

Programming Basics

Heute

- Dokumentation

PAUSE

- Softwareentwicklung
 - Versionsverwaltung
 - Gute Versionsnachrichten

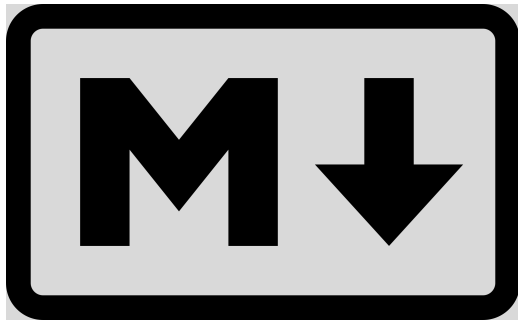
PAUSE

- Lernpfade Git

Dokumentation

- Erkenntnisse und Hilfestellungen sollen festgehalten werden.
- Darstellung von Programmcode bspw. mit Word ist unpraktisch, da keine visuellen Hilfestellungen (Einrücken, farbliche Hervorhebung) gegeben werden.
 - ☐ Für die Dokumentation von Code wird deshalb oft die Auszeichnungssprache Markdown verwendet.
- Dokumentation direkt im Code als Kommentar

Auszeichnungssprache (markup language)

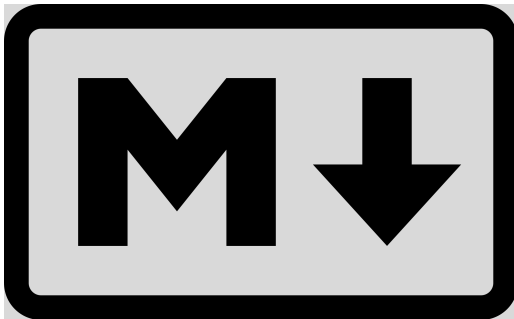


Maschinenlesbare Sprache für die Gliederung und Formatierung von Text.

Eigenschaften (bspw. Link-Adresse) oder **Darstellungsformen** (bspw. *kursiver* Text) werden im Dokument als Text festgehalten.

Bspw. HTML, XML, MathML,... und Markdown

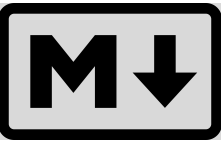
Markdown



Nicht nur **Maschinenlesbar** sondern auch **Menschenlesbare** Sprache.

Beispiel:

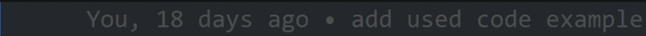
```
1  # VS Codium + Python
2  ## Code Runner
3  VS Codium Plugin
   *formulahendry.code-runner*
4
5  Python code in einem beliebigen File mit
   `.py`-Dateiendung schreiben, anwählen und
   ausführen.
6
7  Ist kein Text ausgewählt, wird das ganze
   File ausgeführt.
8
9
10 ![Code Runner](../media/code-runner.gif)
11
```



Markdown (*.md-Files) + VS Code

```
code-runner.md - programming-basics - Visual Studio Code

docs > vscodium > code-runner.md > abc# VS Codium + Python > abc## Code Runner

You, 5 minutes ago | 1 author (You)
1 # VS Codium + Python
2 ## Code Runner
3 VS Codium Plugin *formulahendry.code-runner*
4
5 Python code in einem beliebigen File mit .py-Dateiendung schreiben, anwählen und ausführen.
6
7 Ist kein Text ausgewählt, wird das ganze File ausgeführt.
8
9  You, 18 days ago • add used code example
10 ![Code Runner](../../media/code-runner.gif)
11
12 Das gezeigte Beispiel scripts.py ist unter [code/scripts.py](../../code/scripts.py) abrufbar.
13
14 [Zurück zur Übersicht](../../README.md)
15
```



Überschriften

```
# Titel 1
```

```
## Titel 2
```

```
### Titel 3
```

```
#### Titel 4
```

```
##### Titel 5
```

```
##### Titel 6
```

→ gibt Möglichkeiten,
Darstellungsfarmen zu verändern

Titel 1

Titel 2

Titel 3

Titel 4

Titel 5

Titel 6



Auflistungen

- Unsorted
- Lists
 - with nesting
 - and more nesting

1. Ordered
2. Lists
 - 1. with nesting
 - 1. and more nesting

entsteht
geordnete
Liste

- Unsorted
- Lists
 - with nesting
 - and more nesting

1. Ordered

2. Lists

i. with nesting

a. and more nesting

→ far mehr
auch wieder
veränderbar



Textmarkierungen

```
This Text is *italic*, **bold**,  
~~strikethrough~~
```

This Text is *italic*, **bold**,
~~strikethrough~~

als Referenz
auf externe
Dateien

Bilder

↳ Bilder werden
eingebunden

HTML

src = source

Nachteil von Pixel → untersch.
Geräte

↳ auch per
drag & drop
möglich

gleiche Logik bei
Links

`<!-- displays the full image up to 100%
page width -->`

`![Alternativ Text](./random.png)` Anzeige, was
das Bild ist

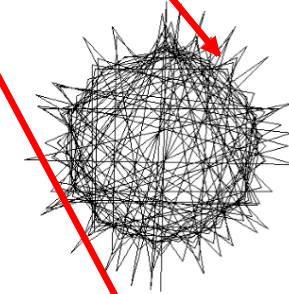
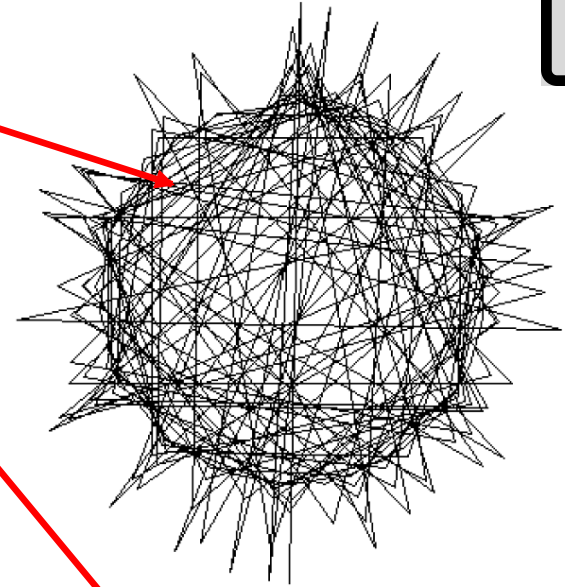
`<!-- specify width (or height) yourself as
percentage -->`

``

`
 <!-- manual linebreak -->`

`<!-- specify width (or height) yourself in
pixels -->`

``



Pfade

— Festplatte = großer Speicher, über Betriebssystem Ordner anlegen

- Anleitung zum Auffinden eines Ortes (Ordners oder Datei) auf einem Datenträger
↳ Abstraktionen, nicht realer Speicher
- Beispiel: **Benutzer > mueller > Dokumente > beispiel.md**
- Als Trennzeichen wird statt einem >
 - Bei Windows ein Backslash \
 - Bei OSX/Linux: ein Slash /
 } braucht Abstraktion zum Übersetzen

verwendet.

Pfade

Navigatoren `.` und `..`

- `.` Verweis auf aktuellen Ordner

Benutzer / mueller / `.` / `.` / Dokumente / `.` / beispiel.md
ergibt
Benutzer / mueller / Dokumente / beispiel.md

- `..` Verweis auf den vorhergehenden Ordner

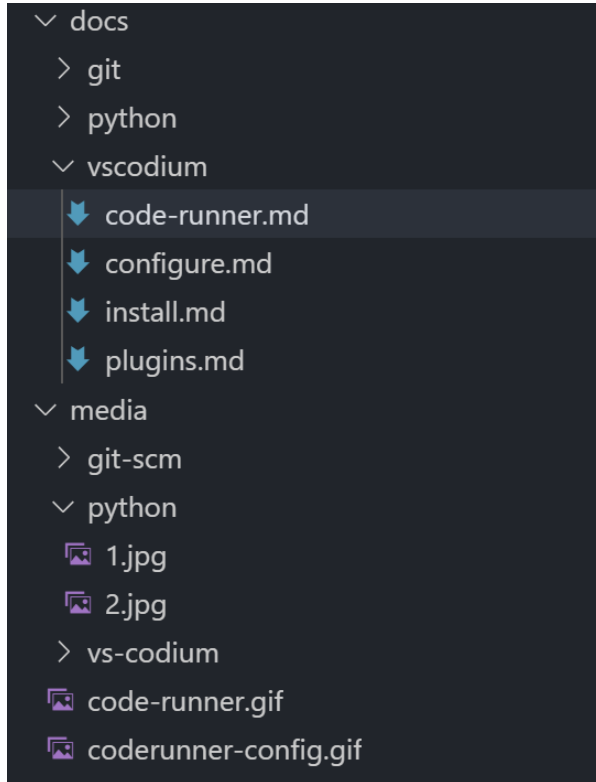
↳ relativer Pfad

Benutzer / mueller / Dokumente / `..` / Bilder / beispiel.png
ergibt
Benutzer / mueller / Bilder / beispiel.jpg

↳ absoluter Pfad

Relative Pfade beginnen mit . oder ..

↳ relativ zur aktuellen Datei



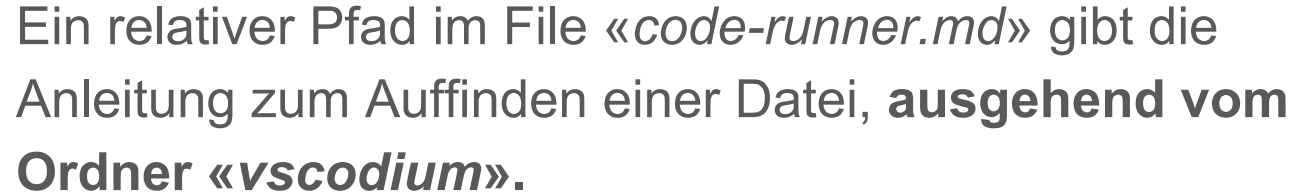
Ein relativer Pfad im File «*code-runner.md*» gibt die Anleitung zum Auffinden einer Datei, **ausgehend vom Ordner «vscodium»**.

In Markdown und Python wird für **relative Pfade** immer ein *Slash /* als Trennzeichen verwendet.

Wie lautet der relative Pfad im File «*code-runner.md*» zur Datei:

- plugins.md
- code-runner.gif
- 2.jpg

↳ Phase testen!



In Markdown und Python wird für **relative Pfade** immer ein *Slash /* als Trennzeichen verwendet.

Wie lautet der relative Pfad im File «*code-runner.md*» zur Datei:

- plugins.md **./plugins.md**
- code-runner.gif **../..../media/code-runner.gif**
- 2.jpg **../..../media/python/2.jpg**



Links

```
1  [SRF](www.srf.ch)
2
3  [Links](#links)
4
5  [Anchors](#anchor-links)
6
7  [Relative](./titles#title5)
8
9
10
11
12  # Links
13
14
15  ## Anchor Links
16
```

SRF

Links

Anchors

Relative —> andere Dokumente


Links

Anchor Links



Code

Sprache



```
```py
from ispw_functions import *
forward()
right()
forward()
forward()
```
```

or inline with single backticks like `forward()`.

```
from ispw_functions import *
forward()
right()
forward()
forward()
```

or inline with single backticks like `forward()`.

Markdown Cheat Sheets

- <https://daringfireball.net/projects/markdown/>
- <https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet>
- <https://docs.github.com/de/get-started/writing-on-github>

Kommentare im Python Code

Einzeilige Kommentare

```
from turtle import forward, right
# import all functions from math library
from math import *
```

```
def forward(step: int = 10):
    '''Moves the pen forward

    Parameters
    -----
    step : int, optional
           pixels to go forward
    ...
    forward(step)
```

Mehrzeilige Kommentare,
«Docstrings»

↳ Beschreibung, was Funktion macht

! Alles, was hochgeladen wird, soll auch dokumentiert sein !

Auftrag: Dokumentation für UML-Python Beispiel erstellen

(Mermaid Code-Block:

```
```mermaid
```

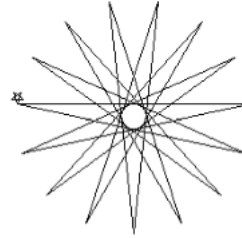
```
```
```

```
)
```

15er Stern

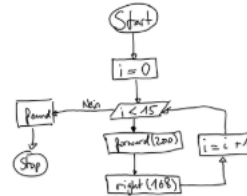
Ziel

Ein Pythonprogramm welches den folgenden Stern zeichnet:



Bei einem 15-Zack Stern beträgt der Innenwinkel eines Spitzes 12° . Nach jeder gezeichneten Linie muss also 168° nach rechts gedreht werden.

UML



Python

Folgender Code muss kopiert und 15 mal untereinander eingefügt werden:

```
forward(200)  
right(168)
```

Die letzte Zeile des Programms ist `found()`. *exit on click*

Pause



"FINAL".doc



FINAL.doc!



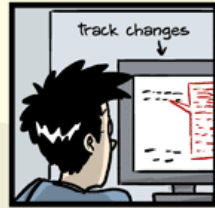
FINAL_rev.2.doc



FINAL_rev.6.COMMENTS.doc



FINAL_rev.8.comments5.
CORRECTIONS.doc



FINAL_rev.18.comments7.
corrections9.MORE.30.doc

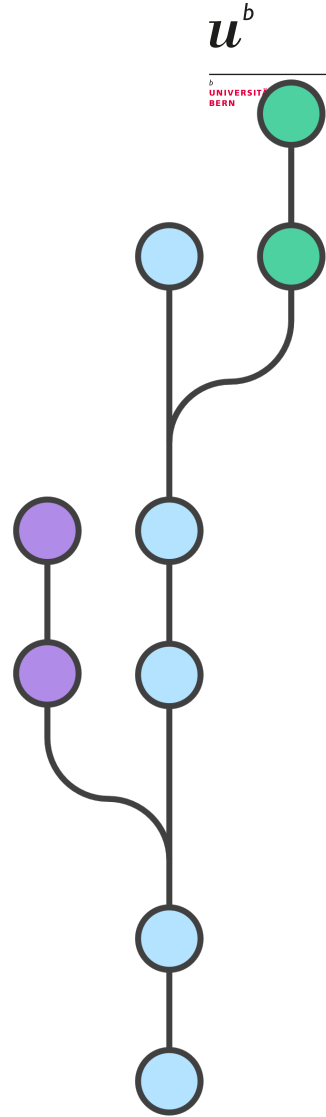


FINAL_rev.22.comments49.
corrections.10. #@\$%WHYDID
ICOMETOGRADSCHOOL????.doc

Versionierung

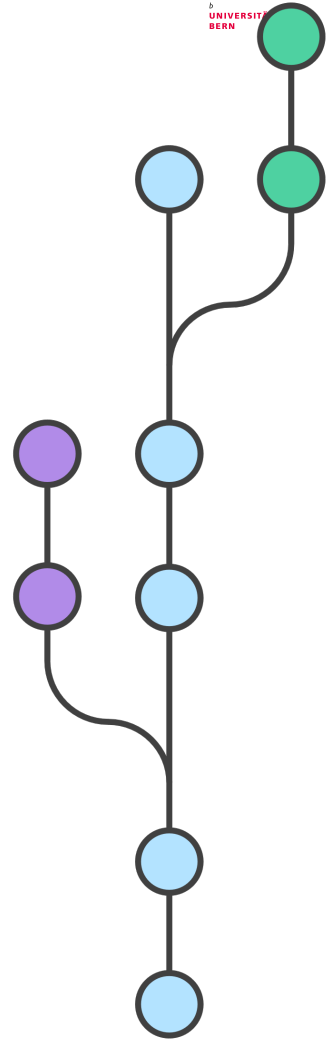
Ziele:

- Nachvollziehbarer Änderungsverlauf → was für Änderungen & wann?
- Absicherung vor Datenverlust
- Wiederherstellung früherer (funktionierender) Versionen
- Förderung von «Modularem Denken»
 - ❓ Zusammengehörige Änderungen als eine Änderung vermerken



Versionsverwaltung mit git

- Aufzeichnung von Änderungen gegenüber der Vorversion (Zeile n hinzugefügt, Datei X entfernt)
- Automatische Zusammenführung verschiedener Bearbeitungsäste
- Verteilung der Datenspeicherung → Ausfallssicherheit



Versionsverwal

- Aufzeichnung von Änderungen (Datei hinzugefügt, Datei gelöscht)
- Automatische Zusammenführung von Bearbeitungsständen
- Verteilung der Daten

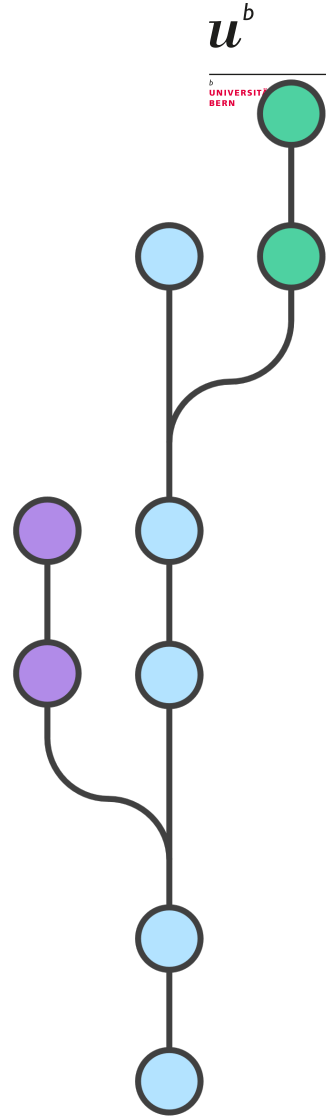
THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.



on (Zeile n



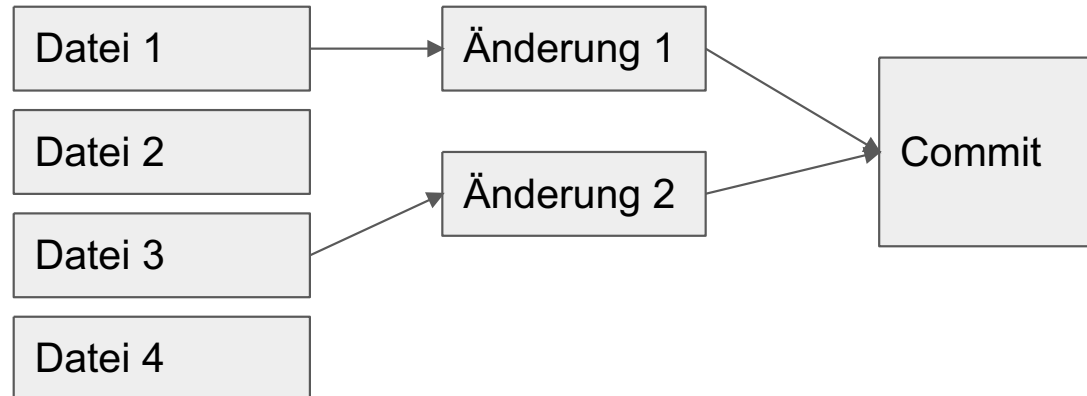
u^b

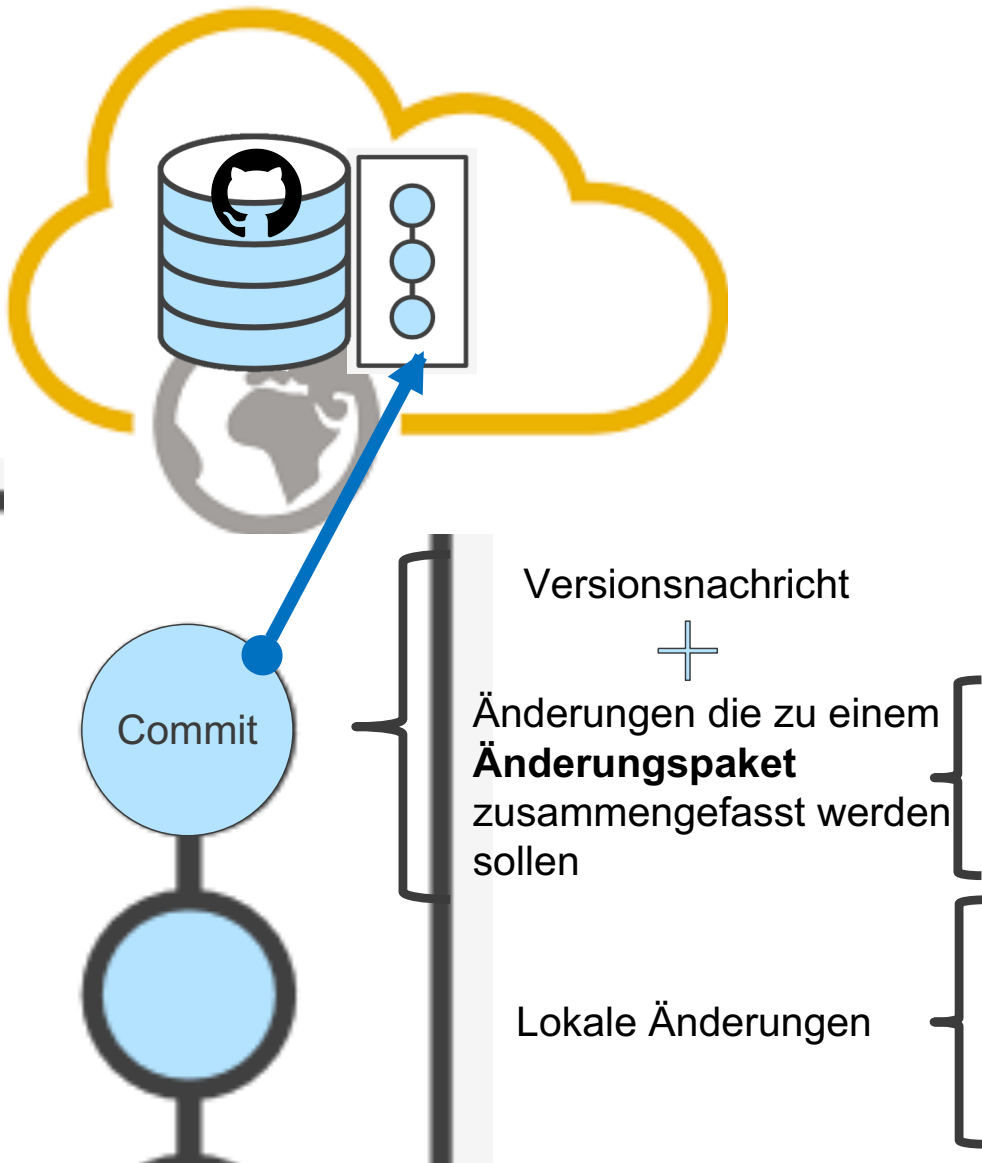
UNIVERSITÄT
BERN

Idee

3 Behälter

- **Arbeitskopie**, die Version der Dateien, die gerade verändert wird
- **Stage**, alle Änderungen, die zu einem Änderungspaket zusammengefasst werden sollen
- **Commit**, ein Änderungspaket mit *Versionsnachricht*





▼ T2GIT

- 001_variables.md
- circle.py M
- hexagon.py U
- star.py M

SOURCE CONTROL: GIT ✓ ↺

import turtle library

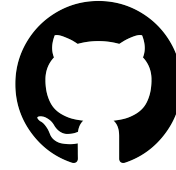
STAGED CHANGES 2

- circle.py M
- star.py M

CHANGES 2

- hexagon.py U
- star.py M

Arbeitsfluss Git + GitHub



- Änderungen an Quellcode/Dokumentation machen
- Sobald ein *Feature* fertig ist, alle dazu nötigen Dateien in **Stage** aufnehmen
- **Commit** mit Versionsnachricht erstellen
- *Repositories* können mit dem lokalen Git-Projekt synchronisiert werden:
 - *pull*: Holt Commits vom «origin» (bei uns von GitHub)
 - *pull from...*: Auswählen von wo Änderungen geholt werden; z.B. von **origin** oder von **ispw**
 - *push*: Lade lokale Commits auf GitHub hoch
- Bei jedem *push* können auf GitHub Aktionen ausgeführt werden – bspw. Tests laufen lassen, ob der hochgeladene Code korrekt läuft.

Versionsnachrichten

| Graph | Description | Date |
|-------|---------------------------------------|------------------|
| | master HAAAAAAAAAAAAANDS | 9 Sep 2019 20:24 |
| | MY HANDS ARE TYPING WORDS | 9 Sep 2019 10:14 |
| | AAAAAAAAAAAAA | 9 Sep 2019 00:24 |
| | Merge branch 'CIRCLE' | 9 Sep 2019 01:24 |
| | CIRCLE ASDFADSF | 5 Sep 2019 23:33 |
| | HERE HAVE CODE | 5 Sep 2019 23:25 |
| | MORE CODE | 5 Sep 2019 22:52 |
| | CODE ADDITIONS/EDITS | 5 Sep 2019 21:29 |
| | HEXAGON BUGFIXES 2 | 5 Sep 2019 21:22 |
| | HEXAGON BUGFIXES | 5 Sep 2019 20:21 |
| | HEXAGON ASSIGNMENT | 5 Sep 2019 20:01 |
| | SETUP PYTHON SKELETON FOR ASSIGNMENTS | 5 Sep 2019 19:23 |
| | ADD UML FOR HEXAGON | 5 Sep 2019 19:19 |

AS A PROJECT DRAGS ON, MY GIT COMMIT
MESSAGES GET LESS AND LESS INFORMATIVE.

Git commit messages

- Englisch oder deutsch, aber immer gleich für ein Repository

Subject: 50 - 72 Zeichen, keine Satzzeichen

Bsp. «wait for userinput before quit for better usability»

Body: beschreibt auf neuer Zeile *was* und *wieso* die Änderung gemacht wurde. Kann oft weggelassen werden.

- Hilfestellung für die **Titelzeile:**
 - «If applied, this commit will...»
[?] *update the getting started documentation*
 - **Imperativ verwenden:**
 - «Putz dein Zimmer»
 - «Schliess die Türe»
 - «Entferne überflüssige Funktion 'run'»
 - Kein Satzzeichen

<https://chris.beams.io/posts/git-commit/>

Git(hub) : Arbeitsaufträge (in class)

Arbeiten im Lernpfad und (mit Markdown) Erkenntnisse festhalten.

- Versionierung & Dokumentation auf github.com

<https://learn.microsoft.com/de-de/training/paths/collaborate-markdown-github-pages/>

- Github und VS Code

<https://code.visualstudio.com/docs/sourcecontrol/overview>

- Mehr:

<https://learn.microsoft.com/de-de/training/paths/manage-project-lifecycle-github/>

Pause



Woche 2: Arbeitsaufträge

(Nach-)Arbeiten im Lernpfad und (mit Markdown) Erkenntnisse festhalten.

- Versionierung & Dokumentation auf github.com

<https://learn.microsoft.com/de-de/training/paths/collaborate-markdown-github-pages/>

- Github und VS Code

<https://code.visualstudio.com/docs/sourcecontrol/overview>

- Mehr:

<https://learn.microsoft.com/de-de/training/paths/manage-project-lifecycle-github/>

- Python-Lernpfad (vor «Funktionen» aufhören)

<https://learn.microsoft.com/de-de/training/paths/beginner-python/>

- Arbeiten im Buch (Kapitel 1-2) und (mit Markdown) Erkenntnisse festhalten

<https://ssr-2024.github.io/ssr2024/>

(this commit will) solve exercise 1
by assigning the correct strings
to each variable

welche Aufgabe bearbeitet X kerninhalt