Diamond:
An Online Tool for Card Sorting and Tree Testing

Christopher Oser, Markus Ruplitsch, Markus Stradner

706.041 Information Architecture and Web Usability WS 2020/2021
Graz University of Technology

1 Feb 2021

Abstract

This report summarizes the functionalities of Diamond, an application that offers both, tree testing and card sorting. Since the tree testing capabilities had already been implemented previously, the focus of this report is an overview on the new card sorting option that was implemented. Some insights into card sorting in general as well as the necessary technologies to implement the application are also provided.

Contents

List of Figures

Chapter 1

Introduction

Diamond: An Online Tool for Card Sorting and Tree Testing was created as an extension of the TreeTest tool, which was implemented by Ajdin Mehic as part of his Master's Thesis [Mehic 2019]. Because of providing both functionalities a card sorting tool and a tree testing tool now the application was combined under the new name.

In this report you will find a brief introduction to card sorting and an explanation of all technologies used to create Diamond. The implementation steps are also listed and explained in a constructive manner. Finally, there are expansion ideas for the future and final remarks mentioned.

1.1 Creating a Card-Sorting Application

Since there are already many tools out there, it is not easy to find a good approach to implement a new app in this sector. One of the best existing examples is OptimalSort [Optimal Workshop Ltd. 2021].

At Diamond the main target was to keep it simple, but also to equip it with many features as possible. Therefore, the card sorting process itself is designed very intuitive, but also the setup process of a new study is created clean and straightforward. Here it is possible to create cards rapidly via .csv import.

In addition to editing and deleting, the status of each single study can be determined. For this you can switch between launch and unlaunch. For launched studies you can easily send an invitation link to the users.

Finally, after completition of all the participants, you also have the opportunity to export the results as .csv in different ways.

Chapter 2

Card-Sorting

In general, card sorting is a technique in user experience. Participants of a study have to assign the different cards into the matching category, which is more or less a low-tech approach.

It is mostly used for designing a navigation structure, but it can be used for evaluating an information architecture of a website or any other information structure, too. A sample image of card sorting can be viewed in Figure 2.1.

2.1 Procedure

Participants of a card sort study assign cards to categories. This can be done either physically with paper or carton cards, or nowadays more often in digital form where the user is given a list of cards and needs to assign them to groups. The exact procedure depends on the defined variant.

2.2 Variants

There are three different types of card sorting. In closed card sorting the users have a fixed setup for the categories and in open card sorting they have to create the different categories for their own. But there is also a hybrid variant where you actually have a closed card sort but you are also be allowed to add additional custom categories.

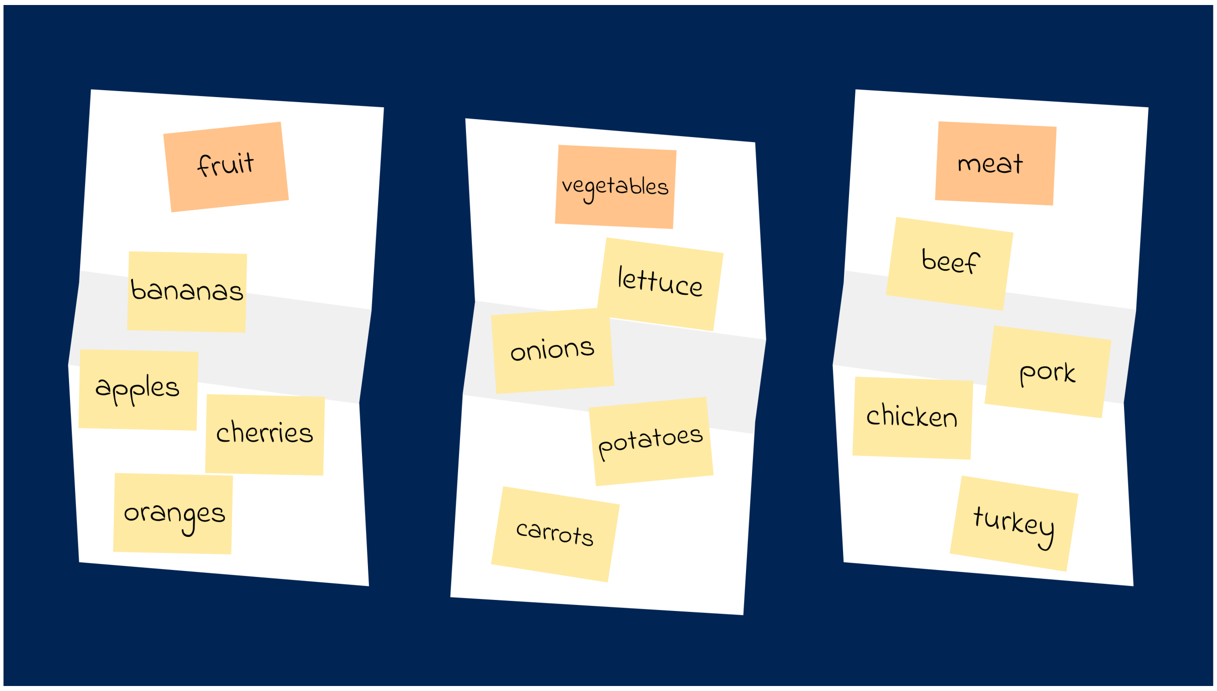


Figure 2.1: A sample image of a card sorting shows dividing cards in three categories. [Drawn by Markus Stradner, inspired by the illustrations accompanying Sara Jee Watson's blog post (`https://boagworld.com/usability/card-sorting/`).]

Chapter 3

Technologies

This chapter covers the different frameworks and technologies that were used to create Diamond. While many of these technologies provide a wide array of features, only the features used during the development of Diamond will be mentioned here. For more in-depth information it is suggested to have a look at the different documentations.

3.1 Angular

Angular is one of the most popular design frameworks for web applications. Angular applications are structured in such a way that each visible part of the app has its own .css, .html and .ts file and is called a component. As with other web applications the typescript file contains the logic, the HTML file contains the markup and the CSS file defines the look and feel of the component. TreeTest was originally written in Angular 7 and was updated to the newest version of Angular (Angular 11) before development of Diamond began.

3.2 Node.js

Node.js is a JavaScript runtime environment that was used to set up the server side of application. It simplifies many of the different web protocol interactions and allows developers to handle requests and responses asynchronously without having to implement multi threading themselves. Node.js was also used to store and retrieve the information stored on the MongoDB database.

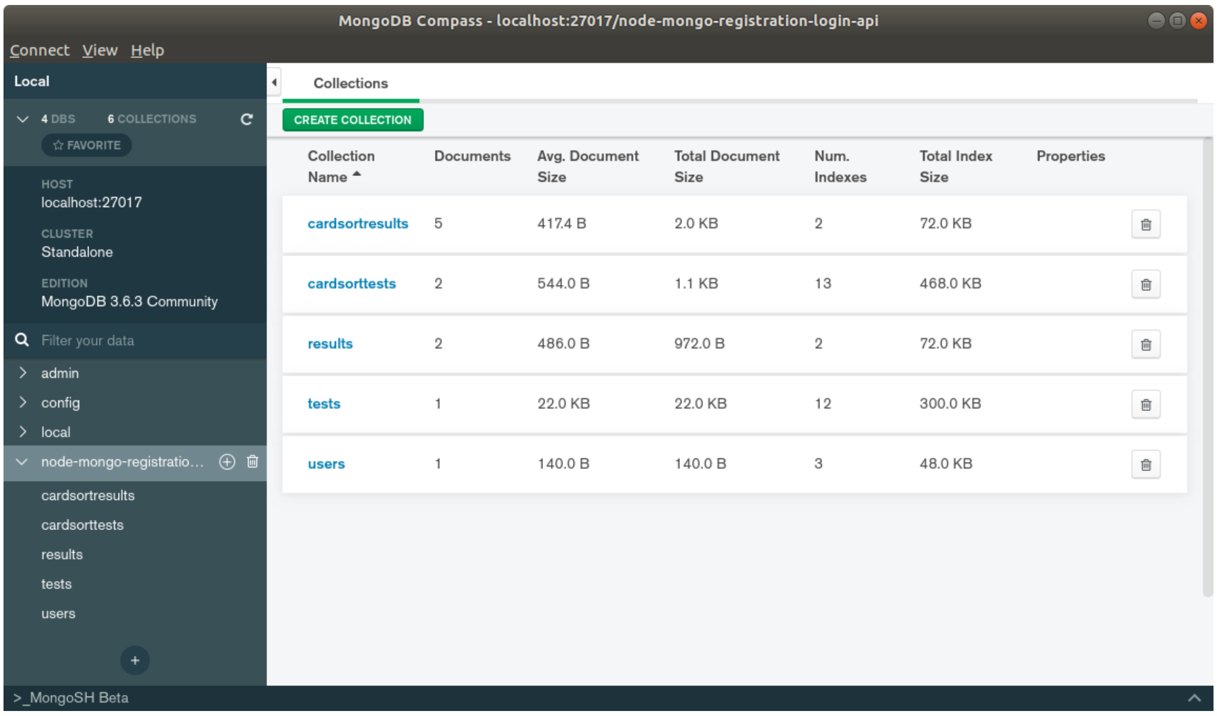
Additionally, npm is a package manager that comes with Node.js by default, and was used to clone all used packages and keep them up-to-date during development. This makes the use of external libraries significantly easier and reduces the overhead when committing changes to git.

3.3 MongoDB

To store all user related information a MongoDB database was set up. The communication between our server and the database was handled by Node.js. Furthermore, MongoDB compass was used to visualize the database and manually interact with it, in case some erroneous information was stored. A screenshot can be seen in Figure 3.1.

3.4 Heroku

Heroku was used for online hosting. In general it is a container-based cloud Platform as a Service (PaaS). This means that everything about setting up a server, online deployment and hosting is provided by it. It is very intuitive to use and free of charge, but an account is needed.


MongoDB Compass - localhost:27017/node-mongo-registration-login-api

Connect View Help

Local

4 DBS 6 COLLECTIONS

☆ FAVORITE

HOST
localhost:27017

CLUSTER
Standalone

EDITION
MongoDB 3.6.3 Community

Filter your data

- admin
- config
- local
- node-mongo-registratio…
  - cardsortresults
  - cardsorttests
  - results
  - tests
  - users

Collections

CREATE COLLECTION

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
cardsortresults	5	417.4 B	2.0 KB	2	72.0 KB	
cardsorttests	2	544.0 B	1.1 KB	13	468.0 KB	
results	2	486.0 B	972.0 B	2	72.0 KB	
tests	1	22.0 KB	22.0 KB	12	300.0 KB	
users	1	140.0 B	140.0 B	3	48.0 KB	

>_MongoSH Beta

Figure 3.1: MongoDB Compass showing the database with some stored information [Screenshot was captured by Markus Ruplitsch using MongoDB Compass.]

Heroku offers to connect a GitHub repository to that account. This is quite convenient, because you can choose a branch to enable auto-deployment after every single push on this branch. For this reason you can keep your web application up-to-date very easily.

It is recommended to created an Heroku branch and to push on that branch, when there is a larger number of changes to deploy online. And the main branch should just be configured with the localhost settings for testing locally.

Chapter 4

Implementation

In this chapter the core functionalities of the card sorting capabilities of Diamond are presented. Most importantly, the actual sorting implementation is explained. Furthermore, all other aspects, such as creating a card sort study or exporting and possible analytics of the results are also discussed.

4.1 Creating a Card Sort

The main structure of card sort study creation was taken from the previously implemented TreeTest, but it was not copied entirely. Adjustments were made to better represent a card sort study.

Every study has a name and an option to include a mandatory password to guard the study from unwanted participants. Then a welcome message, instructions as well as a thank you message and feedback message can be specified. These messages are customizable to facilitate easy adjustments according to the needs of the respective study.

Cards can be added manually through an input text field. After creation there is also the possibility of renaming or deleting previously added cards. Another way of loading the wanted cards is by importing the dataset via a .csv file. Here any number of cards, separated by a comma or semicolon, are included into the study card list for later sorting. Note that using the import function clears any previously added cards, as this helps unintentionally mixing datasets.

A screenshot of the card sort creation can be viewed in Figure 4.1.

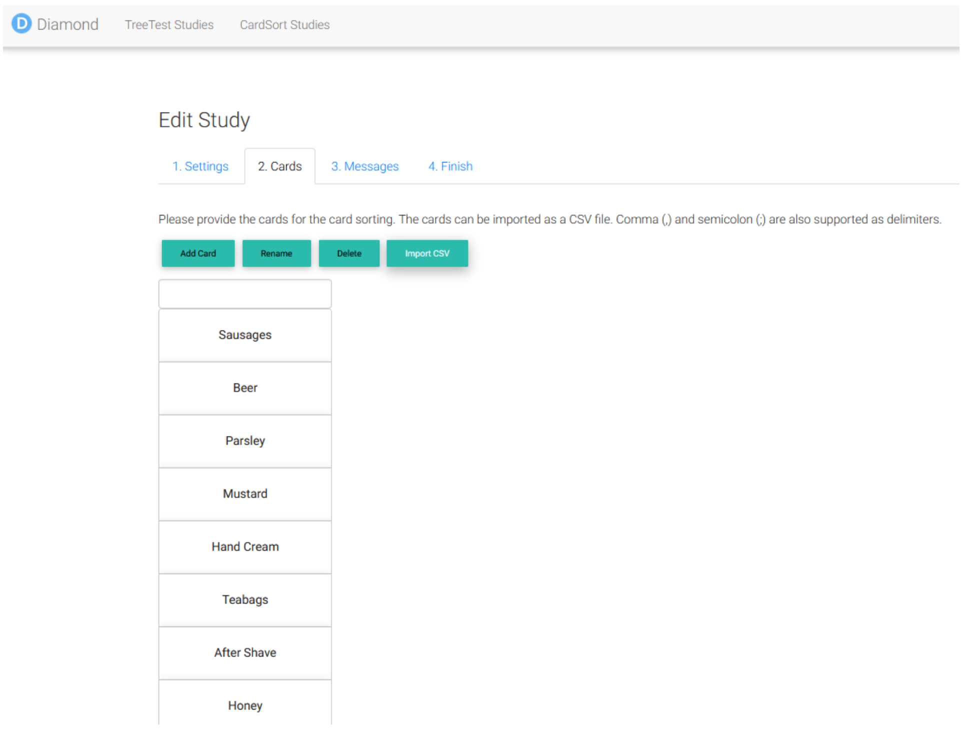
4.2 Taking part in a Study

Once a card sort study has been created, it is possible to share it to users. This can be done via a link and possibly a password to further control the users taking part in the study. Each user needs to specify their name and receives the previously defined welcome message and instructions before attempting the actual sorting.

The card sorting itself is made up of the card list, which is presented to the user on the left of the screen, and a dedicated area for groups that are to be defined by the user. The cards in the card list are stacked vertically and the remaining amount of cards in the list can be seen at the very top.

Currently Diamond only supports open card sorting, which, according to Prof. Keith Andrews, "..is the only true form of card sorting". Therefore all groups need to be defined by the user. This can be done via a input text field. Once a group is added, cards can be dragged from the card list to a group of choice. All groups can be renamed or deleted. Did a deleted group contain cards, these cards are then added back to the card list to be sorted again. A screenshot of the card sorting process can be viewed in Figure 4.2.

Once all cards have been assigned to groups the user has the option to finish their sort. The next step for the user is to explain their mindset during the sorting, to facilitate better analytics of the results. After this, the user is presented with the option to provide general feedback concerning the study.



Please provide the cards for the card sorting. The cards can be imported as a CSV file. Comma (,) and semicolon (;) are also supported as delimiters.

Figure 4.1: The graphical user interface during the adding of cards to a card sort study. [Screenshot was captured by Christopher Oser using Diamond.]

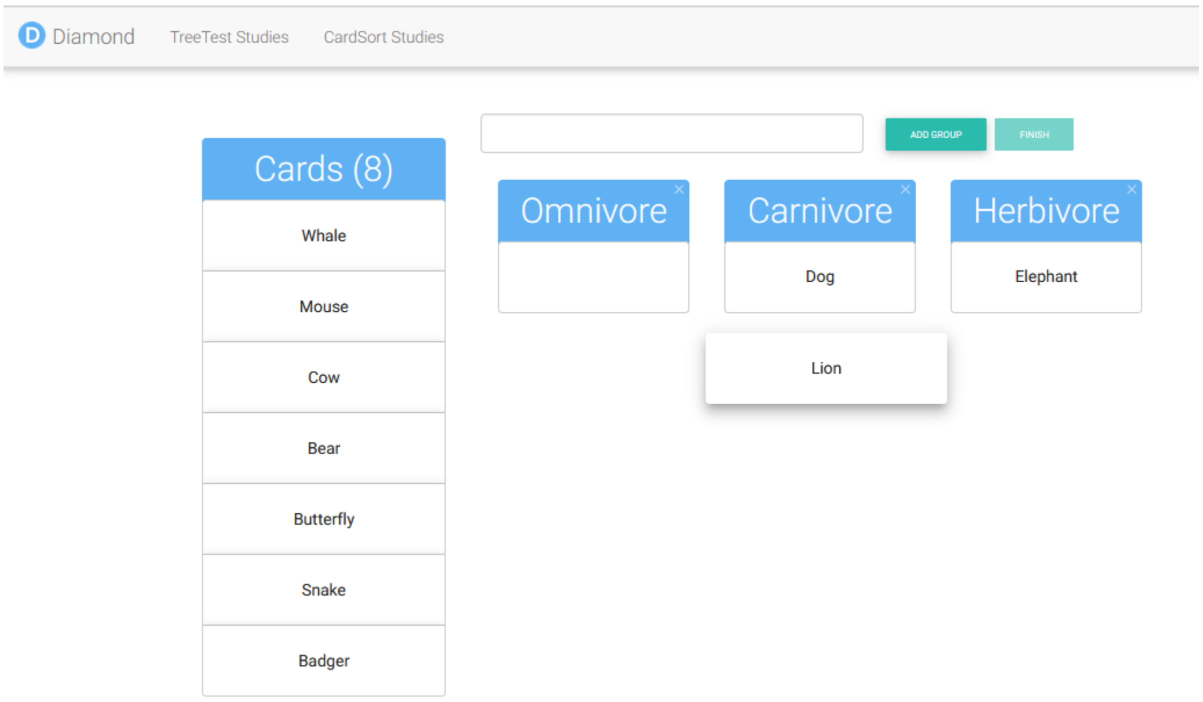
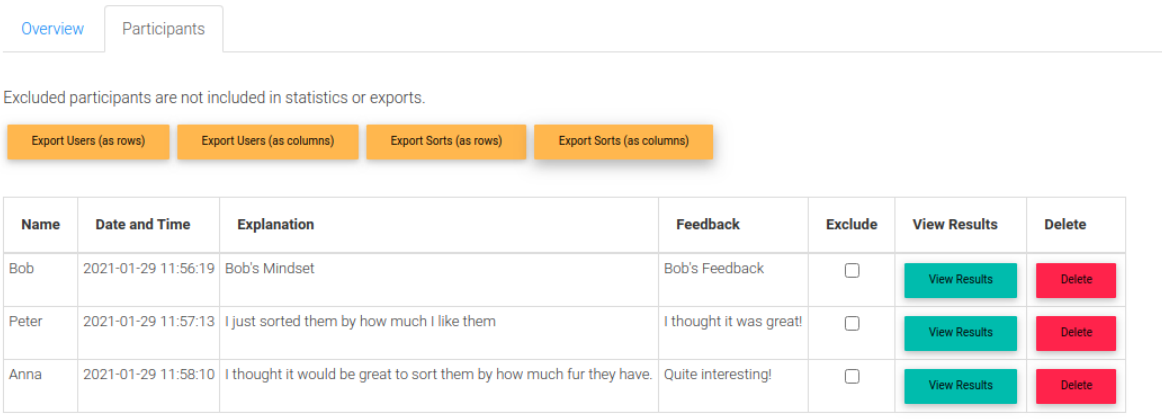


Figure 4.2: The graphical user interface during the sorting of cards by the user. [Screenshot was captured by Christopher Oser using Diamond.]



Excluded participants are not included in statistics or exports.

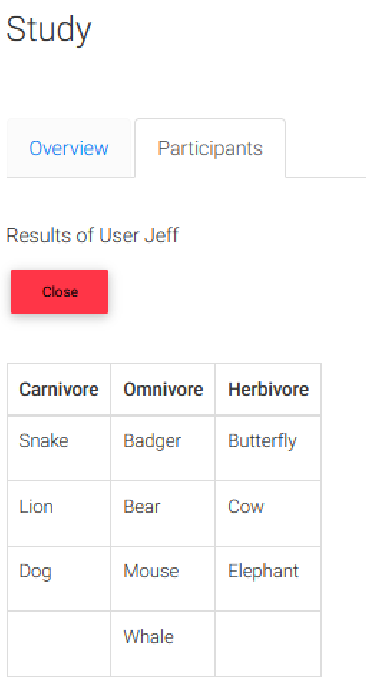
Name	Date and Time	Explanation	Feedback	Exclude	View Results	Delete
Bob	2021-01-29 11:56:19	Bob's Mindset	Bob's Feedback	☐	View Results	Delete
Peter	2021-01-29 11:57:13	I just sorted them by how much I like them	I thought it was great!	☐	View Results	Delete
Anna	2021-01-29 11:58:10	I thought it would be great to sort them by how much fur they have.	Quite interesting!	☐	View Results	Delete

Figure 4.3: The overview over the results of a card sort study in Diamond. [Screenshot was captured by Christopher Oser using Diamond.]

4.3 Evaluating Results

At any point during the study, the study manager, the person who created the study, can view and export the results of the study. The results are composed by the name of the user, the date of the sorting, the sorting results, the mindset and the feedback message. The general overview of the results of a study can be viewed in Figure 4.3. The sorting results are displayed on a separate page for each user, they are made up by a table where each column represents one group. A sample table can be viewed in Figure 4.4.

The results can also be exported for later use in .csv format. There are four different files that are ready for export. The first two files are the user data, comprisedby names, dates, feedback messages and mindsets. The data can be exported as either rows or columns. The third file is the sorting data over all users. Here the data is represented by all cards in the first row, followed by a one user per row group assignment to the respective card in the column. The last file contains the same information as the third one, but stores the data in columns instead of rows. So each user's sorting data is represented by a row of groups.



Results of User Jeff

Carnivore	Omnivore	Herbivore
Snake	Badger	Butterfly
Lion	Bear	Cow
Dog	Mouse	Elephant
	Whale	

Figure 4.4: The table that represents the sorting results of one user. Each group is represented by a column and the respective cards are listed below the first row. [Screenshot was captured by Christopher Oser using Diamond.]

Chapter 5

Conclusion

Now that all the details of the implementation of Diamond have been covered, this chapter concludes the report and ends with some possible improvement for the future and some final remarks.

5.1 Possible Improvements

Due to limited time, some features, which are present in many other card sorting applications, were not implemented. Here is a short list of features we thought of:

- Currently, the only information on the participants that is stored is their name. It could be interesting to give the creator of a study the possibility to create custom questionnaires to extract more information about the participants
- It would be nice to allow users to decide whether they want to create an open, close or hybrid card sorting study. This could be done with only minor additional effort.
- While an overview of the results is provided within the app, for most use cases it is necessary to export the results and then import them into some third application for further evaluation. The tracking and display of statistics such as average sorting duration or how often users changed their minds when sorting cards would be quite helpful and interesting.
- It might be interesting to create sub-groups when sorting the cards. This would require rescaling the size of the groups during the sorting process and abstracting the cards into another component.

5.2 Final Remarks

As of the time of writing this report, Diamond is hosted publicly on Heroku under `https://iaweb-diamond.herokuapp.com/`.

The git repository is `https://github.com/somestudentcoder/Diamond`.

Bibliography

Mehic, Ajdin [2019]. *TreeTest: Online Tree Testing for Information Hierarchies*. Master's Thesis. Graz University of Technology, 2019 (cited on page 1).

Optimal Workshop Ltd. [2021]. *OptimalSort*. 27 Jan 2021. `https://www.optimalworkshop.com/optimalsort/` (cited on page 1).