**Vejledning til at bidrage til dokumentation af GC2/Vidi**

Indhold

[Generelt 2](#_Toc156458684)

[Introduktion til reStructuredText 2](#_Toc156458685)

[Skærmbilleder til dokumentationen 3](#_Toc156458686)

[Engelsk version af dokumentationen 4](#_Toc156458687)

[Arbejde på GitHub 4](#_Toc156458688)

[Strukturering af dokumentationen 5](#_Toc156458689)

Generelt

Dokumentation ligger på GitHub, så det kræver en GitHub konto at bidrage.

Dokumentationen ligger på Mapcentias GitHub konto. Og i Readthedocs visningen linkes over til Github placeringen.

<https://vidi.readthedocs.io/en/latest/>

<https://github.com/mapcentia/vidi/tree/bootstrap5/docs>

Dansk dokumentation laves i markup sproget reStructuredText så filerne hedder \*.rst

Dokumentationen udgives ved hjælp af readthedocs platformen.

Introduktion til reStructuredText

reStructuredText er et markup sprog, dvs. at der bruges bestemte tegn til at opdele eller tilføje formatering til dokumentet.

Dokumentet læses så af en parser, som oversætter formateringen til HTML, som så kan vises i den platform der vælges til publicering.

Det kræves derfor et vist kendskab til reStructuredText tegn og begreber, men man kommer langt ved at kopiere det der findes i rst filerne i forvejen.

Her er et eksempel på hvad der sker.

A screenshot of a computer

Description automatically generated

Der findes meget information om ReStructuredText på nettet, det er meget brugt til teknisk kommunikation.

Her er nogle gode steder at starte.

<https://en.wikipedia.org/wiki/ReStructuredText>

<https://docutils.sourceforge.io/rst.html>

Skærmbilleder til dokumentationen

I dokumentation bruges der skærmbilleder når det er relevant. Det er besluttet, at skærmbillederne laves så enkelt som muligt, med cirkler der er fortløbende nummereret, og der henvises så til i teksten til numrene.

Skærmbillederne gemmes i media mappen på GitHub under docs mappen.

Det anbefales at bruge OpenSource programmet Flameshot som kan hentes <https://flameshot.org/>

Programmet er nemt at bruge, og har en del funktioner, bl.a. en funktion der automatisk laver cirkler med fortløbende nummerering. Der er også let adgang til at tegne pile, skrive tekst osv. Når skærmbilledet er færdigt, gemmes med ctrl+s og der kommer en dialogboks hvor filnavnet kan angives.

Filerne navngives med bindestreg mellem ordene, og gives en sigende tekst på engelsk. F.eks. ”print-complete.png”

Her kan du se hvordan Flameshot ser ud på skærmen.

A screenshot of a computer

Description automatically generated

Opsætning af program til automatisk opstart.

1

Vælg Start-knappen windowslogo, og rul for at finde den app, du vil køre ved start.

2

Højreklik på appen, vælg Mere, og vælg derefter Åbn filplacering. Dette åbner den placering, hvor genvejen til appen er gemt. Hvis der ikke er en indstilling for Åbn filplacering, betyder det, at appen ikke kan køres ved start.

3

Med filens placering åben, skal du trykke på Windows-tasten icon-windowslogo + R, skrive shell:startup, og derefter vælge OK. Dette åbner Start-mappen.

4

Kopiér og indsæt genvejen til appen fra filplaceringen til Start-mappen.

Engelsk version af dokumentationen

Der laves en engelsk udgave af dokumentationen ved at bruge Gettext

Det betyder, at der automatisk oprettes en engelsk version af den danske dokumentation. Den gemmes i \*.po filer. Dvs. at det er de danske tekster der gemmes. De danske tekster skal så oversættes og når po filerne er oversat, så vises indholdet i engelsk version. Skærmbilleder genbruges til begge versioner.

Når man ændrer en af de danske .rst filer eller opretter en ny. Så skal man køre to Python kommandoer, for at få .po filerne opdateret. Det vil nok i praksis være Martin der skal have besked om at gøre det.

Arbejde på GitHub

Der skal arbejde med dokumentationen på GitHub. Så det er nødvendigt at vide lidt om hvordan der arbejdes med tingene på GitHub.

Det kan enten gøres via browseren på GitHub.com eller Git kan installeres på egen computer.

Hvis man vil arbejde med det over længere tid, er det klart en fordel at installere Git på egen computer og så forke en udgave af f.eks. Vidi branchen til sin egen computer, og så kan man arbejde med dokumentationen helt lokalt. Når man så mener at man har lavet det færdigt man vil arbejde med (gerne i flere forskellige filer og tilhørende screenshots), opretter man et samlet pull request. Martin vil så gennemse det er der lavet, og trække ændringerne ind i hoved branchen.

Det foregår på samme måde hvis man arbejde med det i browseren, men så skal der oprettes et pull request hver gang man gemmer et dokument.

Mere information om at arbejde på GIT

<https://codegym.cc/da/groups/posts/da.379.kom-godt-i-gang-med-git-en-omfattende-guide-til-nybegyndere>

Git på din egen computer

For at komme i gang med Git på din egen computer skal du downloade programmet [her](https://git-scm.com/downloads).

Derudover skal du have en Github konto. Du kan oprette en konto [her](https://github.com). Det er gratis at oprette sig som bruger.

Når du har installeret Git skal du tilføje dit navn og E-mail. Dit navn og E-mail vil være synligt på Github, når du har skubbede (push) dine ændringer op.

Åben Git Bash, hvor du kan skrive følgende.

**Tilføj dit navn**

git config --global user.name "Mona Lisa"

**Bekræft dit brugernavn er rigtigt**

$ git config --global user.name

> Mona Lisa

**Tilføj din E-mail**

git config --global user.email "[mona-lisa@eksempel.com](mailto:mona-lisa@eksempel.com)"

**Bekræft din E-mail er rigtig**

$ git config --global user.email

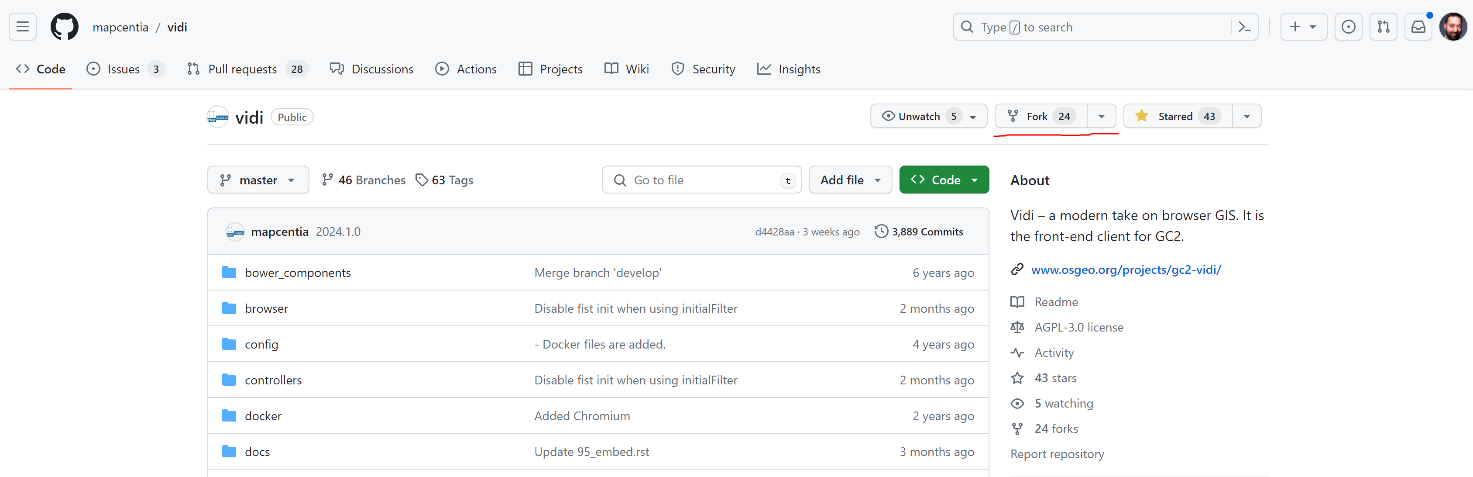
> [mona-lisa@eksempel.com](mailto:mona-lisa@eksempel.com)

Så er Git sat op på din computer, og du er nu klar til at gå videre. I det følgende eksempel tager vi udgangspunkt i dokumentation til Vidi. Du kan finde Vidi [her](https://github.com/mapcentia/vidi) og [GC2](https://github.com/mapcentia/geocloud2) her.

1. Log ind på [Github](https://github.com).

2. Du skal starte med at gafle (forke) Vidi over til din egen Github konto. Det at forke betyder at kopiere. Det gør du, ved at gå over hvor Vidi’s kode ligger. Det kaldes et repository. Så tryk på linket https://github.com/mapcentia/vidi

3. Tryk på ”Fork” for at kopiere Vidi til din egen konto.



4. Gå tilbage til din egen konto ved at trykke på Github logoet (katten) i øverste venstre hjørne.

5. Klon din nye kopi

For at få koden fra Github ned på din computer kloner du repositoriet.

Så på dit dashboard i Github klikker du nu på linket til din Vidi kopi (fork). Derefter klikker du på den grønne knap, hvor der står "Clone or Download". Klik på HTTPS og kopier url'en.

På din egen pc åbner du Git Bash og skriver (paste urlen ind du lige har kopieret efter git clone):

git clone https://github.com/DIT\_GITHUB\_BRUGERNAVN /vidi.git

F.eks.

git clone https://github.com/BoMarconiHenriksen/vidi.git

6. Naviger til dit lokale repository

Da du klonede repositoriet blev der lavet en subfolder, hvor Vidi ligger i. Så for at navigere til Vidi skriver du:

cd vidi

7. Synkroniser din fork

Hvis du lige har forkede (kopierede) Vidi, så vil den være opdateret med den seneste kode, og så kan du springe dette trin over.

Er det derimod et stykke tid siden du har forkede Vidi, så er det vigtigt, at synkroniser din kopi med Mapcentia’s Vidi. Når du synkroniser bliver de ændringer Martin har tilføjet kopierede over til din udgave af Vidi.

Det er vigtigt at huske at synkronisere inden du ændrer i dokumentationen ellers kan der opstå konflikter, hvis der bliver ændret i de samme filer.

Gå tilbage til din fork (kopi) af Vidi på Github. Klik på knappen, der hedder ”Sync fork”, og derefter på ”Update branch”.

8. Hent de seneste ændringer til din computer

Hvis du lige har forkede (kopierede) Vidi, så vil den være opdateret med den seneste kode, og så kan du springe dette trin over.

For at få de seneste ændringer ned på din computer skal du ”trække” (pull) dem ned. Det gør du ved at gå tilbage til Git Bash, og skrive:

Git pull

9. Lav en branch

Før du går i gang med tilføje dokumentation, så skal der laves en branch. En branch gør det lettere at bidrage til et projekt, og det du tilføjer i branchen er isoleret fra ”master” branchen. Det som ligger i ”master” branchen, er det som kommer videre ud i produktion. Det vil vi helst ikke ændre i direkte.

Føst laver vi en branch og derefter skifter vi til den branch. Giv branchen et beskrivende navn ift. det som du vil tilføje.

Lav en branch:

git branch BRANCH\_NAVN

Eksempel:

git branch tilføj-dokumentation-til-vidi-api

Skift til din branch

Git checkout BRANCH\_NAVN

Git checkout tilføj-dokumentation-til-vidi-api

10. Tilføj dokumentation

Du er nu klar til at tilføje dokumentation. Det kan du gøre i en tekst editor f.eks. [Visual Studio Code](https://code.visualstudio.com/). Den er gratis at bruge.

11. Skub ændringer op til din fork

Når du er færdig med dine ændringer skal de tilføjes, og derefter skubbes op i din fork (kopi) på Github. Det gør du ved at skrive.

git add .

git commit -m "EN\_KORT\_BESKRIVELSE\_AF\_DET\_DU\_HAR\_ÆNDRET"

git push

add . fortæller Git at du vil tilføje alle dine ændringer, og commit tager et snapshot af dine ændringer og tilføjer en besked. Push skubber dine ændringer op til din fork (kopi) på Github.

12. Lav et pull request

Vi skal nu have ændringerne videre over i Mapcentia’s Vidi repository. Så går over til din Vidi fork (kopi) på Github. Klik på ”Contribute” og derefter på ”Open a pull request”.

Du vil nu se en side, hvor du kan se forskellen på din fork og Mapcentia’s repository. Klik på ”Create pull request”.

Herefter kommer du til en side, hvor du kan skrive en titel og beskrivelse. Skriv en titel og en kort beskrivelse af, hvad du har tilføjet/ændret. Derefter klikker du på ”Create pull request”.

Det er nu op til Martin at acceptere dit pull request. Når Martin har accepteret dit pull request så husk at synkronisere din fork.

**Opsummering**

Nogen ting skal gøres en gang og andre ting skal gøres hver gang du vil tilføje noget nyt. Så punkt 1 til 6 skal gøres en gang, og punkt 7 til 12 gøres hver gang du vil tilføje ny dokumentation.

Vær opmærksom på at dokumentation findes i den mappe, der hedder docs.

Strukturering af dokumentationen

Den laves relativt stramt efter strukturen i programmerne. Det vil give den mest logiske tilgang til at finde tingene.

Forslag struktur dokumentation

Strukturforslag Vidi

Kom godt igang med Vidi udgår.

Vidi brugerflade

Lagstyring

Måleværktøjer

Tegne værktøjer

Projekter

Print

Redigering af data

Signatur

Vidis konfigurations muligheder

Konfiguration i kontrolcenter

Standard query string

Systemkonfiguration

Der er også et stort afsnit om lagopsætning fra GC2 under Vidi. Der bør linkes over i GC2 istedet.

Indlejring af Vidi i andre hjemmesider

Anden brug af Vidi

Vidi API

Extensions

Standard

Ikke standard

Struktur forslag GC2

Kom godt igang med gc2 udgår. Erstattes evt med en introduktion til begreber.

Det bliver i stedet en gennemgang opbygget efter strukturen i admin tool skal kontrolcenter og dermed skal konfigurationer vel være under GC2. Der kan jo linkes fra VIdi

Kontrolcenter

Opret ny database

Schemaer

Sub-brugere

Konfigurationer

Admin tool

Kort fanen

Lag oversigt

Nyt lag evt link til lag gennemgang under databasefanen

Reload

Lag Muligheder

Class wizard

Manuel styling

KLasser

Indstillinger

Tile cache

Signatur

expressions

Base

Symbols

Labels

Start edit

Hurtig tegn

Kort vinduet

Database fanen

SCHEMA vinduet øverste del

nyt lag (stor gennemgang af lag muligheder)

Flyt lag

Omdøb lag

Slet lag

Kopier egenskaber

Schema

Nyt Schema

Lag vinduet nederste del

Struktur

Data

Elasticsearch

Fanerne i toppen

Privilegier

Workflow

Avanceret

Tjenester

Tags

Meta

Ryd tilecache

workflow fanen

Scheduler fanen

Oplysninger om kørte schedulerjobs

Log fanen

Andre muligheder i CC2

Styling med Qgis

Tilføj extension

Traccar extension