Last Modified

30-Oct-21

Software Version

0.5.0.0

Clyde River Application

Programmer’s Manual

Table of Contents

[Introduction 2](#_Toc86444749)

[Dependencies 2](#_Toc86444750)

[Decisions & Justifications 3](#_Toc86444751)

[Business Aims 4](#_Toc86444752)

[System Architecture 5](#_Toc86444753)

[Class Diagrams 5](#_Toc86444754)

[Sequence Diagrams 6](#_Toc86444755)

[Other Resources 8](#_Toc86444756)

[Glossary 8](#_Toc86444757)

# Introduction

This user manual documents the Clyde River Desktop Application System. This application allows oyster farmers to access and view data from sensors on the Clyde River.

This document aims to document the technical side of things and allow the reader to efficiently and effectively manage and maintain the application.

# Dependencies

This application relies upon:

|  |  |  |
| --- | --- | --- |
| **Name** | **Dependencies** | **Current Version** |
| XChart |  | 3.8.0 |
|  | VectorGraphics2D | 0.13 |
|  | Graphics2D | 0.30 |
|  | PDFBox | 2.0.22 |
|  | FontBox | 2.0.22 |
|  | Commons Logging | 1.2 |
|  | Animated Gif Library | 1.4 |
| JDatePicker |  | 1.3.4 |

This Application Used The Following Development Environment Tools:

|  |  |
| --- | --- |
| **Name** | **Current Version** |
| Eclipse IDE | 4.10.0 |
| WindowBuilder | 1.9.7 |
| Junit | 5.3.1 |
| Maven | 4.0.0 |

# Decisions & Justifications

Decisions made about this application are many and varied and are documented in various other documents. Important decisions you should be aware of include:

* Java was chosen as the language of choice because it is highly flexible but also because it was the language the original developer was most fluent in.
* XChart was selected as the charting library of preference because it was lightweight, fully featured easy to use and well documented
* Maven was chosen primarily for dependency management
* WindowBuilder facilitated quick GUI building

With the target audience being Oyster farmers, the following decisions were made:

* Salinity was prioritized as **THE** most important data to provide.
* Salinity will therefore be supplemented with Temperature and Rainfall data.
* Temperature was deemed the third most important data simply because Rainfall has an influence on Salinity.
* The application is designed from a user perspective and therefore seeks to present the most relevant information for the user first while providing ease and convenience to access wider and more broadly helpful information and data.
* The application has been designed with the least tech literate user in mind and therefore uses as plain language as possible.
* The application seeks to minimize user input where not necessary to provide a more ease of use experience.

# Business Aims

The primary purpose for the creation of this application was to provide a more convenient and easy to use system for accessing and viewing data from sensors on the Clyde River.

As such, some assumptions were made, including:

* Oyster Farmers may use any kind of operating system
* The Application needs to be compatible with any viable desktop operating system.
* The application should be easier to use than the current system

As a result of these assumptions, Java was the preferred programming language as it allowed the application to be highly compatible with whatever the end user’s operating system may be.

Due to this, the decision to use XChart as the charting library was made in part because it is compatible with Java.

XChart was also selected due to the simple, clean and easy to read charts that it can render as well as the fact it was very lightweight and highly featured which would allow it to easily and quickly provide the charts for the application.

XChart also had a good level of documentation which makes for a much easier application to be maintained in the future as well as to be created in the first place.

A decision to create a central backbone navigational system was partly driven by the need for high levels of ease of use. As such, the menu navigation system quickly and easily allows the oyster farmer to find the data that they need at that moment.

A core part of making the application easy to use was the design layout. The application makes use of a number of summary graphs which allows the user to quickly and easily see the general trend of the sensor at a glance and as such comprehend quickly what the general sense of the data for that sensor is without having to drill into the minute details.

The application also seeks to have a user centered approach with its layout. It therefore seeks to present highly localized data first which would be of most interest to a user and then seeks to expand to data from the area more broaly.

As such a simple guiding principle to data order would be localized summary data, followed by localized data in full, followed by broad summary data and finally followed up by broad data in full from the area.

# System Architecture

## Class Diagrams

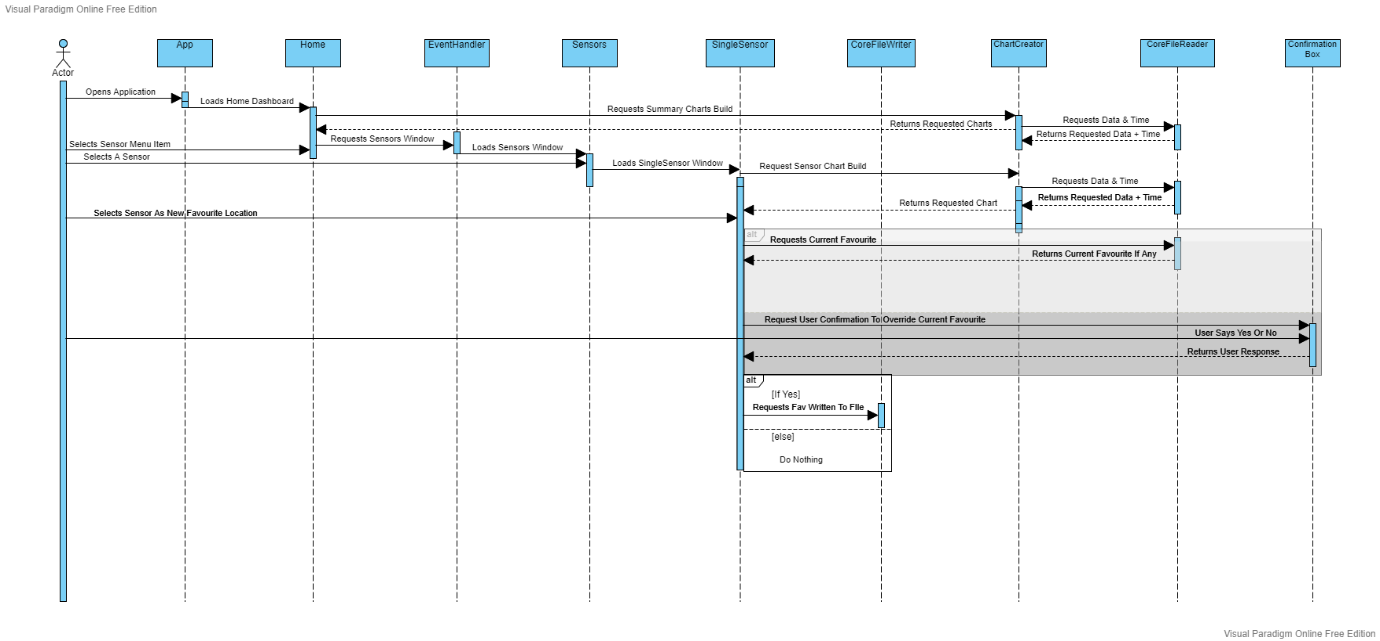
Diagram

Description automatically generated

You will notice that as a result of this diagram that the classes are highly interconnected. You will see that a bulk of the application lies in CoreFileReader and SingleSensor, supplemented by EventHandler and ChartCreator.

You should also note that there is a note under ChartCreator that provides more information about the corresponding method in that class.

## Sequence Diagrams

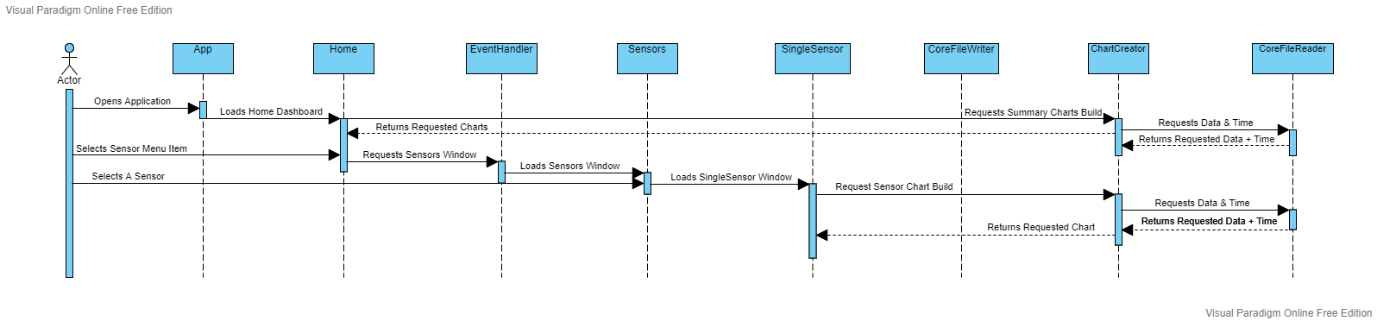


Above you can see the sequence of events for a user selecting a favourite location for the first time or for replacing their existing favourite location.

You’ll notice a lot of the actions are automatically carried out by the system on the user’s behalf. This is designed to reduce user input which reduces the risk for errors and ensures a smooth and easy to use application and a more enjoyable experience for the end user.

You will also notice that some classes are significantly more core and critical to the application than others are. In particular the ChartCreator and CoreFileReader with special mention of the EventHandler Class which runs the backend of the navigation.

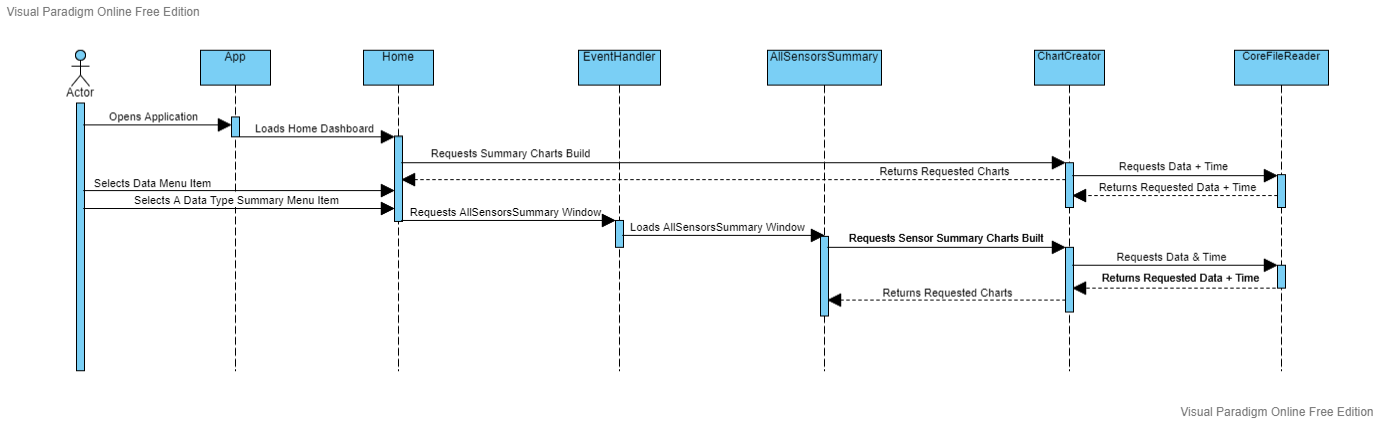
The name is somewhat less descriptive but it is called EventHandler because it handles the mouse click events on the menu, thus EventHandler.



Above you will notice the sequence of events for a user navigating to a single sensor to view it.

The process is much less complex than saving a favourite location but both being with a similar start. Again you’ll notice user input is reduced, exclusively to mouse clicks.

It’s important to note that early on you’ll see there’s calls to the ChartCreator and CoreFileReader and once they’re done then there is more user input.



Above you will notice the sequence of events is nearly identical to the previous sequence diagram. In this case the user navigates to the summary screen of all sensors as opposed to a single sensor.

Primarily these are going to be the main things a user will want to do with the application.

It is also worth noting that while not within the scope of the original project, the application provides the ability for the end user to save charts as images to their computer by right clicking on them and selecting save as…

As well as that the application provides the ability for users to also print the charts off if they so desired.

These functions are handled directly by the charting library XChart and are therefore not documented as part of the application.

These UML diagrams have been included as separate files alongside this manual to provide more convenience and flexibility with understanding how this application works.

# Other Resources

There are a number of documents that you may find helpful with regard to understanding the application beyond this manual. These documents include:

|  |  |
| --- | --- |
| **Document Name** | **Current Version** |
| user manual.docx | 1.0 |
| project-vision.md | 1.0 |
| functional requirements.md | 1.0 |
| Clyde River Mobile Application Requirement Model.md | 1.0 |
| architecture-notebook.md | 1.0 |
| project-summary.md | 1.0 |
| risk-list.md | 1.0 |
| master-test-plan.md | 1.0 |
| wireframes.md | 1.0 |
| Clyde River Application Wireframes.pdf | 1.0 |

# Glossary

**Animated Gif Library** - A re-package of the Animated GIF processing classes made available by Kevin Weiner - <https://javalibs.com/artifact/com.madgag/animated-gif-lib>

**Commons Logging** - Apache Commons Logging is a thin adapter allowing configurable bridging to other, well known logging systems. - <https://javalibs.com/artifact/commons-logging/commons-logging>

**Eclipse IDE** - Eclipse is an integrated development environment used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. - <https://www.eclipse.org/ide/>

**FontBox** - The Apache FontBox library is an open source Java tool to obtain low level information from font files. FontBox is a subproject of Apache PDFBox. - <https://javalibs.com/artifact/org.apache.pdfbox/fontbox>

**Graphics2D** - Graphics2D Bridge for Apache PDFBox - <https://javalibs.com/artifact/de.rototor.pdfbox/graphics2d>

**Java -** Java is a high-level, class-based, object-oriented general-purpose programming language intended to let programmers write once, run anywhere. - <https://www.java.com/en/>

**JDatePicker** - JDatePicker and JDatePanel is a Java Swing component. It is designed with the Model View Controller pattern and it uses the Java Calendar object as the model. The component may be used via a popup control or embedded on a panel. - https://javalibs.com/artifact/org.jdatepicker/jdatepicker

**Junit** - JUnit is a unit testing framework for the Java programming language. - <https://junit.org/junit5/>

**Maven** - Maven is a build automation tool used primarily for Java projects. Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages. - <https://maven.apache.org/>

**PDFBox** - The Apache PDFBox library is an open source Java tool for working with PDF documents. - <https://javalibs.com/artifact/org.apache.pdfbox/pdfbox>

**VectorGraphics2D** - Graphics2D implementations to export various vector file formats - <https://javalibs.com/artifact/de.erichseifert.vectorgraphics2d/VectorGraphics2D>

**WindowBuilder** - WindowBuilder is composed of **SWT Designer and Swing Designer** and makes it very easy to create Java GUI applications without spending a lot of time writing code. - <https://www.eclipse.org/windowbuilder/>

**XChart** - XChart is a light-weight and convenient library for plotting data. Its focus is on simplicity and ease-of-use, requiring only two lines of code to save or display a basic default chart. - <https://javalibs.com/artifact/org.knowm.xchart/xchart>