the same discount multiple times before moving to lower prices. You want to avoid looking desperate, so don't devalue your own game in the eyes of your potential customers.

Updates

Depending on your game's genre, you may come up with updates that offer more content, additional languages, controller support, multiplayer, and other features. Releasing regular updates is a great way to keep your game in the center of attention. Major updates that add content and new features can result in press and influencer coverage.

To get an exposure boost, you can utilize Steam visibility rounds. It's a useful feature that allows you to announce major updates to your customers as well as users who wishlisted your game. Visibility rounds will place your game in a dedicated spot on the Steam front page as well as list it in the "Recently Updated" section. It will remain featured for up to a month, depending on how large your customer base is. Since Steam lets you use this feature only a couple of times, save visibility rounds for major updates only.

DLCs

If your launch went well and there's much demand for more content, consider making a DLC. It could be a new level, a new character, or some significant upgrade that adds value to the base game. DLCs have separate store pages and marketing materials, so launching a DLC is not much different from launching a game. A successful DLC launch will come with press coverage, video playthroughs, and live streams. Producing a DLC that you can charge for takes time and money, so do that only if your customers show a lot of interest in buying potential expansions.

Soundtrack

As of writing this book, Steam considers soundtracks a special kind of downloadable content. It can be purchased without having to own the base game. If your game features professionally written music that is adored by

your players, consider selling it. Just make sure that your agreement with the composer gives you the right to do that.

Participate in competitions

While winning awards won't have a significant impact on your sales, it will give your studio long-term PR value and notoriety. Submit your game to every festival and competition you can find. There are many awards you can be nominated for: Bafta, Game Awards, Independent Game Festival, Game Audio Awards, and many others. Some involve a paid application process, while others simply nominate you if your game gets noticed. Either way, do your best to get your game nominated for awards. Hopefully, you can win some too.

Update your marketing materials

Now that your game has been out for a while, it's time to update your marketing materials. You can show off short snippets of reviews in your trailer and create new teaser trailers showcasing quotes, awards, and user review scores. The important thing is to keep your store page looking fresh and up to date. Nobody wants to buy a game that looks abandoned by its creator.

Stay active on social media

It will not be as easy as before to come up with original content for your social media channels. It would seem like there's nothing more to say—after all, the game is now finished. However, don't hesitate to repost some of your old GIFs or create new ones. There's still an enormous number of people out there who have never heard about your game. Don't stop

marketing your game just because it’s been released. It’s just the beginning of your game’s life.

Giveaways and contests

Running giveaways and contests on social media can be an effective way to attract attention to your game. Everybody likes the idea of winning a free game, so don’t hesitate to ask for something in return, for example, sharing or retweeting your original post.

Other platforms

Since porting can be costly and time-consuming, make sure you start with the platform that is the most popular among your fan base. Post a poll on your social media; the results will help you determine what platform should be your priority.

As described in Chapter 7, porting is a complex process. Consider delegating it to a port house or a publisher while you continue marketing your game on the PC platforms.

Merchandise

I hope you still own the intellectual property rights to your game. If you do, now is a good time to start selling merchandise. There are a lot of services that allow you to design merchandise. I use **Teesping.com** and quite like it for its simplicity. You can have your merchandise store up and running in a matter of hours. There’s a lot of different kinds of merchandise to choose from, so you can find something that fits the theme of your game. The service lets you be in charge of designs and pricing. They take care of manufacturing, payment processing, and shipping. Merchandise is produced only when it’s ordered and paid for, so you don’t have to buy anything in

bulk. While it's not the most profitable way of doing it, it doesn't take much effort to set up.

If there's much demand for custom-made merchandise, consider using **Merchoid.com.** It's a company that produces custom merchandise for established video games. It's more than just t-shirts and mugs. They actually make custom toys, figurines, and other unique items.

Bundles

Don't even think of participating in bundles until your game has been out for *at least* a year. Once you notice a significant drop in sales, consider reaching out to one of the key retailers I mentioned in Chapter 7. Getting your game into a bundle can generate more revenue and spread the word about your game. Since bundles are cheap, a lot of people buy them. While you might not make a lot of money, at least you'll get a boost in exposure.

Permanent price decrease

When your game gets old and its revenue drops as a result of newer titles entering your niche, consider a permanent drop in price. While it gives you less room for discounts, you have to do what it takes to stay competitive.

Licensing

If your game sold well, you could license your IP to other companies to produce derivative works. For example, if you don't want to deal with a mobile port yourself, why not sell a license to a big mobile publisher so they can make a small mobile game based on your IP? If you already have a sizable customer base, it could be a lucrative deal for both parties. You can make similar offers to book writers, comic book artists, etc. Even if such

deals don’t result in a lot of money, new derivative content will keep your IP alive and extend the lifespan of your game.

Projected revenue

Having an estimate of your future revenue gives you the freedom to plan ahead. Knowing what to expect during the first year of sales can help you come up with the strategy to keep your game relevant.

As mentioned before, your game can become wildly successful even if your launch didn’t go well. Having said that, the first week of sales usually serves as a good indicator of your game’s performance in the future. Jake Birkett, the developer of *Ancient Enemy* and *Shadow Hand*, came up with a formula that estimates a game’s annual sales based on its first week’s performance. According to the formula, your first-year revenue will equal your first-week revenue multiplied by five. The formula resulted from analyzing the sales data of 14 indie titles. While it’s a relatively small sample size, many indie developers confirm that the formula is quite accurate. As of writing this book, *DARQ* hasn’t hit its one-year mark yet. However, it looks like *DARQ*’s revenue will exceed the projections.

Sequel

Making a sequel of your game could be a great idea if there’s a demand for it. There are many advantages to making a sequel of a successful title:

You already know how to make this kind of game.
This time, it’s going to be a lot easier and faster. You won’t have to learn and experiment as much. You have already discovered what works and what doesn’t.

You can reuse some of your code and assets.
Not having to write miles of code from scratch is a big advantage.

Perhaps you can reuse some of the game assets, like the protagonist character or some of the environments.

You already have the customer base.
Your social media followers and Steam customers already like your game. You won't have to work as hard to build a new audience from scratch.

While there are significant advantages to making a sequel, you should also consider that you may not sell as many copies as you did the first time around. Some portion of your audience probably wouldn't spend their money on a game that provides a very similar experience to the original game they already played. If you were to sell fewer copies of the sequel, you have to make sure you can get it done significantly faster than the original game. Even with fewer copies sold, it might still be a valid business idea.

New IP

Think of ways you can make your life easier. Your biggest asset right now is your audience. Since they bought or wishlisted your first game, they probably like games in this genre. Can you think of a fresh idea for a game in the same genre? Try to make a completely new game that reuses some of your old code. If you're able to make it appeal to your current audience, that could be your next project.

Buying IP Rights

One of the ways you can grow your studio's notoriety is by acquiring a license to a pre-existing IP that already has a large audience. Think of books, movies, and other works of art that could be turned into a video

game. While some works may be in the public domain, others would require purchasing a license.

> *Blair Witch was the first pre-established IP that the Bloober Team and I worked on. Not just established. It's legendary. It's an amazing opportunity, but also a huge challenge. How do you adapt an IP from the silver screen (or any other medium) to a game without losing its character? Our approach was simple in concept: focus on recreating the experience. People don't remember specifics like cinematography, sound editing, or visual effects. They remember how those things made them feel. You can't just repeat all the tricks from the original. You have come up with new ways, more suitable for a game, that recreate those emotions. Keep this mindset while making every design decision and you'll create a game that not only bears the name of a franchise but also feels like it's an integral part.*
>
> **Barbara Kciuk** | Narrative Designer
> *Blair Witch*

Key Takeaways

Update your game frequently, patch bugs, and stay in touch with your community.

Consider releasing a series of DLCs to keep your game in the spotlight.

Don’t abandon your social media. Stay active and keep posting content about your game.

Port your game to other platforms to maximize your revenue.

Create a merchandise store and try making IP licensing deals with other companies.

If there’s enough demand for it, make a sequel of your game. Making a sequel is a lot easier than creating a new game from scratch.

If you’ve decided to make a new game, try to utilize some of your old code to save time.

Consider acquiring a license to an established IP, so you can create a game that already has a large audience.

Afterword

While game development is a business and should be treated as such, it's also an art-form and a way to connect with people. Similarly to a painter or a composer, most of your development time will likely be spent in solitude. However, once your creation comes to life, it will become your way to connect with your audience on a deep and personal level. Behind every line of code you write, there's a potential to touch somebody's heart. Behind every polygon you create, there's an opportunity to impact somebody's life. While some people might see your game as another form of entertainment or distraction, others will remember it as one of the most moving experiences of their lives. Don't underestimate the amount of impact your game can have. Even if it has a relatively small player base, thanks to content creators and streamers, it can be seen by millions.

Ultimately, it all comes down to connecting with other human beings. Video games have become a universal language used by people around the world as a means to connect. As you gain more experience, you could become "a poet" of that language. Yet, even if you write beautiful code and make your game look amazing, what is your message? Unlike larger studios who have shareholders to please and loot-boxes to monetize, you have a unique opportunity to connect with your community in a truly personal way. Tell them something that matters to you. Communicate it through your game and in any other way you can. Connecting with your audience and having the ability to improve their lives even in the smallest way is the ultimate reward of an independent game developer.

In the end, the most fulfilling aspect of my journey with *DARQ* was the honor to serve my community. Offering free DLCs was one of the ways to communicate that. While *DARQ* didn't have a global impact on the world, it surely touched the lives of many. To some, it became an unforgettable experience that will forever be remembered as meaningful. At least, that's

what some of the user reviews say. Making a difference in people's lives is what drives me. Writing this book is another attempt to do just that.

Thank you for choosing to connect with me by getting to know my story. I hope this book inspires you to become a designer of your own game, as well as your own life. I'm looking forward to connecting with you and hearing about your journey. You can reach out to me on Twitter (@unfoldgames) or email me directly at info@unfoldgames.org. I'll be happy to guide you on your path to becoming a game developer to the best of my ability.

In every chapter of this book, I tried to bring you as much value as possible, and I truly hope this book will have a positive impact on your life. **If you enjoyed this book, it would mean the world to me if you took a minute to write a review on Amazon. Even a short review makes a difference**.

If someone you know is interested in game development, please send them a copy of this book. Whether you gift it to them on Amazon or simply email a copy of the PDF, it will make me happy either way.

Finally, if you'd like to get free bonus content that can help you during the development of your first game, consider signing up for my newsletter at GamedevBook.com—you'll receive 100 game design tips and other free content that didn't fit into this book.

Acknowledgments

In my relatively short life, I've been challenged by incredible hardships and seemingly impossible obstacles. I wasn't expecting help from anyone, yet I was lucky to have been given a chance by one of the most renowned composers in the world—John Corigliano. He was the first to believe in me, and for that, I'll forever be grateful. That's why I dedicate this book to him.

In times of hardships, it's important to have family to lean on. While I'm technically an orphan, I would like to thank my brother Damian and my sister-in-law Marina for being there for me and encouraging me to pursue my dream, no matter how unattainable it seemed at first.

I must also thank my dear friends and collaborators, Chris Vick (*DARQ*'s producer), Richard Gale (creative consultant), Bjørn Jacobsen (*DARQ*'s sound designer), as well as the talented contractors who contributed to the creation of the game.

Interacting with the *DARQ* community on a daily basis has been incredibly motivating to me. I can't thank all the *DARQ* fans enough for their support during the wildest journey of my life. All the support the game received during launch means the world to me. I'll always be grateful for it.

DARQ wouldn't have reached its level of success if not for the lessons I've learned from other developers and industry professionals. While I got to meet some of them in person, others taught me valuable lessons through tweets, articles, and GDC talks. It's for this reason I asked them to share snippets of their knowledge and experience throughout this book. I would like to express my deep appreciation to Mark Kern, Quentin De Beukelaer, Piotr Babieno, Scott Millard, Barbara Kciuk, Wojciech Piejko, Michał Mielcarek, Dan Adelman, Austin Wintory, Rami Ismail, Oskar Stålberg, Jason Rohrer, Jake Birkett, Stephen McArthur, and Toms Martin.

Before I became a developer, I got to write the music for the video game called *A Cat's Manor* by Tariq Mukhttar. Scoring his indie creation was pure joy. I was amazed that he had made the game all by himself. That's

when I got the idea to try game development as a hobby. Little did I know that it would lead me to discovering my life’s passion. I would like to thank Tariq for that.

I also want to give my compliments and gratitude to my literary collaborators, Sheree Haley, and her talented son Brendan Haley. Since English is my third language, this book wouldn’t have come to fruition without their generous contribution. Apart from being an editor, Sheree also happens to be an accomplished graphic designer who created the logo for my studio, Unfold Games.

Finally, I want to thank my partner, Kelsey, who has been with me through the thick and thin of this turbulent journey. I don’t know what I would do without her.

Resources

As a bonus, I would like to share with you my favorite resources that helped me along my journey. I hope they will help you as well.

Websites:

- **IndieDb.com –** One of the largest communities for indie developers. You can list your game here, upload screenshots and videos, create a press kit, connect with other developers and write articles about your game. The most popular articles get retweeted on their active social media account, so it's a great way to start getting attention for your game.
- **TigSource.com –** One of the largest forums for indie game developers. You can ask questions here, find collaborators, and post a development blog.
- **Reddit (r/gamedev) –** If you like Reddit, you should definitely become a part of this community. There's always something interesting to read. It's a good place to learn about game development. It's *not* a good place to show off your game though, so don't try marketing it here.
- **Gamasutra.com –** A large website featuring video game industry news and developers blogs. You can learn a lot about the industry here.
- **GamesIndustry.biz –** An incredible resource for learning about video game business. You should visit it daily; there's always a lot to learn.
- **NewZoo.com –** The world's most trusted source for games and esports analytics and market research. A free account allows you to access a lot of industry data when performing market analysis.

YouTube Channels:

- **YongYea –** One of the biggest channels for video game industry news. Watch it daily to stay updated on the constantly changing market of video games.
- **Brackeys –** The biggest channel for Unity tutorials. It's mostly for beginners, so it's a great place to start.
- **Game Maker's Toolkit** – Features lots of incredibly helpful videos when it comes to learning game design. As a first-time developer, you should watch as many of them as possible.
- **Jonas Tyroller** – A personal channel of Jonas, an indie developer known for *Islanders*—a successful indie title made by a small team. Jonas has a lot of entertaining videos that cover a lot of game development topics. As of writing this book, Jonas is developing his next game, *Will You Snail*. He uploads a lot of behind-the-scenes videos that provide helpful insight into the life of an indie developer.
- **Sunder** – If you want to learn level design in an entertaining way, this is the channel. Sunder provides detailed level-design analysis of various games. There's a lot to learn here.
- **Code Monkey** – A great resource to learn Unity and programming in C#. The channel is full of useful tutorial videos made by an indie developer, known for such games as *Ninja Tycoon* and *Hyper Knights*.
- **Thomas Brush** – A fantastic channel by an indie developer known for his critically acclaimed games, *Pinstripe* and *Neversong*. His channel features a lot of behind-the-scenes videos that are a must-watch for a first-time developer. Many of his videos offer valuable insight into the creation of 2D art and animation. Thomas also provides general advice on game development and entrepreneurship.
- **GDC** – Game Developers Conference is the world's largest professional game industry event. Their YouTube channel features lots of educational lectures on various topics by industry professionals. While the most recent content is accessible only through paid membership on their website, you can still learn a lot

from the older videos listed on their YouTube channel. The talks feature such topics as coding, animation, game design, level design, publishing, distribution, marketing, and others.

Books:

- **Blood, Sweat, and Pixels**, by Jason Schreier**.** It's a must-read collection of eye-opening interviews with 10 developers who explore the harsh reality of making video games. There are 10 games featured in the book, both triple-A and indie. My favorite chapter, the one I think you would enjoy the most, is about the creation of *Stardew Valley*—a wildly successful indie game made by Eric Barone. *Stardew Valley* was his first game, and he made it completely solo. There's a lot to learn from his testimony. Reading Jason Schreier's *Blood, Sweat, and Pixels* inspired me to share *my* story and ultimately write this book.
- **Maximum Achievement**, by Brian Tracy. I was lucky to have read this book as a teenager. It had a profound effect on me and allowed me to build a mindset that helped me achieve success in multiple industries, despite all the hardships and obstacles I had to face.
- **Startup CEO**, by Matt Blumberg. If you're new to the world of business, you have to read this book. As a first-time developer, you'll likely wear many hats, one of which includes being a CEO of your company. This book will teach you basic business skills and terminology you will need to run your company effectively.

Thank you for reading.
WLAD MARHULETS

Copyright

Unfold Games, LLC
5042 Wilshire Blvd, #39434,
Los Angeles, CA 90036
USA

FIRST EDITION

Foreword: John Corigliano
Editing & Logo Design: Sheree Haley
Editing: Brendan Haley
Proofreading: Nancy Haight
Cover Art: Kelsey Haley

ISBN: 978-1-7352325-1-5 (Kindle)