

DANIEL HINES

UP747406

Introduction to Mobile Applications

DESIGN DOCUMENT

**Project Overview**

ElfenTrail is a mobile text-based fantasy adventure game with elements of idle/incremental games, created using HTML5, CSS3 and JavaScript. It follows the player as they are unexpectedly taken out of their daily life in Azmir, the Elf Kingdom, and thrust into adventure in the name of the great Elf Queen to investigate the recent emergence of a mysterious dark force. The story will roll out as the player navigates the interface using a series of buttons corresponding to the position in the narrative.

Gamers are the target demographic, more specifically older millennials who will remember DOS games and text-based/point-and-click adventure games of the 90s and to those with an affinity for fantasy stories and retro games in general. As well as appealing to older gamers the game also seeks to combine elements from newer browser/app games such as AdVenture Capitalist and Cookie Clicker which have become popular in recent years so as to extend an invite to a potentially newer audience to the text-based genre. Titles like Cookie Clicker fall under the genre of ‘incremental games’, these are games where the player must continually accumulate a certain resource or set of resources to progress, there may or may not be a story but the drive of the game is in accumulating the resource and they may not have a finite endgame.

In ElfenTrail the primary focus will be the narrative but the accumulation of resources will be woven into the plotline so that it is vital to progression and not just a feature added as an aside. The purpose of using an incremental system is to find novel ways to solve problems, the game will feature no violence as an option to progress and so will use this system of increasing resources along with certain item requirements to offer a unique means to progress through the story.

**Mechanics and Gameplay**

The game mechanics will be built using JavaScript as it arguably the most widely used and open programming language on the web, so it is better documented and supported than other programming languages. It is also constantly evolving and many developers have released libraries which will come in useful during development, such as the implementation of tabs for separate menus.

Possibly the most important aspect of the mechanics ElfenTrail is the “Resource Factory”, which will handle almost every in-game resource from coins to shop items. It is a JavaScript object which contains a library of variables and methods to handle all of the interactions a resource may need. The factory will outline a set of default attributes for each object that can be edited or added to upon creation of a specific item, for example the amount of that resource owned. The methods inside perform functions like creating/updating HTML elements or a timed increase on another resource, if an item provides gold/sec for example.

The narrative will be held piece by piece in a nested array called the Story Box, with the first item in each inner array being an updated ‘steps’ variable that is used to mete out the story one bit at a time. For example:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

var story = [

{ steps: 1, text: 'Opening your eyes you find yourself in the palace of the Elf Queen, Whitespire, although weren’t you just at the Forest Tavern enjoying a cold Leafbrew? As you regain your senses you realise you’re in the White Spire, so the Court Mages must have summed you to consult with the Queen, maybe it’s about all those rumours of some great evil lurking in the forest you’ve been hearing lately.<br>' },

{ steps: 2, text: 'As is custom you take a knee and offer your prayers to the Queen, it is said that those who are blessed carry the light of the Queen wherever they go.<br>'}

]

The game will also feature ‘encounters’, this could be anything from a bandit sticking the player up for their gold or a decision of which path to take. These interactions, as well as most of the story progression, will take place in an Actions pane where the player will spend most of their time. No weapons exist in the game to physically fight any enemies, this is a creative choice to encourage more creative encounters. The player will instead have to have the correct item or the correct amount of gold or make a choice to pass these encounters with an eventual ‘game over’ if the player fails certain parts of the game.

An encounter, as part of the script may look something like:

||Encounter: pay 75% of gold owned to continue or take a 50% penalty blessing if blessing owned is over 20 and have Holy Amulet in inventory||

You feel the Holy Amulet starts to rattle against your ribcage, followed by a blinding light and a calming aura. You slip away amid the confusion, shaken but grateful to have escaped the bandits.

OR

Without a weapon to defend yourself there is no option but to hand over your gold to the bandits, luckily you manage to hide some in a small hole inside your tunic first.

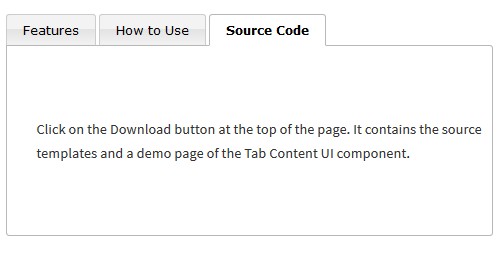
ElfenTrail will make use of a tabbed menu system, which in the prototype is designed using a JavaScript component by Menucool called Tabbed Content and was initially chosen for use in the final design, however due to limitations on the customisation of the tab position and format the final design will boast a completely custom tabbed menu system built again using JavaScript to hide all but one menu at a time. One of the main shortcomings of the Tabbed Content is that the tabs may only be positioned on the top of the element, whereas the final design demands they be on the side, so a custom option may be more time consuming but has a greater payoff. Some of the limitations can be overcome by purchasing a more comprehensive version of Tabbed Content but the cost is more than that the cost in work-hours to create a custom menu, it is also a gamble as to whether the full version allows enough customisation without breaking the source code apart which again could be more time consuming than a bespoke solution.

Figure : A Tabbed Content template, can be styled but requires a paid license for more customisability.

Players will also be able to make us of Save and Load Game facilities which will take advantage of the devices local storage as opposed to cookies, as local storage is read client-side and also is not wiped when the browser cache is cleaned. Not even session cookies are needed as the game is all on one page. Storing the players’ progress will be in the form of a list of all the independent variables that make up their game so far. Should the player encounter a Game Over scenario, they may then load their previous save and not have to restart the entire story from the beginning.

**Design Stage**

Visually ElfenTrail aims to combine retro 8bit fantasy graphics and current trends in internet art. The reason these styles were chosen is that they’re already somewhat connected, recent internet trends like vapourwave artwork uses imagery associated with outmoded, dated computer renderings and 90s web design which is also the era of 8bit graphics.

The chosen style is also culturally relevant and popular with the target demographic, there has been a resurgence in popularity of retro style graphics, games like award-winning The Escapists (Figure 2) and the ever-popular Minecraft are great examples of hugely popular modern games with an old-school feel.

Figure 2: The Escapists, an example of a modern take on retro graphics

Internet art, or net art, is artwork distributed via the internet and is extremely popular with the boom in popularity of smartphones, laptops and the internet in recent times. Again chosen as an influence with the target demographic of millennials in mind who grew up at the advent of the internet, so they will be aware of the associations and familiar with the styles and trends, even if only subconsciously. Current styles, such as artwork associated with the music genre Vaporwave which originated from and is deeply involved with internet culture, evokes a retro tech feel with anime influences, make use of lots of pastel-style gradients and features frequent use of fuchsia/aqua colour combinations.

Figure : An example of 'Vaporwave aesthetic', a recenet trend in Net Art

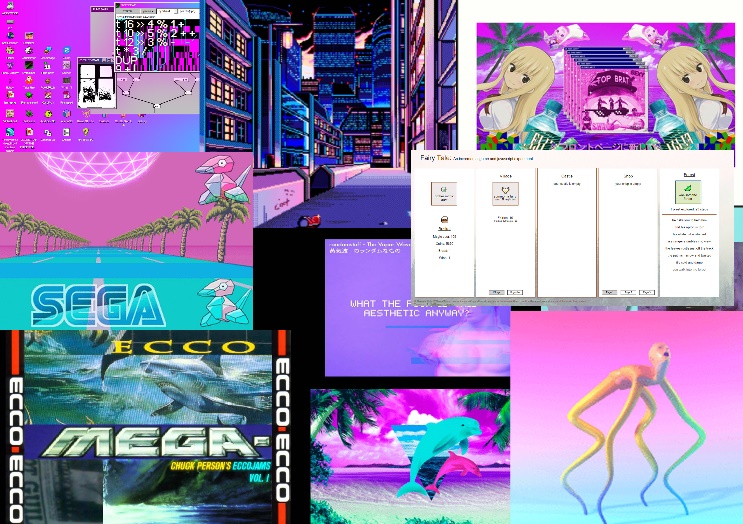
Figure 3 shows a compilation of images representing the two styles, used for visual reference during the initial stages of design:

Figure 4: Design Collages

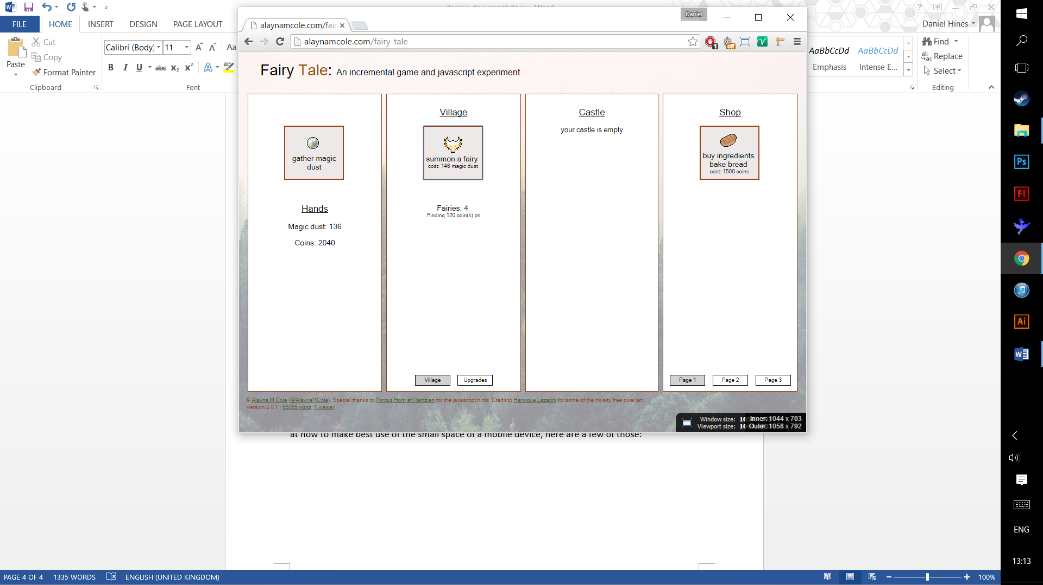
Market research was conducted into competitors and similar titles to inspire layout designs and look at how to make best use of the small space of a mobile device. Fairy Tale is more of an experiment than a game but has some similar features to ElfenTrail, particularly with the fantasy theme and a Shop menu. The game makes very little use of graphics, the graphics used are mostly pixel art though and in a similar vein to the style ElfenTrail is trying to capture. The menu system suits a desktop but doesn’t resize or reposition elegantly for mobiles, there is a nice discreet tab system in use here though so the code for that will be useful to look at when developing the ElfenTrail tabbed menu.

Figure :Fairy Tale interface

Here is a design mockup inspired by Fairy Tale:

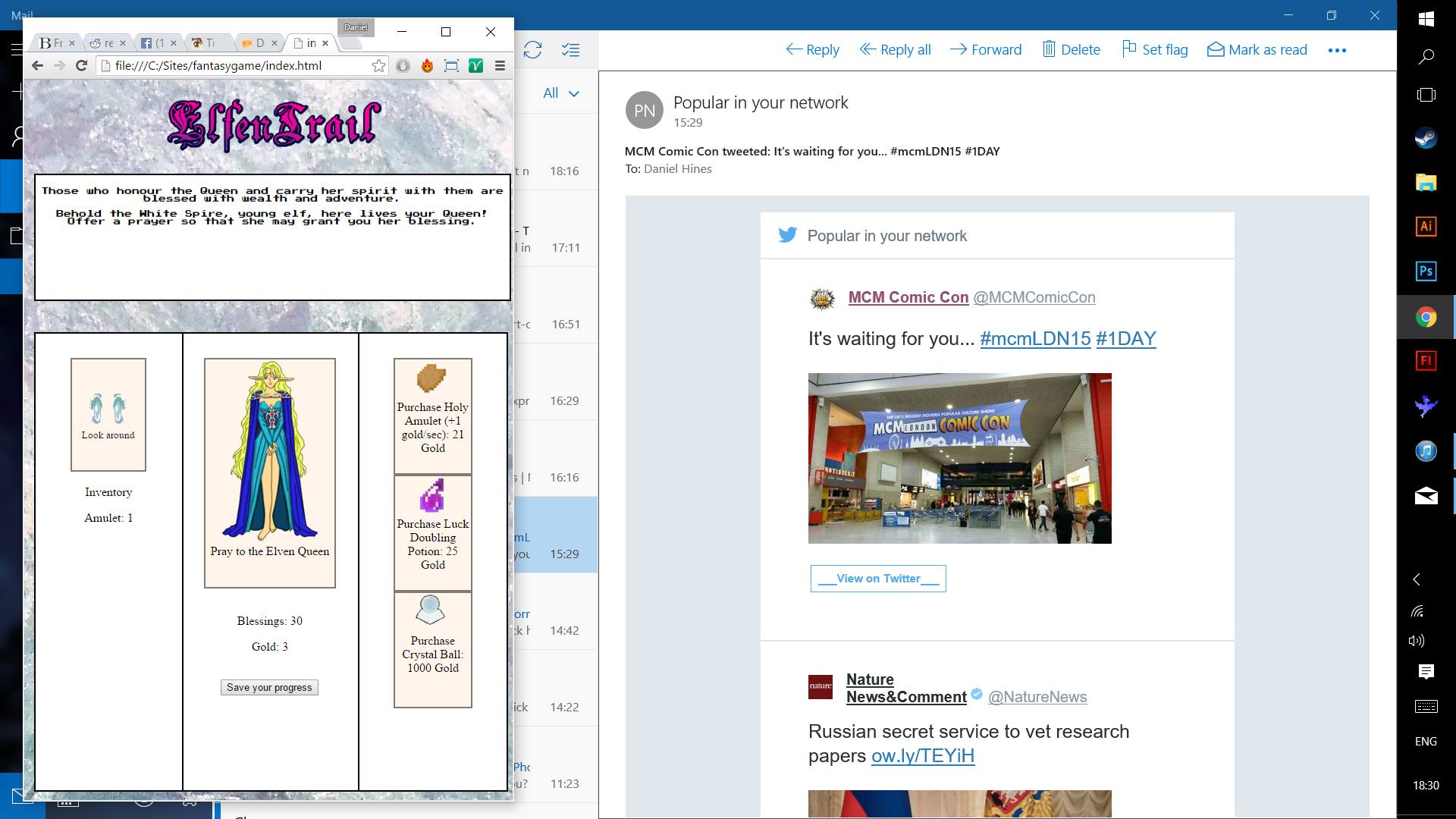


Figure : ElfenTrail early design mockup



The motivation behind this design was to have every button on a single screen, also to fill out the interface, but as the player progresses more options and shop items will become available which may overcrowd the limited space of a mobile device. The colour scheme was also deemed too dull and flat, it lacked a vibrancy that would be useful in drawing attention to the application. The idea behind the marbled background was again inspired by the Vaporwave aesthetic, which features lots images of marble artefacts particularly from Ancient Greece and the Roman Empire. Most of the small pixel-art graphics used are placeholders that match the design concept, however they are not the final artefacts.

Figure 7: Vaporwave art featuring marble statues/marbled backgrounds