StormWare Version 4

McMaster University

Department of Computing and Software

COMPSCI 2XB3

April 10, 2018

Group 15

Anirudh Verma

David Hospital

Sijie Zhou

Zijing Chen

**Table of Content**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Team Members and Role Assignment**

|  |  |  |
| --- | --- | --- |
| Team Members | Student Number | Roles |
| Anirudh Verma | 400039737 | Owner |
| David Hospital | 400015029 | Developer |
| Sijie Zhou | 400038163 | Quality Assurance |
| Zijing Chen | 400020376 | Designer |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Timestamp | Originator | Version Number | Comments |
| 180209T1014 | All | #1 | Topic selected |
| 180209T1014 | All | #1 | Team roles decided |
| 180209T1014 | All | #1 | Setup github repo |
| 180209T1014 | All | #1 | Implementation platform decided |
| 180216T1107 | All | #1 | Create log file |
| 180216T1107 | All | #1 | Specification decided:  Using google map API and possibly machine learning for data parsing |
| 180228T1405 | All | #1 | Completed requirements specification |
| 180302T1100 | All | #2 | Start prototype |
| 180302T1100 | All | #2 | Discussed prototype design and requirements |
| 180305T0200 | David Hospital | #2 | Create Android Studio project and commit it to github repo |
| 180307T1300 | All | #3 | Discussed relevance of machine learning and abandon machine learning |
| 180309T1000 | All | #3 | Continued work on Parser.java and created Test.java.src/.../Parser.java |
| 180316T1000 | All | #3 | Discussed project objective and how much data to display at once |
| 180323T1000 | All | #4 | Final version confirmed and requirements modified |

*By virtue of submitting this document we electronically sign and date that the work being submitted by all the individuals is the group is their exclusive work as a group and we consent to make available the application developed through CS-2XB3 project, the reports, presentations, and assignments (not including my name and student number) for future teaching purpose.*

**Contribution**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Role | Contribution | Comment |
| Anirudh Verma | Owner | Parser module |  |
| David Hospital | Developer | Proposal |  |
| David Hospital | Developer | Data and DisasterEvent module |  |
| Sijie Zhou | Quality Assurance | Requirement spec |  |
| Sijie Zhou | Quality Assurance | Presentation slide |  |
| Sijie Zhou | Quality Assurance | Design spec |  |
| Zijing Chen | Designer | MainActivity module |  |
| All | Group 15 | Design spec | The design spec is not finished by an individual. It’s a group effort. |

**Executive Summary**

The application is an Android mobile program using Google Map API which will help user to have an overall view of natural disasters in America. Users can choose one exact disaster type and the output will show by the form of heatmap.

**UML Class Diagram**

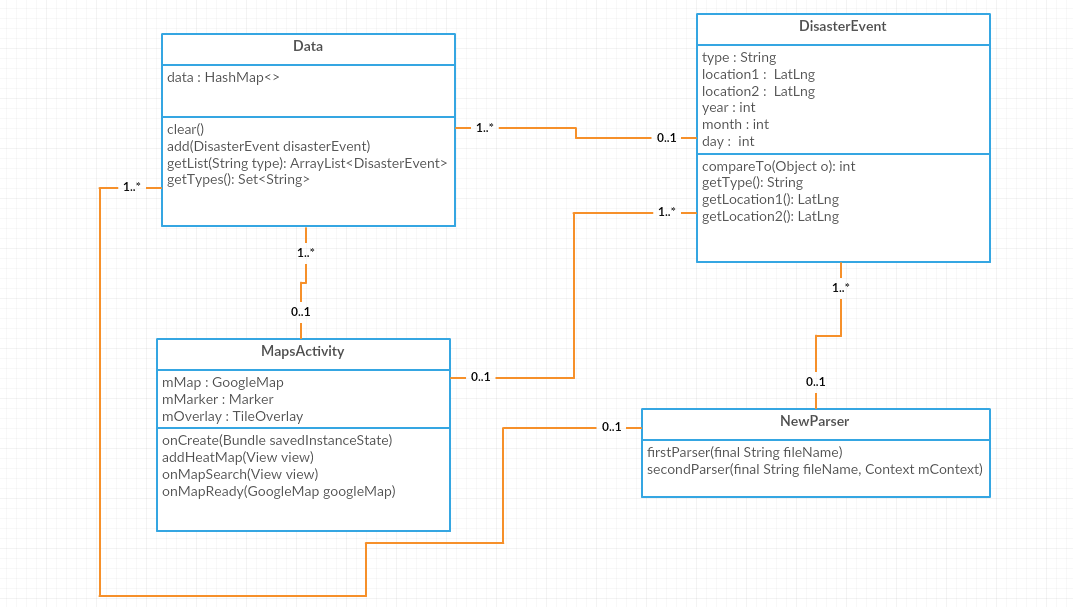


Figure 1 DisasterEvent uses nothing. Data uses DisasterEvent. NewParser uses data and DisasterEvent. MapsActivity uses NewParser, Data and DisasterEvent.

DisasterEvent:

* Module for creating DisasterEvent objects, used to store information about each event.
* Has start and end location, time (year, month, and day), and event type variables.

Data:

* Module for Storing and sorting a large collection of DisasterEvent objects.
* Uses a HashMap to partition events into different lists, separated by their type.
* Lists can be accessed by using the event type as a key in the HashMap
* When a event is added and the HashMap does not contain a list for that type, it creates a new list and appends the event to it.

NewParser:

* Module for parsing data from specifically designed csv files directly into the Data module
* The method *firstParser* is responsible for condensing the raw data file with over 40 columns into a smaller file with just 8 columns.
* The method *secondParser* is responsible for parsing the condensed data file and creating a DisasterEvent object for each row. These events are added to the Data module using Data.*add*.
* The parsing time for the *secondParser* is 80% faster than *firstParser*, making it a valuable improvement to the module.

MapsActivity:

* Module for handling the controller and view components of the program.
* Uses the Android framework as a backbone for handling most of the input and output events.
* The *onCreate* method is called when the program starts and is responsible for loading the google map view onto the screen.
* The *addHeatMap* method is called whenever one of the type buttons is pressed (UI event). It removes the old heatmap object if there is one, and then adds a new one, by getting the list of DisasterEvent objects from the given type from the Data module. It then passes that list to the google map framework to create a heatmap and displays it on the map.
* The *onMapSearch* method is called whenever the user presses the search button (UI event). The text that is currently in the search bar is sent to the google map framework which returns a list of possible addresses that it might match. The first one is picked (best result). A marker is created (the old one is removed) on the map at that result and the camera is translated to focus on the new marker.