

Dokumentation

M122 LB1 «AutoPusher» | Valentin Wettstein | Ap18b

Inhaltsverzeichnis

[Umgebung & Technologie 3](#_Toc66795690)

[Funktion & Ablauf 3](#_Toc66795691)

[Struktogramm 3](#_Toc66795692)

[Entry Point 3](#_Toc66795693)

[Konstanten / Dependencies 4](#_Toc66795694)

[Logik 4](#_Toc66795695)

[Schlusswort 5](#_Toc66795696)

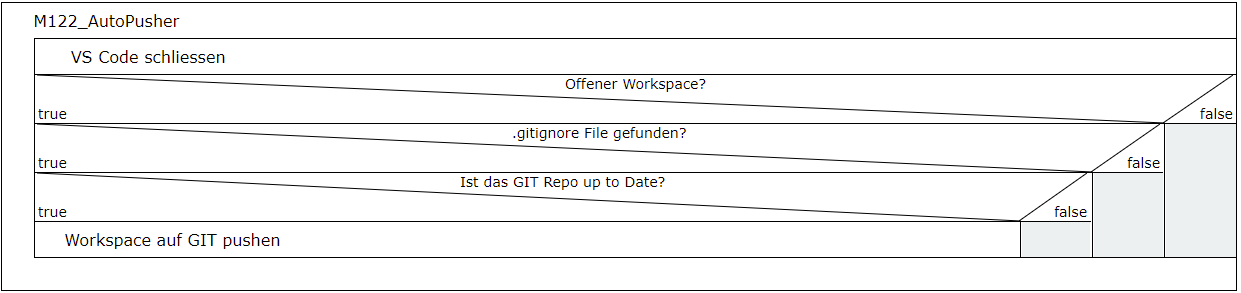
# Umgebung & Technologie

Da dieses Skript mit Visual Studio Code interagiert, machte es am meisten Sinn dieses in Form einer Visual Studio Code Extension zu programmieren, nicht nur wegen der Integration in das Programm sondern auch wegen des einfachen Installationsprozesses.

Als Sprache wählte ich Javascript (Node JS), da diese für die Aufgabe in meinen Augen am besten geeignet ist und mir viele Libraries zur Verfügung stehen.

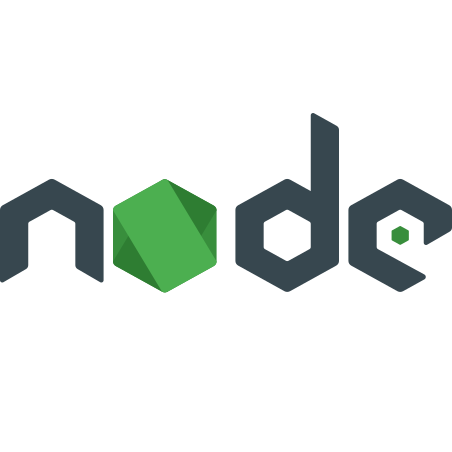
# Funktion & Ablauf

## Struktogramm



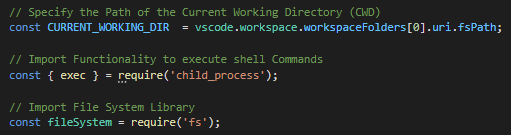
## Entry Point

Der Entry – Point des Projektes ist die deactivate() – Methode im File «extension.js». Diese Funktion wird aufgerufen, sobald Visual Studio Code geschlossen wird. In dieser Funktion befindet sich die gesamte Programmlogik.



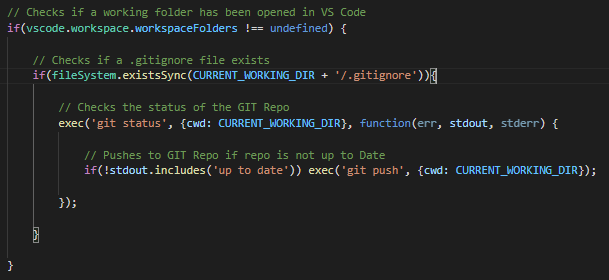
## Konstanten / Dependencies

Für die Funktion von AutoPusher werden das Current Working Directory (CWD) sowie zwei Dependencies benötigt; Exec lässt Shell – Commands asynchron ausführen. Diese wird benötigt, da ich keine funktionsfähige GIT – Library für Node JS gefunden habe. FS wird benötigt, um mit dem FileSystem zu interagieren.



## Logik

Als Erstes schaut das Programm, ob man im Visual Studio Code überhaupt einen Ordner geöffnet hat. Ist ein Order geöffnet, schaut das Programm, ob der Ordnerinhalt ein GIT – Repository ist. Ein einfacher Weg, um dies herauszufinden ist es zu schauen, ob ein .gitignore – File vorhanden ist. Dies macht das Programm auch. Ist dieses File vorhanden, so wird der Status vom GIT – Repository per Shell – Command abgefragt. Ist das Repository nicht «up to date», so wird ebenfalls durch Shell – Command den Ordnerinhalt auf GIT gepusht.



# Schlusswort

Bei der Implementation hatte ich Schwierigkeiten, Teile des Designs exakt gemäss Anforderungen / Design umzusetzen. Beispielsweise konnte ich keine Eingabefenster (Prompt Window) für die Erfassung vom Namen des Commits implementieren, da das Skript erst ausgeführt wird, wenn Visual Studio Code bereits geschlossen ist. Trotzdem konnte ich einige Anforderungen des Skriptes abdecken und ein funktionsfähiges sowie praktisches Skript entwickeln. Für die Zukunft nehme ich mir vor, die Umgebung genauer zu studieren, bevor ich mit Entwerfen bzw. implementieren beginne, um frühzeitig ermitteln zu können, ob und wie gut mein Projekt umsetzbar ist. Trotzdem war dieses Projekt für mich sehr lehrreich. Ich hatte noch nie so lange für so wenige Zeilen Programmcode gebraucht, da ich viel Testen, debuggen und allgemein tüfteln musste.