Super Fish Quest: A Video Game

Melissa Wiener

Western Oregon University

Follow this and additional works at: https://digitalcommons.wou.edu/honors_theses

Part of the Software Engineering Commons

Recommended Citation

Wiener, Melissa, "Super Fish Quest: A Video Game" (2012). Honors Senior Theses/Projects. 87.
https://digitalcommons.wou.edu/honors_theses/87

This Undergraduate Honors Thesis/Project is brought to you for free and open access by the Student Scholarship at Digital Commons@WOU. It has been accepted for inclusion in Honors Senior Theses/Projects by an authorized administrator of Digital Commons@WOU. For more information, please contact digitalcommons@wou.edu, kundas@mail.wou.edu, bakersc@mail.wou.edu.
Super Fish Quest: A Video Game

By

Melissa Ann Wiener

An Honors Thesis Submitted in Partial Fulfillment of the Requirements for Graduation from the Western Oregon University Honors Program

Dr. Scot Morse,
Thesis Advisor

Dr. Gavin Keulks,
Honors Program Director

Western Oregon University
June 2012
Abstract

Video game design isn't just coding and random number generators. It is a complex process involving art, music, writing, programming, and caffeine, that should be approached holistically. The entire process can be intimidating to the uninitiated programmer, which is why I've written an all-inclusive guide to game design. With the creation of my own original video game, Super Fish Quest, as a model, I analyze each part of the design process, discuss the technical side of programming, and research how to raise money and publish a game as an independent game developer.
Table of Contents

1. Prologue
2. Introduction
3. Part 1: Design
   - Genre
   - Perspective
   - Markets
   - The Idea
   - Game Characters
   - Environment
   - Game Dynamics
   - Stories and Narrative Elements
   - Descriptive Elements
   - General Design
   - Official Design Document
4. Part 2: The Programming and Creation
   - The Programming Language
   - Animation Framework
   - Ribbons, Bricks, and Sprites
   - Collision Detection
   - Sounds
   - Organization
   - Problems, Difficulties, and Future Development
5. Part 3: Indie Publishing
   - Publishing Platforms and Social Networking
   - Raising Money
6. Conclusion
7. Epilogue
8. Appendices
   - Appendix A: The Official Design Document
     - Story Outline
     - Characters
     - Level Design
     - Game Mechanics
   - Appendix B: Class Diagram
   - Appendix C: Screenshots and Design Process
9. Bibliography
Prologue

“Melissa! Wake up! Your thesis has been kidnapped!”

“What... what?!”

“The turtles! The turtles stole your thesis and took it to the castle!”

So began the strangest journey of my life.

Introduction

I hold to the fervent belief that one should have fun with everything that one does. There are times and places for seriousness, but I chose not to write a dry, academic research paper (though those who do choose to do so ought to be commended. Bravo). Instead, I chose to orient my thesis around a particular hobby of mine – video games. My original intention was to completely develop a fully functional, fun, and original game. This would include planning the schedule, writing the storyline, drawing the artwork, creating the music, and programming it all myself. Although that is still my intention, the project itself is regrettably unfinished
in time for presentation. What follows instead is a much smaller version of what I had in mind.

I have created a demo of my game, dubbed Super Fish Quest, and written up a holistic guide to game development – including the designing and programming of Super Fish Quest, as well as publishing in the indie (independent) gaming industry. Sit back, strap in, and prepare yourself.

**Part 1: Game Design**

Super Fish Quest began as a tongue-in-cheek idea to parody the illustrious Super Mario Brothers. It is the story of one reluctant cat named Hero who must rescue the kidnapped fish princess from an unknown villain. Hero's motivations are not for glory or conventional heroism at all, but rather, he is, for some arbitrary reason, the only one who can save her. What the fishy proletariat don't know, is that Hero, a restaurant dish washer and would-be chef, is only answering their cries for help because he secretly wants to fry up the fish princess, perhaps in a nice white wine and garlic sauce...

Hero travels from kingdom to kingdom, searching for the villain who kidnapped the fish princess: his key to culinary fame. Sadly, he is met over and over again with the frustrating message: "I'm sorry, but your fish princess is in another kingdom." At which point, Hero eats the messenger and continues on his quest,
growing fatter all the while.

But an idea does not make a whole game. What follows in an in-depth look at each aspect of game design described by Lecky-Thompson (2008), and how I did it with *Super Fish Quest*.

**Genre**

When describing a video game, we first define the game by its genre. Unlike other works of fiction such as literature and films, a video game's genre is independent of its setting. Different games can be set in fantasy worlds, outer space, or a modern city, and still be an action game by genre. In video games, the traditional literary genres instead become sub-genres. This is because the player interaction is a higher operator in terms of audience reception than the story told by the author – something that isn't found in literature or films. The genre describes what there is to know about the game – its perspective, its goals, and its general gameplay feel. For example, take the well-known game *Super Mario Brothers*. *Super Mario Brothers* is a side-scrolling, platforming game. Anyone who is familiar with video games will instantly know that a platformer means the game is played by controlling the character as it jumps to and from suspended platforms or over obstacles. The goal of the game is to traverse across the level and avoid death. Video games are not limited to one genre; many games mix genres to create a new and unique gaming experience.

There are many different game genres. Many of the first mainstream video
games, released on arcade machines, such as *Space Invaders* and *Pac-Man*, are action games. The action game’s emphasis is on quick reactions, fast-paced action, concentration. Today the action genre is most often merged with another genre to create a deeper experience than that of an arcade game. Another early genre is simulation games. Simulation games arrived in arcades as racing simulations with rally cars and motorcycles. Other genres include (but are not limited to) sports simulations, adventure games, role-playing games, puzzles, and the aforementioned platforming and action games.

In the early planning phases of *Super Fish Quest*, I decided that I wanted to harken back to the good old days of 2D platforming (a game where a character can only travel on an x or y axis on the screen). The genre, especially in two-dimensional form, is no longer produced much now, except for by indie game developers. It is my belief that comparatively simple platforming games can be just as much (if not more than) of a rewarding and fun experience as a big budget, high-graphics modern game such as *Mass Effect 3*.

**Perspective**

The playing perspective is a combination of a player’s position in the game universe coupled with the way that they are allowed to perceive that universe. The perspective can change or stay the same throughout the game. In many cases, the perspective can make or break the game – a poor perspective choice can make the controls frustrating or even unplayable.
The first component of the playing perspective is the view offered to the player’s character. If the character is an individual, then the individual can be viewed either in the first person or the third person. The difference between these two perspectives is significant: a first person perspective may be more immersive, but the player may feel restricted by not being able to see a very large portion of the scene. Third person views can show more of the surrounding scene to the player. The combination of the first and third person views is the “over the shoulder” viewpoint.

The second component is the playing angles, or the camera. For the most part this is only relevant in third-person three-dimensional games. The most common playing angle for 3D games to date has been isometric – a top-down 45-degree view of the playing area, but other views include side views and straight-down. The camera needs to be able to follow the character’s position so as to give the player the best possible view at all times. It can often be useful to have player-controlled cameras instead of stationary cameras, but beware: poor controls or stationary views can ruin the game experience.

Because Super Fish Quest is a 2D platformer, the perspective is simple, and tried-and-true. The character is viewed in the third person, and the camera gives a direct side view of the scene.

Markets

Choosing a target market is also important when designing a video game. The
goal is not to satisfy a particular niche, but to appeal to as wide an audience as possible. Many different factors come into play when choosing a market.

One factor to take into account is the attention spans of your audience. In the days of the arcade machines, gaming experiences were quick, and the player was limited by how much money they wanted to spend. Game designers had to take this into account – after all, the goal is to get the player to keep putting more money into the machine. Now, with home consoles, the gaming experience is no longer so time-limited. Developers thus have a choice: appeal to the “casual” players who just want a quick fix, or the “hardcore” who are willing to play the game for a long time. Games geared towards those with a long attention span include games in the Elder Scrolls or Fallout universes, whereas those aimed towards the quick-fix camps may be more like arcade games in nature, such as *Guitar Hero* or *Dance Dance Revolution*.

Having a puzzle element in your game can be a great way to draw in a larger market. Games such as *Tetris* are solely puzzle-oriented, and appeal to their own special audience. However, including puzzle elements in an adventure game can make your game appeal to a larger audience, so long as they don’t interrupt game flow – it is a factor with more or less universal appeal.

Another factor to consider is violence. Violence in video games has been a hot topic in the media since the 1990s with games like Mortal Kombat. As graphics improve, so have depictions of violence. Excessive gore or violence may be distasteful to some audiences, and violence for the sake of violence may even give
the game an undesirable reputation. It is also an important factor when considering your target market’s age range – excessive gore may not be suitable for small children.

One final factor (though there are countless more) is in choosing multiplayer versus singleplayer gameplay. Multiplayer gameplay has evolved to take advantage of high-speed internet connections, allowing you to play games online with other players across the globe. This could be a one-on-one match, a round with a small group of friends, or even an expansive world with thousands of other players all connected at the same time. Of course, with many games you can still connect multiple controllers and play with your friends at home. The multiplayer market is one that may buy a game just because it has multiplayer, but you may also shut out the market that has no desire to play with other people. Many games offer single and multiplayer modes, to suit both markets’ needs.

Typically platformers such as Super Fish Quest appeal to the same sort of audience every time. It is a fairly large audience – gameplay appeals to both long and short attention spans, puzzles can be easily incorporated but are not necessarily the focus, violence is usually low or nonexistent, and the games are typically singleplayer. Super Fish Quest is meant to appeal to this general audience, but this does not mean that the audience cannot change between other 2D platformers. Trine and New Super Mario Brothers both offer a multiplayer mode, Braid’s gameplay focuses on very challenging puzzles and incredible artwork, Super Meat Boy offers
more gore than your average platformer (your character is quite literally a meat boy, who leaves trails of blood behind when he runs), and *I Wanna Be The Guy* is meant to be incredibly difficult. Each of these appeal to new audiences beyond the “typical” platformer market.

**The Idea**

There is no rule about what makes a good idea for a game. An idea can come from anywhere – and if you don't have an idea, you can take an existing idea and add a twist. The goal when coming up with an idea is to have a hook. A hook phrase is simply how you can describe your game in one sentence – and hopefully intrigue the audience. Designers wanting to clone an existing title should find something that they can sum up in one sentence that makes their game different from other games.

*SUPER FISH QUEST* is in many ways a clone of *SUPER MARIO BROTHERS*, and intentionally so. My hook is not in the gameplay, but in the story: the hero wants to save the fish princess not out of chivalrous intent, but in order to eat her as a delicacy.

Simply having a hook phrase is not enough – now the idea must be refined. A document must be written that refines the idea into something more concrete. The developer must look deeper into each area of the idea, such as level design, dynamics, placement of weapons, story, and so on.
**Game Characters**

Creating good characters is one of the most important parts of game design. Every character should have depth and a role to play in the game. The main character is of course the most important – usually this is the hero of the story, but this is not required. The main character could be the adversary. In *Super Fish Quest*, our intrepid hero is an antihero. In literature, an antihero is a protagonist whose character is not in line with the archetypal hero. Similarly, an antivillain is an antagonist who elicits sympathy or admiration... but discussing the antivillain in *Super Fish Quest* would be spoiling.

At the end of the day, a well-developed character should have a set of defining characteristics that stick out, and are immediately called to mind when the character is seen. For this reason, a character's background information should be “rich but simple”. Additionally, players should be able to relate to the characters. They should care about the hero and feel threatened by the enemies.

**Environment**

The game’s characters must inhabit the game environment. In *Super Fish Quest*, the environment is the set of kingdoms that the hero travels through in search of the fish princess. Each kingdom is different in order to have a rich and varied game universe. Our hero travels from the Grass Kingdom, through jungles and deserts, to an industrial city, and more. Each environment has its own set of enemies.
and special dynamics. Though the environment is not as important as character
development, there should still be significance and a back story to the world your
characters live in.

**Game Dynamics**

Game dynamics are a large part of what makes a game fun and unique to play.
At a certain point, characters and story don't matter so much if every game you play
of that type feels exactly the same. Game dynamics are all the different ways in
which the characters interact with the environment. The hero's speed – how fast he
can run, is one such dynamic. A character's abilities and actions are another
dynamic. Many game dynamics can change throughout gameplay, which can keep
things interesting and challenging for the player.

Game dynamics, like the perspective, can make or break a game. Poor, buggy,
or flawed game dynamics can make a game frustrating and virtually unplayable.
Developers should note that quality game dynamics can be more important than
graphics and special effects. Games that are graphically impressive but poor to play
are not unheard of now, especially as modern home consoles compete with each
other to have the best available graphics.

**Stories and Narrative Elements**

Stories are more important to some games than in others. Puzzle games do
not need any story at all. Others, such as role-playing games, are heavily story-
oriented. The best role-playing games create a rich and expansive world, with many different stories and outcomes that are discovered through quests and missions. Usually, the story is a backdrop to the rest of the game, and the game should appear to grow out of the story. *Super Fish Quest* is meant to be more heavily story-oriented than the average platformer.

Every major character needs to have a place in the story. The world as a whole should have a back story, as should each character. Giving each character a back story helps to develop and flesh out their character – nobody wants an unbelievable and bland hero or villain.

Narrative elements should also be spread throughout the game. These are what propel the story forward and give the situations depth. There can be smaller narratives hidden in the game, such that only players who explore the whole world will find such secrets, but the main narratives should be placed in areas where the character is forced to go in order to further the storyline and get further in the game. In a platformer, these narratives will take place in between levels and worlds.

The narration can be performed through speech, pictures, or text. As a general rule, if the narrative is really important, it should be delivered via semi-permanent text that the player can refer to again at a later time. Not having this ability can be alienating for the player if they are not paying complete attention, or if they put down the game and come back to it later, and forgot what they were supposed to do next.
**Descriptive Elements**

Descriptive elements are different from narrative cues. Descriptive elements apply to objects in the game world; they indicate the object’s worth, age, or use. Game objects with real-world equivalents are generally recognizable to the player, but descriptive elements are even more important for objects that arise from imagination and are new to the player. Objects can be vehicles, weapons, things, tools, food, keys, and more. They should be recognizable for what they are, even if the player has not encountered them before.

Some objects are distinguishable from their texture, such as wood, stone, metal, and plastic. If texture is not enough, color will add additional clues. A pink sword blade, for example, would not make sense. Sounds can also be descriptive clues. When picking up an object, a sound effect may play that indicates what kind of item it is – coins and trinkets will jingle, paper will rustle. Groups of items can have certain sounds associated with them after a time, so not every sound needs to be exactly accurate to what the object may sound like in real life.

**General Design**

The general design issues are the meat of the game structure. Most games have some combination of a scoring system, levels and missions, lives, and unlockables.

Scoring is needed so that players have a way to measure their success. Simply
reaching the end of the game is not enough for some players – they need to know
how well they played the game. In Super Fish Quest this is done by collecting points
from gathering items and beating enemies. Other games may have players collect
money, which can in turn be used to buy new items to enrich the gaming experience
and increase avatar strength – that is, make your character better, faster, stronger...
or simply better looking in the case of cosmetic upgrades.

Levels or missions of increasing difficulty is a structure that is commonly
implemented in games. Players should feel consistently challenged as their playing
skills increase with practice. Lives are another measure of success or failure. If you
die, you lose a life. If you do well, you may earn an extra life. If you lose all your lives,
the game is ostensibly over – many games offer multiple or even infinite continues,
restoring some of your lives and allowing you to keep playing.

Finally, unlockables – these allow you to “unlock” or buy optional features,
such as extra characters, objects, abilities, or secrets. Unlockables are not necessarily
items that have any tangible effect on gameplay. Sometimes they are simply cosmetic
– for example, a player can unlock a shiny new hat for their character to wear.

The Official Design Document

Once the game is designed, an official design document is written. The
purpose of the official design document is to sell the game idea to prospective
financial backers as well as to provide a template to build the game on. It should be
well thought out and comprehensive, yet concise, and it should get the main ideas
across early on in order to capture the reader’s attention and keep them interested.

The official design document includes both a paper version with all the information and a CD that contains artwork, a real prototype, and other relevant assets such as audio tracks and video clips. The document starts with an overview page that is essentially an advertisement. The next pages contain the story – the plot, character descriptions and histories, level descriptions, and game mechanics. When pitching your game to prospective financial backers, the document should also include a section on planning (time and resource estimations and requirements) and an overview of the team working on the game. Because Super Fish Quest is a personal project, I have not included these last two sections in my official design document.

See Appendix A for Super Fish Quest’s official design document.

Part 2: The Programming and Creation

The real meat of game development process is the programming, followed closely by the artwork, story, and music. The programming is also the most time-consuming and intimidating part of the process, and I learned a lot in what I accomplished. The most important thing I learned is that no matter how complicated you think the programming will be, in reality it will be ten times more
complicated. What follows is an overview of the programming process and overall organization of Super Fish Quest.

**The Programming Language**

When I started planning for Super Fish Quest, the only programming language I had any experience using was Java. Having never attempted to make my own video game at that point, I figured that using the language I had the most experience with was a good idea at the time. In retrospect, I wish I had done more research before choosing a language.

Java is a fine language for game programming. It is a powerful high-level language, and many games have been coded in Java, for example, Minecraft. It was first released in May 2009 as an alpha version and has since sold over 5 million copies (Persson, 2012). However, it is not the most popular language for game programming, one reason being that it is not supported by game consoles. My main issue was not so much in choosing Java, but in not choosing a Java game library¹ or API². Instead, I used the files given in Killer Game Programming in Java, taking their example 2D sidescroller and attempting to base my game off of that. The example game given in the book is extremely primitive, and I spent countless hours modifying the code to fix fundamental design flaws.

The next iteration of Super Fish Quest will most likely be using LÖVE, a

---

¹ A programming library: collection of resources used to develop software, including pre-written code.
² API (Application Programming Interface): a specification intended to be used as an interface by software components to communicate with each other.
framework for making 2D games with Lua (a lightweight scripting language).

**Animation Framework**

At the core of any game is the animation framework. That is, the game must update constantly and re-draw the screen to reflect changes. In *Super Fish Quest*, the animation framework is embedded in the GamePanel class – a subclass of JPanel, Java's container in which the game window is displayed.

The animation loop is handled by a thread – a process of execution. The Java Virtual Machine allows an application to have multiple threads of execution running concurrently. This allows the developer to have animation handled by one thread, and everything else handled by another, so that everything runs efficiently. In the current iteration of *Super Fish Quest*, there are two threads – the animator, and an event processor.

The main game loop is handled in the run() method in the GamePanel class. In simple terms, it goes like this:

- If the game is running, and not paused, check for and update movement.
- Re-display the graphics as per the updated movement.
- Repeat

However, simply implementing these basic steps without any other safety measures may cause the animation to be too choppy (skipping animation frames) or too fast. In *Super Fish Quest*, I use a double buffering technique. Instead of updating the
graphics directly to the screen, the game is rendered to a buffer – an area that stores data temporarily. The buffered image is then painted to the screen. This technique reduces on-screen flickering. My code also handles the speed of the animation loop – without handling it explicitly, the speed would be dependent on how fast the computer is, meaning that more powerful machines may go through the animation loop too quickly. This is handled by using the Thread.sleep( ) method – which simply pauses the thread's execution temporarily. The end result is an animation framework that runs smoothly and consistently between machines.

**Ribbons, Bricks, and Sprites**

After getting the basic game loop and animation framework set up, my next step was to put together the background. The code given in *Killer Game Programming in Java* has a framework set up for what they call ribbons and bricks.

A ribbon is simply an image that must be wider than the game window. The image is “wrapped” around the screen so that it loops, creating a continuous background. I used three ribbons, one for mountains, and two separate ones for clouds. Each ribbon has a different move factor, so that as you travel across the screen, the mountains go by slower than the foreground (the character and the bricks), and the clouds go by even slower than the mountains. I used two separate ribbons for the clouds and gave them different speeds so that eventually the clouds will overlap and pass each other, giving a more active feeling to the background.
The bricks are foreground images, and they are the platforms that the hero jumps on, over, and to. They are made out of a single image file of a strip of bricks – the position of each brick is separated out into an array, so each brick has an associated numerical index. Using these indices, I am able to draw out a map of where each brick should go:

Illustration 1: The hills ribbon loop

Illustration 2: A small portion of a brick map

In this case, 6 is the index for the ground block, 8 for the blank block, and other digits for various colors.

A sprite is a moving graphical object. Every frame of animation is stored in the same image file. The program saves the locations of each frame and cycles
through which portion of the image is shown at a time. Using one image file instead of several saves time and resources in loading the files. In *Super Fish Quest*, the hero is pictured with sprites. When running to the right, the sprite cycles through the images marked “running right”. When running left, it cycles through the images running left, as is for jumping right and running right. In a later release, there will also be images for animating attacks and deaths. When the hero is standing still, the sprite simply stops looping.

![Illustration 3: The "running right" sprite for Hero](image)

**Collision Detection**

Collision detection is a key part of any game. It’s how the character interacts with the game objects. It is signaled whenever two objects overlap. Collision detection keeps the characters from walking through bricks, determines when a character comes in contact with an enemy (causing damage), and when a character collects an item.

In *Super Fish Quest*, the implemented collision detection is for the bricks.
Every time the character moves, we test to see if that point in the game world is inside of a brick. If it is, then the move is aborted and the character stops moving. The same kind of testing is used when the character jumps – we must test to see if its head runs into the bottom of a brick platform. If there is a collision, the jump does not reach its maximum height, and the character falls back down to the platform below. Similarly, we must check to see that the character does not fall through the bricks from above. Each of these testing methods are located in the BricksManager class, with the methods insideBrick(), checkBrickBase(), and checkBrickTop().

In collision detection, we must take care that the collision detection is accurate. Each sprite or object has a bounding box that encompasses the whole image. However, most objects will not be perfectly rectangular. Thus, with poor collision detection, the game may think a collision has occurred because the two bounding boxes have overlapped, when visually, the two objects are not actually in contact. The better way to handle collisions is to test first by the bounding boxes, then by pixels. It is not good practice to do the entirety of your collision detection by pixels, because it is a slower method that requires more processing power. It could cause the game to lag if there are too many potential collisions to check. Instead, by only checking by pixels when the bounding boxes overlap, we are only checking when a collision is likely to happen anyway.
Sounds

There are two types of sounds in video games: background music and sound effects. Background music plays continuously and sound effects play whenever a specific action is performed – collecting an item or taking damage, for example. Background music and effects are handled separately so that two sounds can be played simultaneously. The background music can be played once, looped, or stopped. A sound effect plays once when it’s called, then stops.

Many different file types can be used, but I used MIDI and WAV files. MIDI stands for Musical Instrument Digital Interface, and is typically a small file size – perfect for quick loading. I used Anvil Studio 2012 to compose my own MIDI tracks, using a variety of electronic instruments and percussion. WAV – or WAVE – is short for Waveform Audio File Format. It can hold compressed or uncompressed audio. I
used Bfxr, an online tool for generating semi-random sound effects.

**Organization**

Each of these components, the animation framework, sounds, sprites, bricks, and ribbons make up the basic organization of *Super Fish Quest*. In some cases, all objects are handled by their respective base class, such as ribbons in the RibbonsManager class. In others, new classes are created as subclasses of a base class. For example, the Hero class is a subclass of Sprite. In programming terms, Hero *inherits* from Sprite – a Hero *is* a Sprite. Every method made available in Sprite can also be called from Hero, where the specific method can be overwritten if needed. Hero can also define its own methods, nonspecific to the Sprite class. For instance, a Sprite has a width, a height, an image, and a location (among other things). Therefore, a Hero also has all of these, but additionally has information and methods for which way it’s facing, if it’s moving, jumping, falling, or staying still.

Another type of class used in *Super Fish Quest* is a helper class. I created the class Mover that links together the ribbons, bricks, and Hero, to streamline programming movement, and ensure that when the hero moves, so does the scenery around him.³

**Problems, Difficulties, and Future Development**

The current iteration of *Super Fish Quest* is simply a demo due to the

---

³ For full class diagram, see Appendix B
difficulties I had in getting the basic gameplay to work. Building a video game from scratch is a very large, difficult, and complicated undertaking. There are some remaining issues to be worked out before continuing to add content:

- Hero sprite does not properly cycle through the animation when moving right and left. Jumping and falling work when not moving horizontally simultaneously.

- World coordinates of the bricks and drawn coordinates of the bricks are not the same. If you run into the edge of the map, then run back, the collision detection is off (running through the drawn bricks and running into invisible ones instead)

- Boundaries appear correct at first, then become wrong. If you scroll up, then back down, there becomes an extra area of space under the bottom bricks. Also does not display the rightmost bricks in the map.

- Can get the background ribbons off track when scrolling around the map diagonally.

I will be moving future development to a new platform – the 2D Game API LÖVE with Lua in order to have a simpler base interface to work with. Future development of Super Fish Quest and other projects can be seen at my personal website:

http://www.icewolf-designs.com
Part 3: Indie Publishing

Once a game is finished, a developer must publish their game and make it known. Today, more and more indie games are becoming popular, due in part to the growing use of online distribution.

Publishing Platforms and Social Networking

A major gaming platform used by PC gamers is Steam, a digital distribution and digital rights management service run by Valve Corporation. As of January 2012, there are over 1500 available games on Steam and 40 million active user accounts. Many major publishers make their games available on Steam as well as smaller publishers and individual indie developers. If your indie game is accepted by Steam, millions of gamers will see it advertised alongside highly anticipated big-budget games. Frequent sales on top of low costs and bundles (buying several indie games together for one low price) make these games highly attractive to users.

Xbox Live Arcade (XBLA) is another digital distribution service, for the Xbox 360, a gaming console made by Microsoft. As of May 2012, there have been almost 500 titles released for XBLA. Games are purchased with Microsoft Points, equating to between $5 and $20, and are commonly aimed towards casual players, with a “pick up and play” feel. There are some games that are available both on Steam and XBLA.

Android and iPhone applications are a growing market for smaller games,
each with their own app store to publish on. The wildly successful iPhone hit *Angry Birds* is one such game, with over one billion game downloads worldwide (1 Billion Angry Birds Downloads!, 2012). Both app stores offer free and paid-for games. Offering your game for free is a great way to publish a first game and get noticed – the goal in this case is not to make oodles of cash, but to get your name out there.

On that note, social networking can play a large part in making your game known. The popular online community Reddit has an indie gaming subreddit (a dedicated content area) with 18,000 subscribers, specifically for submitting news, articles, and your own game releases. The regular gaming subreddit is even more popular, with nearly 1.5 million subscribers, and free game keys always go over well there.

Finally, there are the Humble Indie Bundles – a series of game bundles sold and distributed online for a limited period of time and for a price determined by the buyer. The buyers also choose how to split the revenue between the developers, charities, and the bundle organizers. While this does mean a buyer can get five games for a penny if they want to, it is encouraged to pay more. To date, the fourth Humble Bundle has been the most successful, collecting $2,373,311.56 from 435,250 purchases (Prior Bundle Statistics, 2012).

**Raising Money**

Though you can still get funding for your game by pitching it to potential backers or holding bake sale upon bake sale, many developers are now using
Kickstarter. Kickstarter is a crowd funding website for creative projects – be it a video game, film, comic, music, or more. Members of the general public can browse projects and become financial backers in exchange for special goodies or discounts on the finished product. A product has one month to raise its chosen amount of money – if the full amount is not met, the backers pay nothing and the project is not funded.

Double Fine Studios, a relatively small game studio owned by Tim Schafer (the man behind cult classics Grim Fandango and Psychonauts), recently met incredible success on Kickstarter in March 2012. They pitched a project for a classic point and click adventure game and raised $3,336,371 – 834% of their original $400,000 goal (Schafer, 2012). Double Fine’s success has sparked a boom in Kickstarter projects and funding, bringing over 60,000 first-time backers to the site. As of March 2012, out of that 60,000, 22% have backed another project, pledging to 1,266 different projects and raising $877,171 for other projects – mostly in the games category, but every other category has also had activity from the new backers (Blockbuster Effects, 2012).

“Crowd-sourced fundraising sites like Kickstarter have been an incredible boon to the independent development community. They democratize the process by allowing consumers to support the games they want to see developed and give the developers the freedom to experiment, take risks, and design without anyone else compromising their vision. It’s the kind of creative luxury that most major, established studios simply can’t afford. At least, not until now.” –Tim Schafer, Double Fine Adventure Kickstarter
Conclusion

Creating a video game is no easy task. There's a reason most published and successful games are created at least by a team of more than one person. As I put further work into *Super Fish Quest*, I still hope to do most of the artistic and programming components myself, but I will probably get new music composed by an experienced composer friend. I can safely say that I've learned a lot on this journey, not only about programming but also about art and design. I urge the reader to follow my future work at: http://www.icewolf-designs.com/
Epilogue

Appendix A

The Official Design Document
The Plot

Our story begins at a kitschy roadside diner, where none other than Hero Kat, our feline hero, works. He spends his days in drudgery, washing dishes and wishing for a better life. His prayers are answered when a school of panicked fishy minions comes through the diner, asking anyone and everyone for help with their terrible crisis: the beloved Fish Princess of Lake Fishytopia has been kidnapped!

Unfortunately, nobody cares.

Hero sees the answers to his hopes and dreams in the Fish Princess’s quandary. After all, he doesn’t really want to be a dish-washer—he wants to be a famous and world-renowned gourmet connoisseur. And what better way to make your name in the business than to sample the rarest delicacy of all: Royal Fish Flambé.

So begins our intrepid Hero’s Quest. Nobody knows who stole the Fish Princess, or where she could possibly be. Hero’s only choice is to travel the world, exploring every kingdom, defeating hostiles, and confronting each kingdom’s ruler—all while keeping his true intentions hidden from the fishy minions.

Along the way, Hero battles the Beef Czar, the Duke of Ham, and the Cabbage King (to name a few) but at each castle he is met with the same frustrating and infuriating message: “I’m sorry, but your Fish Princess is in another castle.”

So enraged, Hero slaughters and devours the rulers, plunging the lands into Anarchy and chaos, growing fatter all the while. By the time he reaches the Sea
Kingdom, he's transformed from a sleek, fit feline to a tubby ball of fluff.

Of course, as a cat, jumping into the sea is the last thing Hero wants to do. Rather than give up his quest, he instead contracts a submarine captain to take him under. There, 20,000 leagues under the sea, he finally finds the Fish Princess – but it soon becomes clear that she was not kidnapped at all, but rather she swam off with her ne'er do well boyfriend, the Octopus Prince.

Only one question remains: does Hero fight the Octopus Lord, claiming the Fish Princess for his flambé, or does he leave them to their happy life and inform the fishy minions that their princess is not in any danger?
Characters

Hero Kat

Hero, named for the sandwich, is our unlikely protagonist: an aspiring gourmet connoisseur, stuck in a dead-end dish-washing job. He becomes the hero of the story not because he wants to, but because nobody else was willing to take the task. What everyone else doesn't know is that he only wants to save the fish princess so that he can cook and eat her, breaking free from his mundane life and realizing his dream as a famous (or infamous) connoisseur.

Throughout his journey, he learns his true strengths – his bravery, his courage, his natural heroism... and how much he loves sowing the seeds of chaos in his wake. In the end, he must choose between his new heroic identity – helping the helpless and kicking serious ass – or his dreams of power and fame.
**Fish Princess**

We learn from the fishy minions that the Fish Princess is “the most beautiful, kind, intelligent, kind... kind princess in the whole lake!” In reality, she's just a young fish who's tired of being forced to do what her father says. You would think she is the girl who has everything, but she just wants to be where the cephalopods are. She wants to see, wants to see 'em dancing – swimming around with those – what do you call them? Oh – tentacles. Flipping her fins, she doesn't get too far, so she hires a shady submarine captain to take her from Lake Fishytopia to the sea.

**Octopus Prince**

The Octopus Prince is the largely unknown anti-antagonist. He's just a layabout prince who's always had everything handed to him on a silver platter. He's your typical royal bad-boy. At the beginning of the story, we are led to believe that he's an evil villain who has kidnapped the Fish Princess, but in fact, he had only caught a fleeting glimpse of her on one occasion... until, that is, she showed up on his doorstep.
Level Design

Levels are designed with the kingdom in mind. The first, the Grass Kingdom, is the easiest, and is meant to introduce the player to the controls and gameplay. The player encounters bovine enemies and collectable food items. The Grass Kingdom has a plain background of hills and clouds, with a green floor and multi-colored blocks as platforms. In some areas there are water pits that the player can fall into. The player can travel horizontally and vertically. There is a linear path from start to finish, left to right, but the player can deviate from the path to find extra collectables and secrets. The boss of this level is the Beef Czar.

After the Grass Kingdom comes the Marsh Kingdom – a muddy area with more foliage, ruled by the Duke of Ham. In some areas, the player can slip on mud puddles. The player encounters pig enemies here, which are slightly more difficult than the cows in the previous kingdom. The marsh kingdom also introduces underground and secret areas. In some spots, mud puddles hide pits, which if the player falls into, they encounter special enemies, puzzles, or treasure.

Then comes the Farm Kingdom – similar to the Grass Kingdom, but with more pitfalls and enemies. In this level, sentient plants and tractors become enemies, and secret items are hidden in between foliage and the background. This land is ruled by the Cabbage King – who is not, in fact, a sentient cabbage as some of the enemies are, but rather a rabbit who rules with an iron fist.

Other levels include a factory, desert, and jungle themes. The final level is
under the sea, where Hero confronts the Octopus Prince. As the player progresses, the levels grow steadily more difficult, with more tricky terrain and challenging platforming.

**Game Mechanics**

*Super Fish Quest* is a 2D platformer much like *Super Mario Brothers*, and shares much of its game mechanics with the classic game. At the most basic level, the player can control Hero to walk, run, and jump. Enemies are defeated by jumping on top of them, and items are collected by colliding with them. Soon, the player starts acquiring special attacks and gaining access to special use items such as a Spoon-a-Pult or chopstick stilts.

Every time Hero defeats a kingdom’s ruler, he eats them and gets steadily fatter. At the beginning, this doesn’t have any effect on gameplay, but soon it starts to take a toll. Hero starts running more slowly and jumping less high, but gains new abilities such as stronger stomping and bouncing. Eventually Hero gets so fat that he can’t run or walk at all – only rolls along and bounces off of things.

At certain points, the player can choose to complete special mini-game levels where Hero works out swiftly in a Rocky-esque montage and drops some of his weight. Alternatively, the player can choose a different special level to totally pig out and gain weight quickly. In this way, the player has some flexibility in how they play the game – if they prefer to play as a slimmer Hero, they can do that, or they can swap back and forth between skinny and fat with enough effort.
Appendix B

Class Diagram
Appendix C

Design Process

Hero (front)
Design Process

Hero (Side)
Design Process

Fish Princess
Screenshots:
Composing in Anvil Studio 2012:
Bibliography


Persson, Markus. “Minecraft is passing 5 million sales right now. Celebrating with coffee or whatever.” Twitter / @notch. Retrieved February 18, 2012.


Other Resources

Drawing
Adobe Illustrator CS5.1
GIMP 2
Wacom Bamboo Tablet (CTH470)

Audio
Anvil Studio 2012
Bfxr (http://www.bfxr.net/)

Programming
Netbeans IDE 6.8
Java Programming Language
*Killer Programming in Java* example files