

tt()
wiki schrijven

tl;dr: naast code schrijven ga je ook
documentatie (proces) bijhouden

<https://github.com/Chazzers/frontend-data/wiki>

Chazzers / frontend-data ✓

Code Issues Pull requests Actions Wiki

Welcome to the frontend-data wiki!

Over here you can read about the proces of making a datavisualisation with D3.js.

To get data from the database the use of SPARQL was necessary. More about SPARQL can be read in the chapters: [05. Getting the data](#) and [08. Creating an interaction](#)

Table of contents

- [1. Introduction](#)
- [2. Concept](#)
- [3. Assignment: Clean dataset](#)
- [4. Installing a taskrunner](#)
- [5. Getting the data](#)
- [6. Experimenting with data visualisations in Excel](#)
- [7. Making my data ready for visualisation](#)
- [8. Visualizing the data](#)
- [9. Creating an interaction](#)

Pages 11

Find a Page...

Home

01. Introduction

02. Concept

03. Assignment: Clean dataset

04. Installing a taskrunner

05. Getting the data

06. Experimenting with data visualisations in Excel

07. Making my data ready for visualisation

08. Visualizing the data

Wiki tabje op GitHub

Process	Process is partially documented in the wiki	Process is properly documented	Choices are evaluated and documented; progress is demonstrated; Work tells a story	Significant progress or iterations are demonstrated; Storytelling principles are applied	 What you did this course is amazing; Teachers are in awe of your progress
----------------	---	--------------------------------	--	--	---

Docs schrijven is onderdeel van de Rubric

Understanding	There is substantial own code; the student can explain the code that exists	The student can explain some parts of their code, how some parts works together, and some technical choices	The student can explain every part of their code, how everything works together, and why patterns are used instead of alternatives; the project is structured logically	The project is complex but can easily be understood; alternatives to patterns covered in class was used that were great choices	 The student deeply understands functional programming and can create their own functional code flows

Docs schrijven is onderdeel van de Rubric

OPZET

FUNCTIONAL PROGRAMMING

- Concept (schetsen, wat doet het)
- Research (programmeer principes)
- Data verzamelen (query'es, endpoints)
- Data opruimen (filter, clean)
- Logboek (stand-ups, 1-op-1 gesprekken)

CONCEPT

Het concept en proces - emmaal x +

https://github.com/emmaoudmajer/Frontend-applications/wiki/Het-concept-en-proces

emmajer / Frontend-applications ✓

Code Issues 2 Pull requests 8 Actions Wiki

Watch Star 0 Fork

Het concept en proces

emmajer edited this page on Oct 31, 2019 · 3 revisions

Tijdens het project bij frontend applications ben ik bezig geweest met het bedenken van een concept voor het nationaal museum van wereldculturen NMWC.

Ik begin met een aantal schetsen maken van verschillende ideeen die in me op kwamen.

fotografie door de jaren heen

zoek een collectie

Waar ben je in geïnteresseerd?

- foto's
- separaten
- Cameras
- Druwtu
- taschen
- onderdelen
- uit welke periode
- 2000- nu
- 1940-2000

Home collecties about 1940-2000

opgestoken

Pages 6

Find a Page...

Home

Ember onderzoek framework

Het concept en proces

reflectie

Week 1 Frontend applications

Week 2 Frontend applications

Clone this wiki locally

https://github.com/emmaal

SCHETSEN, UITLEG, INFO OVER OPDRACHT

RESEARCH

The screenshot shows a GitHub wiki page titled "Functional programming". The page content is as follows:

Functional programming

Eyob Westerink edited this page on Nov 14, 2019 · 3 revisions

Wat is functional programming

Functional programming talen zijn gebaseerd op het principe van het wiskundige functiebegrip. Er is een functie, daar stop je input in en je krijgt er een output uit (een returnwaarde). Functionele talen zijn declaratief en je vertelt als programmeur meer wat je wilt uitrekenen. Over het algemeen bestaan er geen globale variabelen, wat betekent dat de enige manier om informatie door te geven tussen onderdelen van je applicatie, is door het meegeven van parameters aan je functies is. Dit heeft als voordeel dat het de returnwaarde van een functie altijd hetzelfde is wanneer je het opnieuw aanroeft.

Functioneel vs Object georiënteerd

Object georiënteerd programmeren is een techniek binnen programmeertalen. Aan de hand van een object wordt een datastructuur gecreëerd met verschillende gegevens en procedures. dit is een manier van programmeren waarbij logica en gegevens worden georganiseerd rondom objecten. Aan de hand van een object wordt een datastructuur gecreëerd met verschillende gegevens en methodes

De functionele manier van programmeren is een heel andere manier van denken dan objectgeoriënteerd programmeren, functional programming ligt dichter bij de wiskunde. Voor sommige taken is dit een veel logischere manier om een probleem te benaderen.

Functioneel
INPUT --> PROCES --> OUTPUT

Object georiënteerd
MODEL MAP REUSE EXTEND

On the right side of the page, there is a sidebar with the following navigation links:

- Pages 10
- Find a Page...
- Home
- Code snippets
- Concept
- D3 experimentals
- Data query
- Enter, Update, Exit pattern
- Functional programming
- Opschonen enquête data
- Weekly recap
- Zoom

Below the sidebar, there is a link to "Clone this wiki locally" with the URL <https://github.com/Eyot>.

FUNCTIONAL PROGRAMMING, JAVASCRIPT PRINCIPLES

DATA VERZAMLEN

The screenshot shows a GitHub wiki page titled "Getting the data". The page content discusses using SPARQL to extract data from the NMVV database. It includes sections on filtering data by location (Japan) and weapon type. On the right, a sidebar lists ten pages related to the project, and at the bottom right, there's a link to clone the wiki locally.

05. Getting the data - Chazzers

https://github.com/Chazzers/frontend-data/wiki/05.-Getting-the-data

Getting the data

To get data from the NMVV database i had to make use of SPARQL. SPARQL (SPARQL Protocol And RDF Query Language) is a RDF query language that is used to get RDF-based data through queries.

Source: [Wikipedia SPARQL]([nl.wikipedia.org](https://nl.wikipedia.org/wiki/SPARQL) › wiki › SPARQL)

SPARQL query

I wanted to get weapons from Japan from the NMVV database so i had to write a query that did just that.

filtering on Japan

This query first filters all the data of the NMVV on getting only items from Japan like this:

```
<https://hdl.handle.net/20.500.11840/termmaster6917> skos:narrower* ?place .
?place skos:preflabel ?placeName .
```

Filtering on weapon types

Next it starts looking into the data of Japan to find weapon types using:

```
VALUES ?type { "zwaard" "Zwaard" "boog" "Boog" "lans" "Lans" "mes" "mes" "knots" "Piek" "vechtketting" "dolk" "bi
```

Pages 11

- Find a Page...
- Home
- 01. Introduction
- 02. Concept
- 03. Assignment: Clean dataset
- 04. Installing a taskrunner
- 05. Getting the data
- 06. Experimenting with data visualisations in Excel
- 07. Making my data ready for visualisation
- 08. Visualizing the data
- 09. Creating an interaction
- 10. Research D3 & Functional programming

Clone this wiki locally
https://github.com/Chaz

ENDPOINTS, DATASET, FORMAT

DATA OPSCHONEN!

The screenshot shows a GitHub wiki page titled "Opschonen enquête data" with the URL <https://github.com/EyobDejene/functional-programming/wiki/Opschonen-enquête-data>. The page contains two main sections: "Functies die andere functies aanroepen (v1)" and "Opschonen met promises & modules (v2)".

Functies die andere functies aanroepen (v1)

- 1. Instellingen
- 2. Cleandata functie uitvoeren
- 3. Inladen data bestand
- 4. Check lege waardes
- 5. Check of de waardes uit woorden bestaat
- 6. Check of de waarde uit meerdere kleuren bestaat
- 7. Check of de waarde een hashtag bevat

Opschonen met promises & modules (v2)

- 1. Modules inladen
- 2. Instellingen
- 3. Uitvoeren van de cleanData functie
- 4. Fetch data
- 5. Schoone data weergeven

1 Instellingen

Hier heb ik instellingen de gedefinieerd zodat wanneer ik een andere row wil ophalen uit het bestand deze gemakkelijk kan aanpassen.

```
const dataSource = 'data/enqete-raw.csv'  
const rowToEdit = 'Kleur haar (HEX code)'
```

2 Cleandata functie uitvoeren

FILTER, MAP, CLEAN, FORMAT, MODEL

LOGBOEK

The screenshot shows a GitHub repository page for 'Coenmathijssen / frontend-data'. The repository name is 'Coenmathijssen / frontend-data'. The 'Wiki' tab is selected. The main content is titled 'Logboek' and contains a single edit history entry: 'Coenmathijssen edited this page on Nov 28, 2019 · 1 revision'. Below this, a section titled 'Maandag - 18-11' contains the following text:

Vandaag weer een nieuwe dag, een nieuwe opdracht. We kregen te horen dat we twee opties hadden: of je ging verder met je bestaande concept, of je ging met iets nieuws beginnen. De enige vereisten voor deze opdracht zijn: het creëeren van een zinvolle visualisatie (dus de gebruiker is op enige manier getrokken, geïnspireerd of geïnformeerd geraakt door de visualisatie met betrekking tot het museum). Na een kleine brainstorm sessie kwam ik erachter dat ik toch graag door wil met mijn huidige visualisatie. Ik ben ontvredeerd met hoeveer ik nu ben gekomen en ben gemotiveerd om dit uit te werken tot een rond concept. Zo heb ik het concept als volgt uitgewerkt:

Bij het rad van fortuin is een schijf aanwezig met eeuwen (tussen bijv. jaartal 1900 en 1999). Daarnaast zijn er een aantal knoppen aanwezig waarmee een categorie gekozen kan worden. Met een spin gaat de schijf draaien en stopt deze op een willekeurige waarde. Vervolgens worden hier de bijpassende objecten voor opgehaald en geplaatst op de wereldkaart via GEO coördinaten. Om te voorkomen dat er alleen maar objecten op dezelfde plek wordt weergegeven, worden er maar 3 tot 5 items weergegeven per continent.

Ook is er een toggle aanwezig. Deze toggled om óf alle objecten te laten zien, óf alle objecten die al ooit

<https://github.com/Coenmathijssen/frontend-data/pulls?q=is%3Apr+is%3Aopen+sort%3Aupdated-desc> wil zien waar het museum trots op is of om te

The sidebar on the right lists 'Pages' (9) with links to 'Home', 'Concept', 'D3 Map', 'D3 Nest en Transform', 'D3 Tooltip', 'D3 Wheel Of Fortune', 'Enter, Update, Exit', 'Herkansing', and 'Logboek'.

STAND-UPS, GESPREKKEN MET SUPPORT OF DOCENTEN

EXAMPLES

WAT GOEDE VOORBEELDEN

- <https://github.com/ManoukK/functional-programming/wiki/Data-opschonen>
- <https://github.com/Chazzers/frontend-data/wiki>
- <https://github.com/SqueezyDough/frontend-data/wiki/Concept>

TIPS

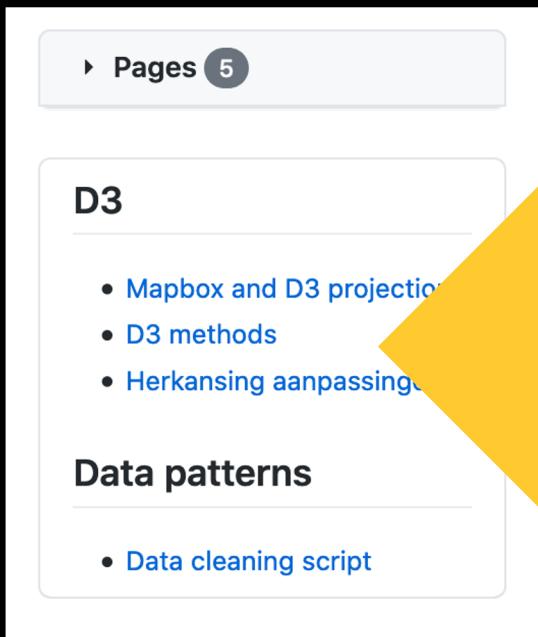
SCHRIJVEN WIKI

- Laat zien wat **je hebt geprobeerd**
- En wat je hebt gedaan om **het op te lossen**
- **Geef inzicht in wat je hebt geleerd**

TIPS

SCHRIJVEN WIKI

- Gebruik kopjes
- Voeg ‘kleine’ code snippets toe
- Engels of nederlands?



Gebruik
subkopjes voor
elk thema

▶ Pages 5

D3

- Mapbox and D3 projection
- D3 methods
- Herkansing aanpassingen

Data patterns

- Data cleaning script

To get data from the database the use of SPARQL was necessary. More about SPARQL can be read in the chapters:
[05. Getting the data](#) and [08. Creating an interaction](#)

Table of contents

- [1. Introduction](#)
- [2. Concept](#)
- [3. Assignment: Clean dataset](#)
- [4. Installing a taskrunner](#)
- [5. Getting the data](#)
- [6. Experimenting with data visualisations in Excel](#)
- [7. Making my data ready for visualisation](#)
- [8. Visualizing the data](#)
- [9. Creating an interaction](#)
- [10. Research D3 & Functional programming](#)

- [1. Introduction](#)
- [2. Concept](#)
- [3. Assignment: Clean dataset](#)
- [4. Installing a taskrunner](#)
- [5. Getting the data](#)
- [6. Experimenting with data visualisations in Excel](#)
- [7. Making my data ready for visualisation](#)
- [8. Visualizing the data](#)
- [9. Creating an interaction](#)
- [10. Research D3 & Functional programming](#)

Home

01. Introduction

02. Concept

03. Assignment: Clean dataset

04. Installing a taskrunner

05. Getting the data

06. Experimenting with data visualisations in Excel

07. Making my data ready for visualisation

08. Visualizing the data

09. Creating an interaction

10. Research D3 & Functional programming

Clone this wiki locally

<https://github.com/Chazzers/frontend-data/wiki>

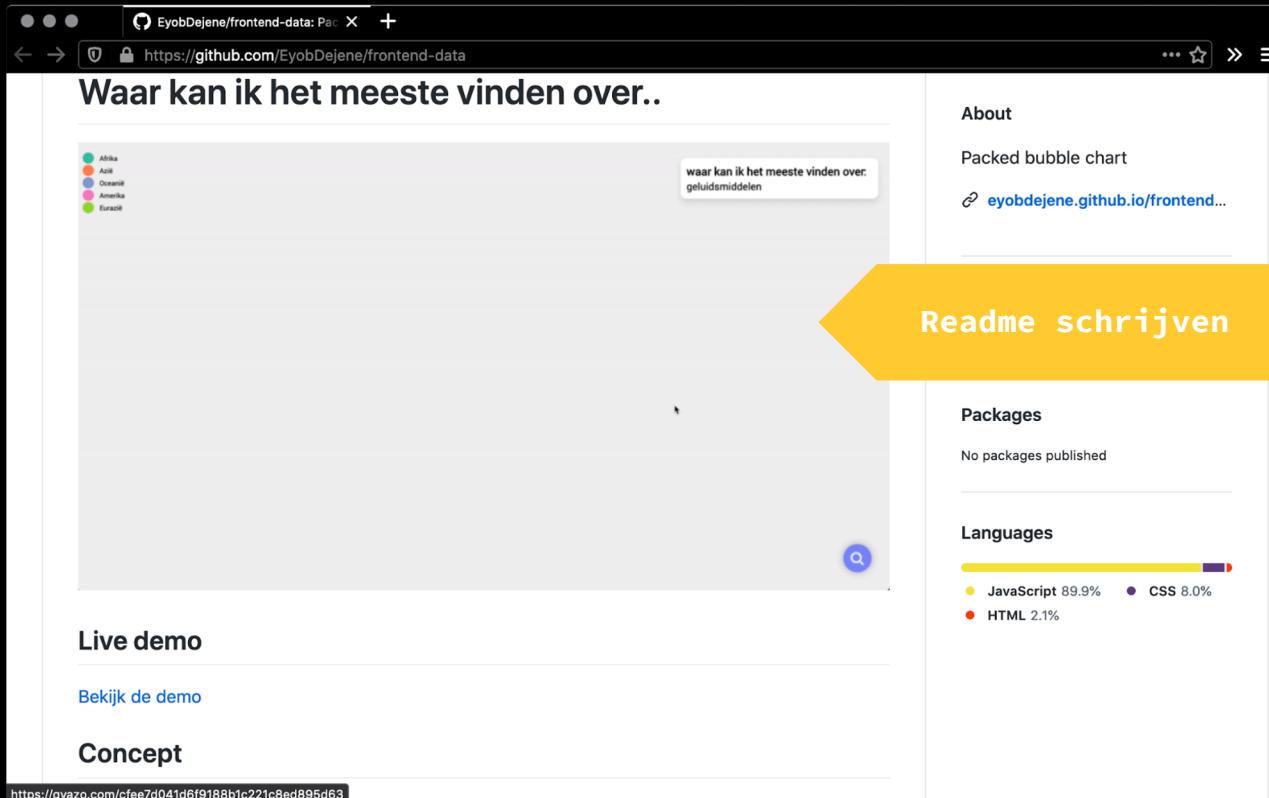
Clone lokaal

Je kan een wiki clonen

TIPS

EARLY AND OFTEN

Ga niet achteraf een wiki schrijven maar net zoals commits; **early and often**. Handig is als je wiki schrijft voor je stand-up, dan vang je twee vliegen in 1 klap. 



Readme.md in root van je repo

TOC

STAPPENPLAN

- **Wat doet je project?** (description)
- Welk **features** zijn er? (features)
- Welke data gebruik je? (dataset)
- Hoe draai ik je project? (install)
- Live link van je project (deploy)
- Credits en license (bronnen)

EXAMPLES

WAT GOEDE VOORBEELDEN

- <https://github.com/EyobDejene/frontend-data>
- <https://github.com/Wiebsonice/frontend-data>

ART OF README

noffle/art-of-readme: Things I've learned about writing good READMEs.

<https://github.com/noffle/art-of-readme>

Art of README

This article can also be read in [Chinese](#), [Brazilian Portuguese](#) and [Spanish](#).

Etymology

Where does the term "README" come from?

The nomenclature dates back to *at least* the 1970s [and the PDP-10](#), though it may even harken back to the days of informative paper notes placed atop stacks of punchcards, "READ ME!" scrawled on them, describing their use.

A reader¹ suggested that the title README may be a playful nudge toward Lewis Carroll's *Alice's Adventures in Wonderland*, which features a potion and a cake labelled "DRINK ME" and "EAT ME", respectively.

The pattern of README appearing in all-caps is a consistent facet throughout history. In addition to the visual strikingness of using all-caps, UNIX systems would sort capitals before lower case letters, conveniently putting the README before the rest of the directory's content².

The intent is clear: "*This is important information for the user to read before proceeding.*" Let's explore together what constitutes "important information" in this modern age.

For creators, for consumers

This is an article about READMEs. About what they do, why they are an absolute necessity, and how to craft them well.

About

Things I've learned about writing good READMEs.

Releases

No releases published

Packages

No packages published

Contributors 31



+ 20 contributors

[HTTPS://GITHUB.COM/NOFFLE/ART-OF-README](https://github.com/noffle/art-of-readme)

LE QUESTIONS?