

Website Report

Name: Matthew Beaney

Student Number: 2418770

Course: Cyber Security and Networking

Module: Introduction to Web and Database Concepts

Date of writing: 9th January 2025

Contents:

Introduction (page 1)

Requirements (page 1)

Implementation/Validation (page 2-4)

Evaluation (page 4-5)

References (page 5)

Appendices (page 5-8)

Introduction

This report will go over the contents of my website I have built, reasons for elements of its design and reasons for lacking features or shortcomings. This will include my reasoning for the website's subject and my implementation of my plans.

Requirements

The purpose of my website is to provide a basic introduction to the wiring system in the videogame "Barotrauma". This covers the wiring components and a general explanation of their functions and going over some examples of usage. This is intended to be for any audience interested in the game.

My reason for choosing this subject matter is due to my own pre-existing knowledge and interest in the subject making it easier for my mind to focus on the work and to come up with ideas for the website's contents. I chose this particular part of the game

due to its similarity to logic gates systems seeming most relevant to a computer science course.

I chose to present the website as a series of images in order to maintain an aesthetic I thought would fit the subject matter, using game files for loading screens as backgrounds.

Implementation/Validation

Document checking completed. No errors or warnings to show.

Used the **HTML** parser.

Total execution time 3 milliseconds.

Barotrauma wiring guide

1

The system for wiring in the game Barotrauma operate on a very basic system of easy power transfer and a separate functionality of components which is almost entirely a system to create logic gates. These components are shown below.

Most of these components have configurable input wanted and output.

-  AND operates as an AND logic gate, checking that both conditions are providing an output, that signal in 1 and 2 are both on.
-  Greater checks two inputs' values and outputs a value dependent on the result. This can be used as a lesser than function by changing the output.
-  Equals operates like an AND logic gate, although it checks both signal in are the same.
-  NOT sends a signal when the input is not receiving a signal.
-  OR sends a signal when at least one signal in receives a signal.
-  RegEx Find seeks for an exact string in an input.
-  XOR functions an XOR gate. When one, but not both, signal in receive a signal; a signal is output.
-  Signal Check is the same as Equals however with a while condition applied to it. When the wanted signal no longer is being sent, the signal check outputs a false signal for the duration until the condition is met again.

There is also a variety of maths & trigonometry components that perform their shown function.
e.g.  Absolute value converter.



Page 1
Page 2
Page 3
Page 4
Page 5

There are also a variety of miscellaneous components for other functions.

- Concatenates two inputs and outputs the result.
- Delays all received signal for a specific amount of time.
- Outputs a stored value that can be updated from other sources.
- Sends a periodic oscillating signal.
- Forwards input connections to the output. Also has a gate function to halt signal and power output.

- The wifi component is the most unique of these components. It provides the unique ability to interact with the in-game text chat and the 10001 (channel 0 to 10000) usable radio channels. It can output print messages to the radio text chat, read messages from it and can transfer signals over long distances to save effort in wiring, although ends up often being more resource intensive if used for that purpose.

The next few pages will go over some examples of wiring and their solutions.

Wiring up an airlock/docking hatch with proper logic to prevent massive flooding.

Setting up a wifi component to report sub equipment's condition.

An automatic flooding system to quench flames.

Page 1

Page 2

Page 3

Page 4

Page 5

Airlock wiring example

Displayed here is an example of an airlock and docking hatch. We want the bottom hatch to be able to open and close at the press of a button.

However, we also want the bottom hatch to be always open whilst the docking hatch is attached to another ship or a station.

We also only want the water pump to function while the door is closed or the hatch is docked.

Firstly, we want the docking hatch's State out to be wired into a signal check component so that it will not interrupt other functions unless docked. While the 1 goes through the signal check component it will override anything else to make the water pump work and hatch to be open.

This will be achieved by setting the False output to be blank and the Output to 1. We then wire the output to the set state of the hatch and add this to an or component.

The hatch is also wired up to buttons that send a signal to the toggle state of the hatch. This makes it so that each press changes the state of the hatch so long as the docking hatch is undocked.

The state out of this hatch connects to a not component, reversing the output. This connects to an or component accepts a 1 from the not component or the docking hatch. The or component sends its output to an and component.

The and component accepts input from the or component and a water detector to make it so that if the ship is docked or hatch is closed and there is water in the airlock, the water pump turns on to start pumping out water.

Page 1

Page 2

Page 3

Page 4

Page 5

Sub equipment condition reporter example

4

There are multiple ways this may be done depending on what is wanted.

One may want a reporter that only reports a piece of equipment's condition when it hits certain thresholds, such as 50% and 25%. This may be done by linking multiple signal check components with blank outputs to the condition out of the specified equipment. The target signal would be those values and the output would be a text message reporting those percentages. These signal components would output to the signal in of a wifi component that is linked to chat, at the a chosen radio channel.

You can more simply use a wifi component on its own, receiving the signal directly from the condition out. However, this will be much more non-specific, only showing the exact number and nothing else.



The 100 is a text terminal I was using to debug setting this up, as it would read out the exact condition value live.

Page 1
Page 2
Page 3
Page 4
Page 5

Automatic fire fighting flood system example

5

This would be achieved by interaction with the pump in a room with items that may catch fire, such as junction boxes or the reactor.

When the smoke detector provides a negative output, the water detector functions normally. This would be done by connecting the output of the smoke detector to a not component, which feeds into the set state of a relay component which passes the output of the water detector. This closes the transmission of signal from the water detector when a fire is present, making the pump not attempt to pump out water. When there is smoke, the smoke detector triggers the output from a signal detector component to set the pump to bring in water from outside and to turn on. This would automatically end when the water quenches the flames as it would not longer detect smoke.

When the smoke is quenched the water detector would function normally and start pumping out the water used to quench the flames.



Page 1
Page 2
Page 3
Page 4
Page 5

Evaluation

To evaluate a website we must provide parameters by which we are judging it and must define those parameters.

Usefulness, which is made up of usability and utility.

Usability is “how easy & pleasant [the website's] features are to use”.

Utility is “whether [the website] provides the features you need” ^[1].

Accessibility

Accessibility is when “websites, tools, and technologies are designed and developed so that people with disabilities can use them” [2].

Search engine optimisation

“Search engine optimisation is the process of improving the quality and quantity of website traffic to a website or a web page from search engines” [3].

I would consider my website to be very utilitarian. It has plenty of text and images explaining its points and detailing information in a reasonable way.

However, I believe it is lacking in the usability and accessibility department. This is because the pages are completely images, this makes it impossible to use font changers, to highlight text as you read. This may make it difficult for those with poor vision, dyslexia or other disabilities to read unless this is changed, or alt text is provided. Although the backgrounds’ colours contrast well in comparison to the text, both in my opinion and averaging the colour and putting it through the provided Luminosity colour contrast ratio analyser and WCAG gives it a very good pass for contrast.

My website cannot be optimised within a search engine as it is a html file that must be locally downloaded to view.

References

- [1] <https://www.nngroup.com/articles/usability-101-introduction-to-usability/>
- [2] <https://www.w3.org/WAI/fundamentals/accessibility-intro/>
- [3] https://en.wikipedia.org/wiki/Search_engine_optimization

Appendices

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Barotrauma wiring guide</title>

<style>

  body {

    margin: 0;

    height: 100vh;

    display: flex;

    justify-content: center;

    align-items: center;

    background-color: #f0f0f0;

  }


  #container {

    width: 1920px;

    height: 1080px;

    position: relative;

  }


  #image-container {

    width: 100%;

    height: 100%;

    display: flex;

    justify-content: center;

    align-items: center;

    overflow: hidden;

  }
```

```

#image-container img{

    width: 100%;

    height: 100%;

    object-fit: cover;
}

/* button container in the bottom right */
.button-container {
    position: absolute;
    right: 20px;
    bottom: 20px;
    display: flex;
    flex-direction: column;
    align-items: flex-end;
}

button {
    padding: 10px 20px;
    margin: 5px;
    font-size: 16px;
    cursor: pointer;
    background-color: #fff;
    border: 1px solid #ccc;
    border-radius: 5px;
    transition: background-color 0.3s;
}

button:hover {
    background-color: #ddd;
}
</style>

<div class="button-container">
    <button onclick="changeImage(1)">Page 1</button>
    <button onclick="changeImage(2)">Page 2</button>
    <button onclick="changeImage(3)">Page 3</button>
    <button onclick="changeImage(4)">Page 4</button>
    <button onclick="changeImage(5)">Page 5</button>
</div>

```

```
</div>
```

```
<script>
```

```
    function changeImage(imageNumber) { // this is done to be more  
efficient in code space than having a seperate function for the  
individual buttons
```

```
        var imgElement = document.getElementById("current-image");
```

```
        imgElement.src = "page" + imageNumber + ".png"; // change
```

```
page
```

```
    }
```

```
</script>
```