HOCHSCHULE HEILBRONN

Fallstudie Entwicklungswerkzeuge (282161)

Visual Studio Code

Suphi Pembe (207617),

Waldemar Granson (210386)

SEMESTER 4

Vorgelegt bei Yunus Erdemir

Inhaltsverzeichnis

Abkürzungsverzeichnis iv

Abbildungsverzeichnis v

1 Einleitung 1

2 Die Chronologie von VSC und Pläne für die Zukunft 2
- 2.1 Code Editor im Browser . . . 2
- 2.2 Relase von VSC . . . 3
- 2.3 Pläne für die Zukunft . . . 3

3 Über die Beliebtheit von VSC 4

4 VSC Überblick 7

5 Code-Editor und IDE im Vergleich 8
- 5.1 Der Code-Editor . . . 8
- 5.2 IDE . . . 8

6 Features von VSC 9
- 6.1 Editor . . . 9
- 6.2 Status Bar . . . 9
- 6.3 IntelliSense . . . 10
- 6.4 Activity Bar . . . 11
- 6.5 Explorer . . . 11
- 6.6 Debugging-Tool . . . 12
- 6.7 Command Palette . . . 14
- 6.8 Git . . . 15
- 6.9 Extensions . . . 15

7 Vor und Nachteile von VSC 17
- 7.1 Vorteile . . . 17
- 7.2 Nachteile . . . 17

8 Vergleich mit anderen Tools 19
- 8.1 (Adobe) Brackets . . . 19
- 8.2 Notepad++ . . . 19

8.3 PHPStorm 19
8.4 Sublime Text 20
8.5 Atom 20

9 Fazit und Ausblick **21**

Quellenverzeichnis **vi**

Ehrenwörtliche Erklärung **vii**

Abkürzungsverzeichnis

IDE Integrated Development Enviroment / Integriete Entwicklungsumgebung

VS Visual Studio

VSC Visual Studio Code

UI User Interface / Benutzeroberfläche

Abbildungsverzeichnis

3.1 Stack Overflow Developer Survey 2019: All Responses 4
3.2 Stack Overflow Developer Survey 2019: Mobile Developer Responses . . . 5
3.3 Stack Overflow Developer Survey 2022: All Responses 6
3.4 Stack Overflow Developer Survey 2022: Learning to Code Responses . . . 6

6.1 VSC Status Bar 9
6.2 VSC IntelliSense 10
6.3 VSC Activity Bar 11
6.4 VSC Explorer 12
6.5 VSC Debugging-Tool 13
6.6 VSC Command Palette 14
6.7 VSC Git 15
6.8 VSC Extensions 16

1 Einleitung

Damit ein IT-Technisches Programm entstehen kann, bedarf es einem System, eine oder mehrere Anweisungen, damit ein bestimmtes Verhalten dargestellt werden kann. Diese Anweisungen schreiben Entwickler mithilfe der gewählten Programmiersprache bzw. mehrerer Programmiersprachen zu Softwarecode zusammen.Dieser Code enthält die spezifischen Anweisungen und Abläufe des Programmes. Um die Arbeit der Programmierer leichter zu gestalten, wurden bzw. werden sogenannte Integrated Development Environments (IDEs) entwickelt. Diese IDEs sind Programme die nützlichen Funktionen für die Entwickler bereitstellen. Die meisten modernen IDEs sind auf einen bestimmten Einsatzszenario bzw. einer bestimmten Programmiersprache angepasst. Ein Beispiel hierfür wären die IDEs von der Firma JetBrains. Sie sind die Hersteller von IntelliJ IDEA, eine Entwicklungsumgebung für die Programmiersprache Java, oder PyCharm, die eine Entwicklungsumgebung für Python ist. Jedoch gibt es auch Umgebungen für Entwickler, die ein breiteres Anwendungsszenario abdecken. Eines davon ist Visual Studio, das von Microsoft entwickelt wurde. Es ist nicht für eine spezifische Programmiersprache ausgelegt und unterstützt Programmierer bei dem Editieren und Verwalten von Code. Es ist besonders praktisch, um Dateien zu bearbeiten die außerhalb eines Projektes sind, da man nicht auf die gewohnten Features eines IDE verzichten muss. Einiger Features davon sind z.B. die farbliche Hervorhebung des Codes oder das installieren von Plugins. Visual Studio Code läuft auf den gängigen Betriebssystemen Windows, Linux und MacOS. Dieser technische Bericht dient dazu die Geschichte und die Funktion dieser Entwicklungsumgebung zu erläutern.

2 Die Chronologie von VSC und Pläne für die Zukunft

Die Chronologie von VCS beginnt bereits vor dem eigentlichen Release im Jahr 2015. Vor diesem Release hatte eine VSC eine andere Intention, welche sich dann anders als geplant entwicklte zu der Anwendung, welche wir heute nutzen. Diesen Abschnitt mit zu Dokumentieren ist notwendig um zu verstehen, warum VSC auch als Editor im Browser nutzbar ist.

2.1 Code Editor im Browser

Das Projekt, welches später VSC hervorbrachte, wollte sich mit Konzept beschäftigen einen Code Editor für den Browser zu entwickeln. Somit wurde 2011 das Projekt gestartet dieses Konzept zu entwickeln. Dabei war die Qualität, vom aktuellen Stand von VS, anzustreben.[1]
Im Projektteam gab es Entwickler, welche bereits Erfahrungen mit der IDE Eclipse von IBM hatten. Diese Erfahrungen nutzten Sie für VSC um damit auch das Konzept von Erweiterungen zu Implementieren. Dabei war ein Augenmerk darauf gelegt, dass die Erweiterungen nicht den Hauptprozess im Programm beeinträchtigen.[2]
Vor dem Release von VSC im Jahr 2015 gab es im Jahr 2013 zusätzlich mit dem Launch von VS 2013 auch Visual Studio Online angekündigt. Mit dem Namen Visual Studio Online "Monacoßoll eine Cloud Umgebung speziell für Azure Entwicklung bereitgestellt werden. Damit war das Entwickeln direkt im Browser ermöglicht worden.[3] Azure ist eine Cloud-Computing-Plattform von Microsoft.
Jedoch fand Visual Studio Online "Monaco"nur wenig Anwender im Bereich zwischen 3000 und 5000 Anwendern.[4] Darauf folgend wurde die Entwicklung eingestellt und man fokussierte sich auf die Entwicklung des Code-Editors als Desktop-Anwendung.

[1] Vgl. 2020, *The History of Visual Studio Code* | Timestamp 3:25
[2] Vgl. ebd. | Timestamp 5:00
[3] Vgl. Somasegar, *Visual Studio 2013 Launch: Announcing Visual Studio Online*.
[4] Vgl. 2020, *The History of Visual Studio Code* | Timestamp 14:00

2.2 Relase von VSC

Im Jahr 2015 folgte dann der Beta-Release von VSC. Das Feature, Erweiterungen nutzen zu können, war bereits implementiert. Jedoch befürchtete man das von Drittanbieter keine Erweiterungen geschrieben werden. Da im Vergleich der Text Editor Atom, ein direkter Konkurrent bereits mit bereits etwa 4000 verfügbaren Erweiterungen.[5] Über die Jahre ist VSC in der Beliebtheit explodiert, so dass 2018 VSC erstmalig zur beliebtesten IDE wurde als Code-Editor. Mittlerweile ist VSC die beliebteste ÏDE"mit einem Vorsprung von etwa 40% im Vergleich zum zweiten Platz VS.[6]

2.3 Pläne für die Zukunft

Die Entwicklung von VSC will sich nicht auf Ihren aktuellen Standpunkt beruhen. Es soll der Data Science Community, welche VSC oft nutzt entgegen kommen werden und Notebooks implementiert werden. Diese Notebooks sollen dann nicht als Standalone Feature existieren sondern auch Erweiterungen unterstützen können. [7]

[5] Vgl. 2020, *The History of Visual Studio Code* | Timestamp 12:10
[6] Vgl. Stack, *Stack Overflow Developer Survey 2022*.
[7] Vgl. 2020, *The History of Visual Studio Code* | Timestamp 24:45

3 Über die Beliebtheit von VSC

Wie zuvor erläutert ist VSC die beliebteste IDE als Code Editor. Jedoch ist zu hinterfragen, wie groß der Abstand zu den anderen IDEs ist, wie auch welche Fakten und Schlussfolgerungen daraus zu entnehmen sind.
Im Stack Overflow Developer Survey von 2019 hat VSC einen größere Beliebtheit von etwa 20% zur SOftware aus dem selben Haus VS. Der Abstand zu den darauf folgenden Plätzen erhöht sich danach nurnoch Marginal.[8]
Der einzige Vergleich in dem VSC nur knapp nicht die Führung hat ist bei der Umfrage ge-

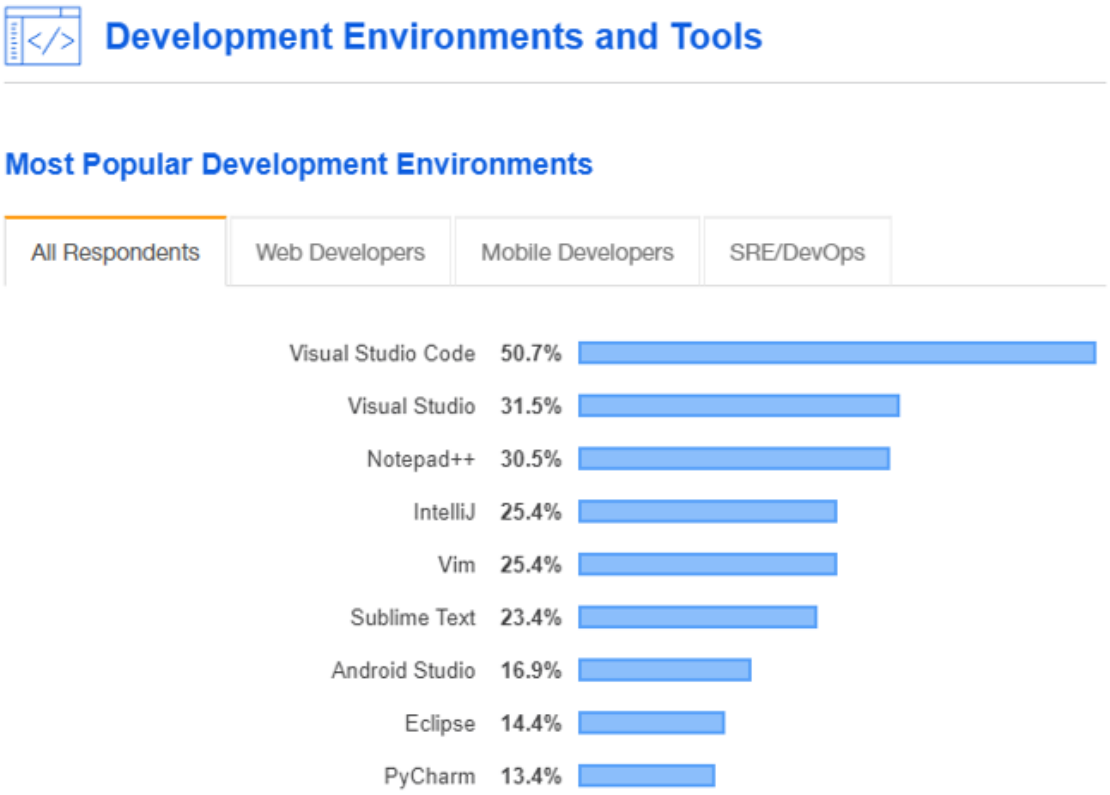

Abbildung 3.1: Stack Overflow Developer Survey 2019: All Responses

filtert für Entwickler von mobilen Anwendungen. Da liegt VSC knapp hinter Android Studio. In Anbetracht, dass Android Studio für den Use-Case der Entwicklung von mobilen Anwendungen ist und VSC ein universelles Tool ist, ist dies beachtlich. Leider erfolgte keine vergleichbare Umfrage in Zukunft bisher.[9] Bei der Stack Overflow Developer Survey von

[8] Vgl. Stack, *Stack Overflow Developer Survey 2019*.
[9] Vgl. ebd.

Abbildung 3.2: Stack Overflow Developer Survey 2019: Mobile Developer Responses

2022 hat sich die größere Beliebtheit von VSC auf etwa 40% erhöht. Auf dem zweiten Platz ist weiterhin VS und die folgenden Plätze haben weiterhin nur nur marginale Abstände.[10]

Jedoch ist auffällig, das unter lernenden Entwicklern VSC eine größere Beliebt von 50% hat. Daraus lässt sich Schlussfolgern, dass VSC in Zukunft noch weiter an beliebtheit wachsen wird.[11]

[10] Vgl. Stack, *Stack Overflow Developer Survey 2022*.
[11] Vgl. ebd.

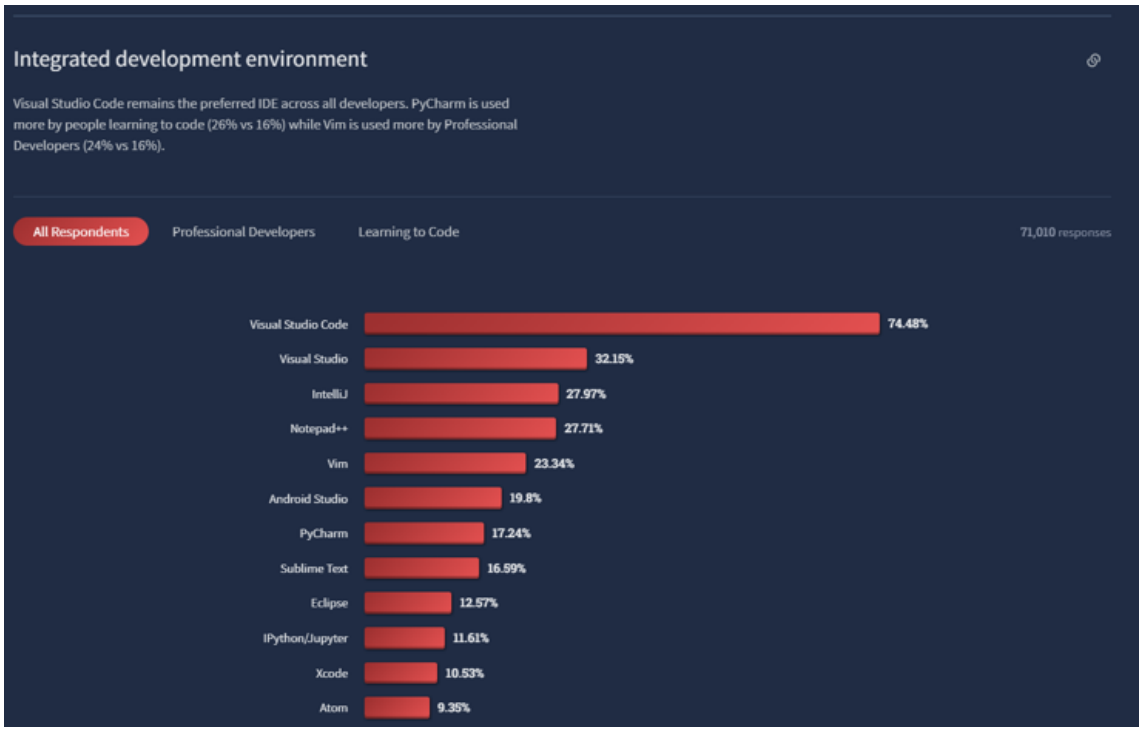

Abbildung 3.3: Stack Overflow Developer Survey 2022: All Responses

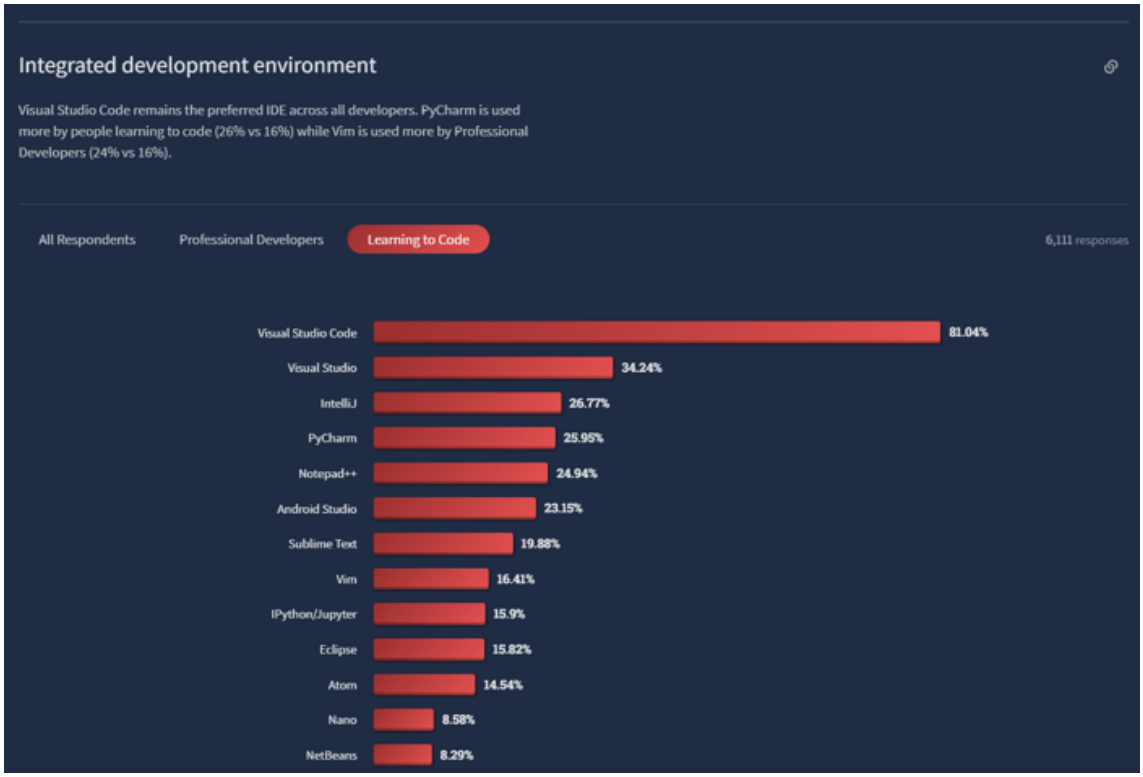

Abbildung 3.4: Stack Overflow Developer Survey 2022: Learning to Code Responses

4 VSC Überblick

Visual Studio Code ist Open Source und von Microsoft entwickelt worden. Es ist ein Code-Editor und keine IDE, dieser Unterschied wird im nächsten Kapitel behandelt. Man kann es auf mehreren Betriebssystemen verwenden, darunter fallen Linux, Windows und MacOS. Obwohl es keine IDE ist, beherbergt es viele nützliche Features die in einer IDE vorzufinden sind. Darunter fallen zum Beispiel, dass Bracket Matching und das Syntax Coloring. Bracket Macthing ist ein Feature, welches die zusammengehörenden Klammern hervorhebt. Dies erleichtert dem Entwickler es zu sehen, was sich in der Klammer alles befindet und mit einem Klick auf eine der zwei Klammerelemente findet man heraus wo sie anfängt bzw. aufhört. Syntax Coloring ist ein Feature, die bestimmte Codewörter farblich hervorhebt, die dem Entwickler orientierungspunkte gibt. VSC kann durch Plugins und Extensions erweitert werden, da einige Funktionen standardmäßig nur für ein paar ausgewählte Programmiersprachen enthalten sind. Weil VSC Open Source ist, können Entwickler ihre eigenen Extensions schreiben, um den Code-Editor nach ihren Wünschen zu erweitern. Aber auch durch die Community gibt es schon viele nützliche Erweiterungen für die gängigsten Programmiersprachen. Anders als bei herkömmlichen IDEs arbeitet VSC nicht mit Projekten, sondern mit einer Struktur von Ordnern und Dateien. Dies bedeutet also, dass man einzelne Dateien öffnen und Editieren kann, aber auch komplette Ordner mit mehreren Dateien können geöffnet werden. Dabei scannt den geöffneten Ordner nach ihm bekannten Dateitypen wie z.B. .json um diese nach einem bestimmten Struktur aufzubereiten. Wenn dem Code-Editor eine Datei unbekannten Typs gegeben wird, so passiert der Prozess ohne eine Strukturierung der Datei. Ein Graphical User Interface-Tool (GUI-Tool) fehlt dieser Anwendung, da es ein reiner Code-Editor ist. Deshalb ist es nicht möglich per Drag and Drop GUI-Elemente zu einem UserIterface hinzuzufügen, dies muss manuell mit der verwendeten Programmiersprache geschehen. Vor allem bei der Entwicklung von Cloud-Anwendungen, der Cross-Plattform und in der Spieleentwicklung mit Unity und Unreal Engine ist VSC oft genutzt. Dies ist besonders der Performance und dem lightweight-Design des Editors geschuldet.

5 Code-Editor und IDE im Vergleich

5.1 Der Code-Editor

Wie erwähnt, ist ein Code-Editor wie VSC zur Bearbeitung von textuellen Dateien vorgesehen. Der geschriebene Code, auch Quelltext genannt, ist hier der Textgegenstand, der mit dem Editor bearbeitet werden soll. Dabei ist ein Code-Editor nicht spezifisch auf eine Programmiersprache ausgelegt, sondern bietet eher grundlegendere Funktionen, wie das genannte Bracket Matching und Syntax Coloring für mehrere Programmiersprachen an. Dies erlaubt es dem Entwickler unabhängig von der Programmiersprache den Quelltext schnell anzupassen. Grundlegende Funktionen sind zwar schon vorhanden und manche Programmiersprachen werden auch schon in der Standardausführung unterstützt, jedoch ist dies bei der großen Auswahl an Programmiersprachen heutzutage oft der Fall, dass viele gewünschte Features erst durch Extensions in den Code-Editor erweitert werden müssen.

5.2 IDE

Hingegen sind IDEs für die Entwicklung in speziellen Use-Cases gefordert. Sie sind für die Entwicklung einer Programmiersprache oder eines Art von Anwendung optimiert. Beispiele dafür sind Android Studio, es für die Entwicklung von Android Anwendung optimiert. Es enthält gegenüber typischen Features noch Beispielsweise eine Android Emulator um den Code auch ausführen zu können. Und auch Pycharm, welches für die Entwicklung mit Python abgestimmt ist. Standardfeatures wie Debugger und Tests sind dann für die Entwicklung optimiert um die Features bestmöglich nutzen zu können.

6 Features von VSC

6.1 Editor

Da VSC ein Code-Editor ist, ist natürlich auch der Editor bzw. die Funktion zum Verändern des Codes das zentrale Feature. Hier ist es einem möglich den Code nach belieben zu verändern und hat wie schon erwähnt ein paar eingebaute Features, wie das Bracket-Matching und auch das Syntax-Coloring. Des Weiteren befindet sich auf der rechten Seite des Editors eine Übersicht der Datei. Dies ist besonders bei großen Dateien hilfreich, da man einen besseren Überblick erhält. Zu dem ist es möglich mehrere Dateien nebeneinander zu öffnen. Dies ist ein praktisches Feature, da man z.B. eine HTML mit der zugehörigen CSS-Datei nebeneinander öffnen und verändern kann. Um diese Funktion zu nutzen ist es möglich die Dateien, die nebeneinander geöffnet werden sollen, mit der Maus per Drag and Drop an die gewünschte Stelle zu schieben. Im oberen Teil des Editors ist außerdem die Navigationsleiste, die einem den Pfad zur aktuellen Datei wie auch die aktuelle Position des Cursors zeigt. Wenn man auf die drei Punkte der Datei klickt, dann ist es möglich die Funktionen und Variablen, die in der Datei vorhanden sind, anzuzeigen.[12]

6.2 Status Bar

Am unteren Bildschirmrand ist die Statusbar. Hier sind Informationen über das Projekt und derzeitig ausgewählte Datei zu sehen. Um genauer zu sein, spricht Microsoft hier von dem „Workspace“. Da es möglich ist mehrere Projekte und einzelne Dateien zu öffnen, deshalb spricht man von einem, auf Deutsch übersetzt, „Arbeitsplatz“. Weitere Informationen, wie der ausgewählte Branch, die Anzahl von Warnungen und Fehler im Code, den verwendeten Unicode Format (wie z.B. UTF-8), wie auch die verwendete Programmiersprache sind in der Status Bar enthalten. Wie viele Elemente des Code-Editors ist auch die Status Bar mit Extensions erweiterbar. Jedoch wird von vielen Extensions, die die Status Bar betreffen abgeraten, da die meisten Erweiterungen schon neue Sachen zur Status Bar hinzufügen.[13]

Abbildung 6.1: VSC Status Bar

[12] Vgl. Microsoft, *Visual Studio Code | User Interface*.
[13] Vgl. Microsoft, *Status Bar in VSC*.

6.3 IntelliSense

Das Feature mit dem Namen IntelliSense ist eine von Microsoft entwickelte Eingabehilfe, die in Office-Programmen, wie auch in VSC, Vorschläge anzeigt und bei Bedarf können diese auch generiert und in die Datei eingefügt. Sie zeigt nicht nur Vorschläge, sondern auch die dazugehörigen Parameter. Es ähnelt deshalb einer „Code-Completion“, wie es auch bei anderen IDEs und Code-Editoren der Fall ist. Standardmäßig ist IntelliSense für JavaScript, TypeScript, JSON, HTML, CSS und SCSS Verfügbar. Generell ist Unterstützt IntelliSense jede Programmiersprache, aber um einen höheren Funktionsumfang bzw. das volle Potential mit anderen Sprachen zu nutzten ist es von Vorteil eine „language extension“, also eine Programmiersprachpaket aus dem Marketplace zu VSC hinzuzufügen. Bei den meisten Sprachpaketen aus dem Marketplace, wie z.B. dem Python Erweiterungspaket, sind dann erweiterte Funktionen für das IntelliSense automatisch dabei.[14]

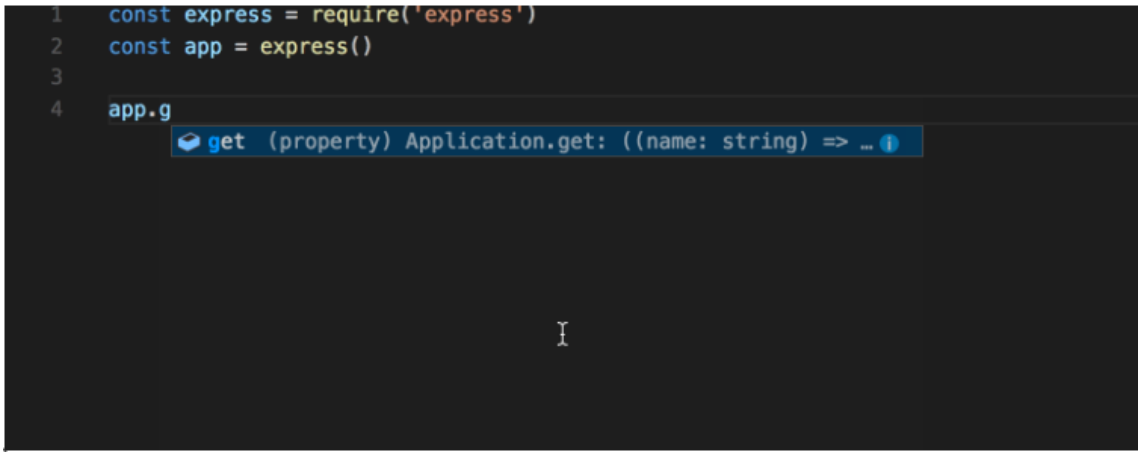

Abbildung 6.2: VSC IntelliSense

[14] Vgl. Microsoft, *IntelliSense in VSC*.

6.4 Activity Bar

Das Navigationselement von VSC wird Activity Bar genannt und befindet sich auf der linken Seite des Programms. Hier sind Widgets vorhanden, die es einem Erlauben schnell auf den Windows-Explorer, auf Git, das Debuggin-Tool oder zu den Extensions zu gelangen. Neue Icons bzw. „Views“, wie Microsoft sie nennt, können der Activity Bar hinzugefügt werden. Dies kann sehr hilfreich sein, da z.B. Views wie ein Notepad oder zu einem bestimmten Ordner erstellt werden können.

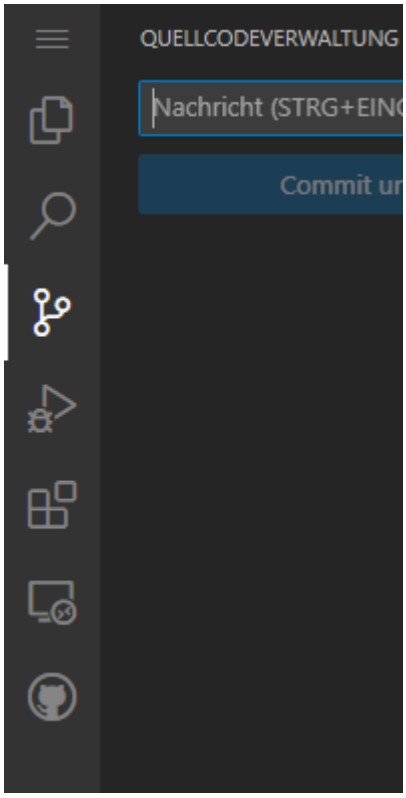

Abbildung 6.3: VSC Activity Bar

6.5 Explorer

Die Navigation und Verwaltung der Dateien erfolgt über den Explorer. Der Explorer befindet sich links neben dem Editor und zeigt die Dateien an, auf die man Zugriff hat. Der Explorer enthält also die Ordner und den darin befindenden Dateien, die ganz einfach geöffnet, verschoben, gelöscht oder umbenannt werden können. Des Weiteren ist VSC super mit anderen Tools, wie dem Befehlszeilenprogramm CMD zu nutzten. Es ist nämlich möglich aus dem Explorer auf einen Order per rechtsklick dessen Pfad direkt im Befehlszeilenprogramm zu öffnen.[15]

[15] Vgl. Microsoft, *Visual Studio Code | User Interface*.

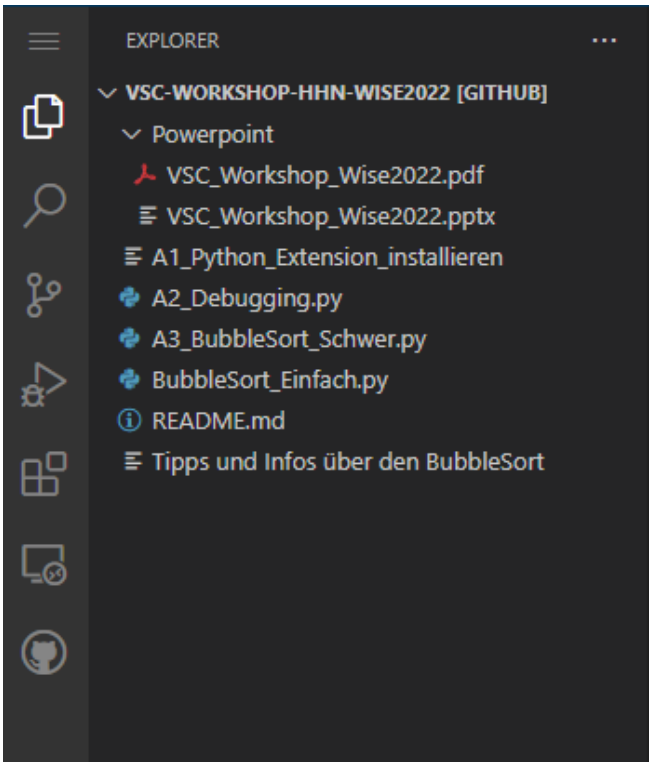

Abbildung 6.4: VSC Explorer

6.6 Debugging-Tool

Obwohl VSC nur ein Code-Editor ist, verfügt das Programm über ein Debugging-Tool. Der Debugger der standardmäßig dabei ist, hilft es Programmierern ihren Code zu analysieren und somit ihre Kompilierungs- und Bearbeitungsschleifen zu beschleunigen. Der Debugger läuft mit der Node.js-Laufzeit, deshalb werden automatisch alle JavaScript-Transpilierten-Sprachen vom Debugger unterstützt. Damit der Debugger auch andere Programmiersprachen unterstützt ist es notwendig die entsprechende Erweiterung aus dem Marketplace zu installieren. Meist ist eine Debugger-Unterstützung für die gewählte Programmiersprache in der Erweiterung der jeweiligen Programmiersprache dabei.

Der Debugger kann über die Activity Bar, der sich auf der linken Seite befindet, oder über die Hotkey-Kombination STRG+SHIFT+D gestartet werden. Der Debugger kann dann über die Schaltfläche Run and Debug gestartet werden, aber auch die vorzeitige Konfiguration eines Launchfiles für den Debugger ist möglich und in den meisten Szenarien auch vorteilhaft. Wenn der Debugger gestartet ist, dann erscheinen Links die Flächen für die Überwachung der Variablen, die Watchschaltfläche die es einem ermöglicht Variablen unter die Lupe zu nehmen und der Call Stack. Rechts davon ist der Code, bei dem es möglich ist, Breakpoints einzusetzen, wenn links neben der Zeilenzahl drückt. Die Breakpoints haben einen eigenen Abschnitt unter dem Call Stack, in dem alle Breakpoints einzusehen sind. Wenn

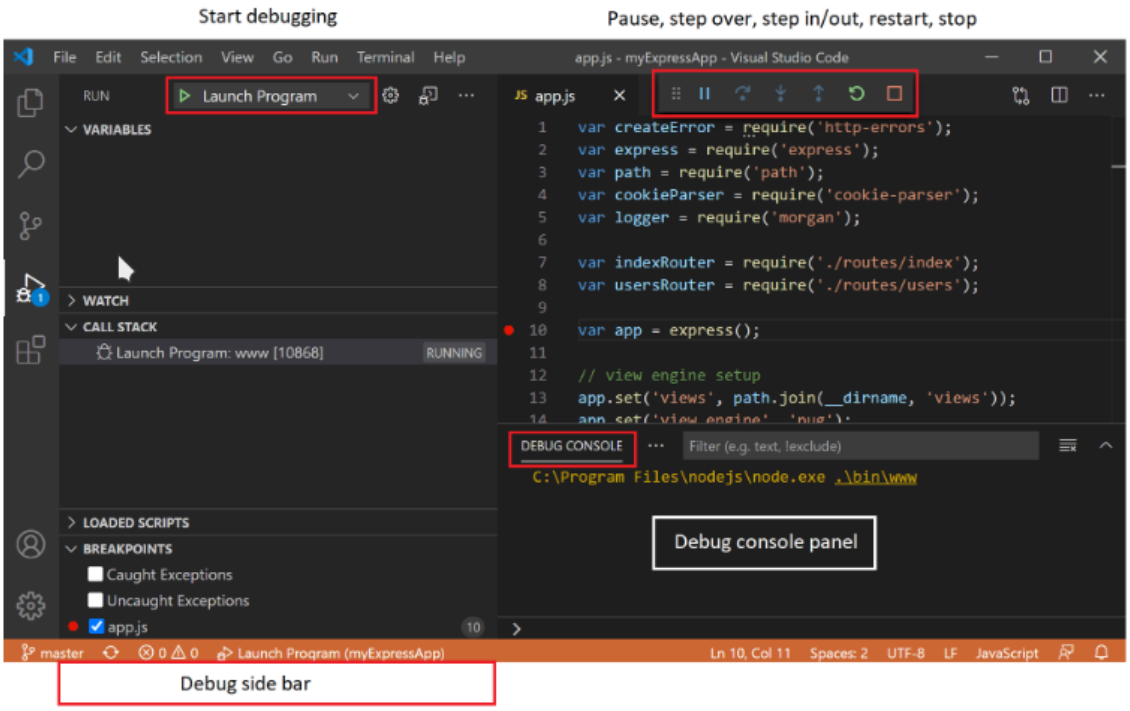

Abbildung 6.5: VSC Debugging-Tool

die Debugumgebung nicht ordnungsgemäß funktioniert und die Breakpoints neu gesetzt werden müssen, dann ist die Funktion „reapply all Breakpoints“ sehr zu empfehlen, da sie alle Breakpoints wieder in die Umgebung setzt. Ist ein Breakpoint gesetzt, dann erscheint über dem Code eine Schaltfläche, die folgende Funktionen enthält.

- Der „Continue“-Button, er lässt den Code bis zum nächsten Breakpoint oder Ende laufen.
- Der „Step Over“-Button, lässt eine Zeile im Code aus.
- Der „Step Into“-Button, macht es möglich in eine lokale Abstraktionsebene zu gehen.
- Der „Step Out“-Button, lässt es zu aus der lokalen Ebene wieder herauszukommen.
- Der „Retry“-Button, lässt den Code erneut vom Anfang laufen.
- Der „Stop“-Button, lässt den Debugger stoppen.

Des Weiteren wird unter dem Code das interne Terminal angezeigt. Es ist durch die Launch-files möglich den Debugger auch die Daten auf ein externes Terminal zu schreiben. VSC bietet außerdem eine Logfunktion, die beim Durchlaufen eines gewählten Punktes eine Nachricht in die Konsole schreibt. Diese Funktion nennt sich Logpoints und ähnelt den bereits Erwähnten Breakpoints. Jedoch unterbrechen sie die Laufzeit nicht, wenn der Code diesen Punkt erreicht. Vor allem bei Produktivsystemen ist dies eine nützliche Funktion, da

der Code nicht unterbrochen wird und es einem ermöglicht Printbefehle im Code zu vermeiden.

Unter dem Terminal befindet sich die Status Bar, die beim Debuggen eine orangene Farbe annimmt und den Debugging-Satus anzeigt. Wenn man auf den Debug-Status klickt, ist es möglich die Startkonfiguration zu ändern und somit das Debugging mit einer anderen Konfiguration starten, ohne die initiale Ausführungsansicht nochmals öffnen zu müssen.[16]

6.7 Command Palette

VSC besitzt von Haus aus viele Befehle, die mit Tastenkombinationen getätigt werden können. Deshalb gibt es eine Suchfunktion die mit STRG+SHIFT+P geöffnet werden kann, um nach bestimmten Befehlen zu suchen. Diese Keybinds, wie Microsoft sie nennt, können

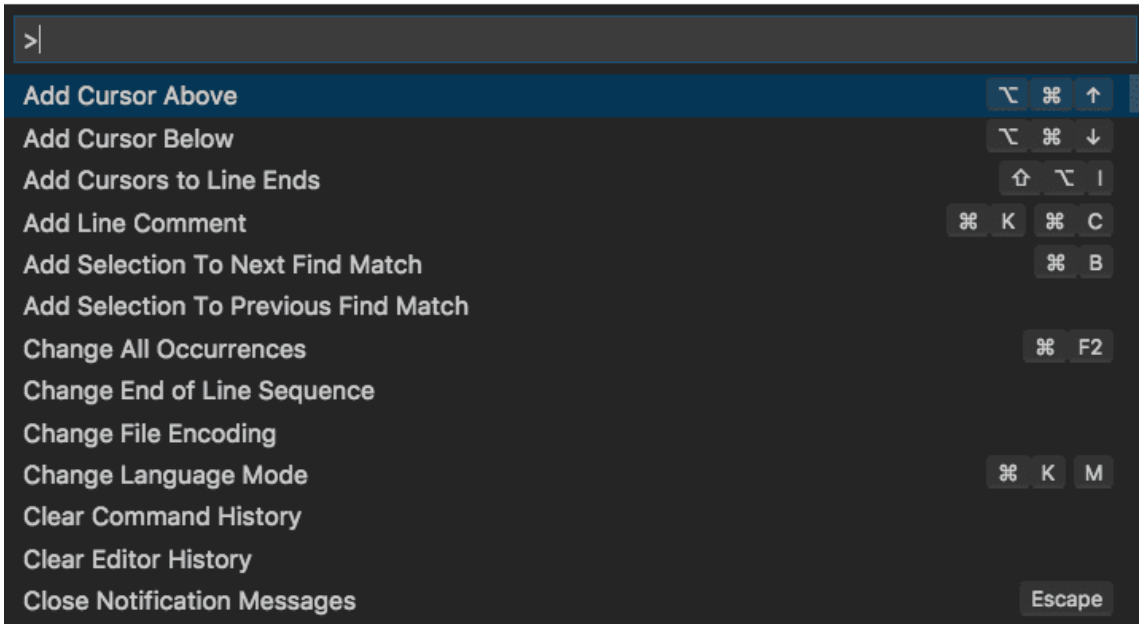

Abbildung 6.6: VSC Command Palette

auch editiert werden. Dafür gibt es ein eigenes Menü, das mit STRG+K+S oder über die Optionsschlaftfläche, geöffnet werden kann. In diesem Falle öffnet sich das Keybind-Menü in einem neuen Tab anstatt, der nur auf der Oberfläche befindlichen Suchfunktion. Bei den Keybinds sind kaum gerenzen gesetzt, sie erlauben es einem schnell neue Zeilen einzufügen, Mehrfachauswahlen zu tätigen, Git-Befehle ausführen, sonstige Funktionen von Extensions zu benutzten.[17]

[16] Vgl. Microsoft, *Debugging in VSC*.
[17] Vgl. Microsoft, *Visual Studio Code | User Interface*.

6.8 Git

Git ist ein wichtiger Teil beim Coden, deshalb ist Git standardmäßig in VSC enthalten. Durch den Workspace-Ansatz von VSC ist es möglich mehrere Repositorys neben einem lokalen Arbeitsbereich geöffnet zu haben. Nach dem Verändern des Code oder bei dem hinzufügen von neuen Dateien ist es möglich die Änderungen zu Commiten. Um eine Commit-Nachricht, die mit dem Commit geschehen soll zu senden, bracht es lediglich den Eintrag in das Textfeld über der Ansicht der veränderten Dateien.[18]

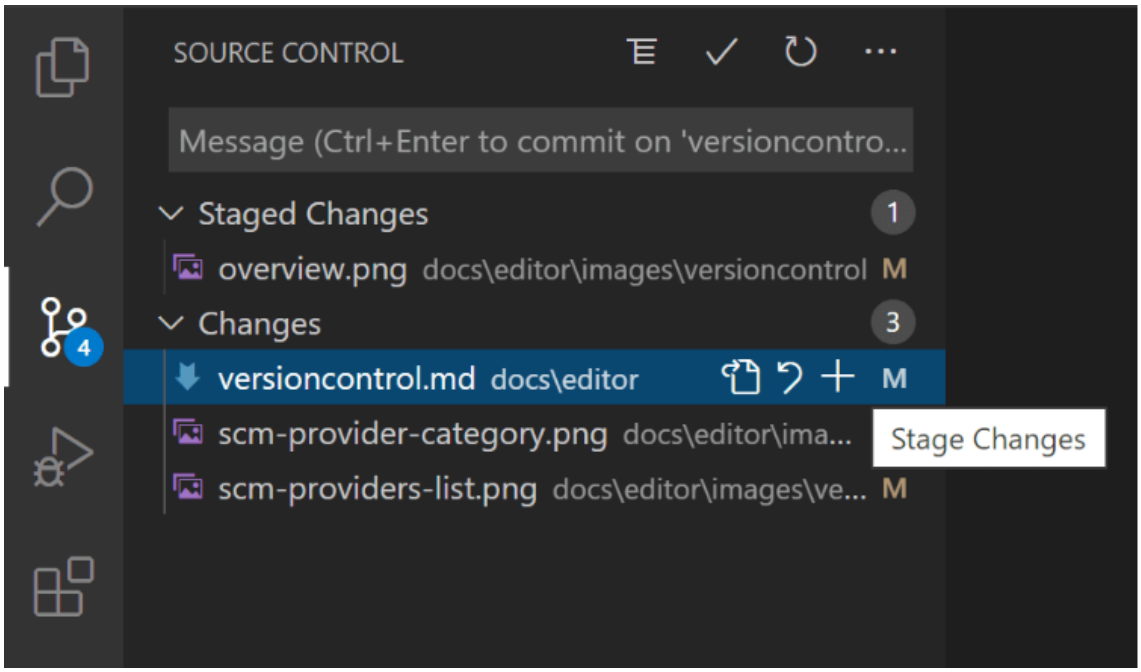

Abbildung 6.7: VSC Git

6.9 Extensions

Erweiterungen sind das A und O dieses Code-Editors. Der Lightweight-Ansatz dieses Editors bedeutet, dass die Anwendung nur das Nötigste bzw. nur das standardmäßig enthält, was den Umgang mit diesem Programm erleichtert. Dadurch ist es häufig der Fall, dass man eine Extension herunterladen muss, da etwas fehlt, dass man gerne haben würde. Zum Glück gibt es hierzu eine sehr große Auswahl, denn Extensions wie z.B. eine andere Programmiersprache ist in dem meisten Fällen vorhanden. Selbst wenn eine Extension fehlen sollte, ist es möglich sich eine eigene Erweiterung zu bauen. Dies ist möglich, da das Programm Open-Source ist und über die API können die Schnittstellen in dem Programm

[18] Vgl. Microsoft, *Source Control with Git in VSC*.

angesteuert werden. Das Extensions Menü ist über das Icon auf der Side Bar links aufzurufen, oder über den Keybind CTRL+SHIFT+X. Hier ist der Marketplace zu finden, der es einem erlaubt nach Extensions zu suchen, um diese Herunterzuladen und auf die Anwendung zu installieren. Auch das Verwalten, sprich die Einstellungen der Erweiterungen, wie auch das Löschen, ist von hier aus möglich.[19]

Abbildung 6.8: VSC Extensions

[19] Vgl. Microsoft, *Managing Extensions in VSC*.

7 Vor und Nachteile von VSC

7.1 Vorteile

Der Code Editor VSC ist ein sehr beliebtes und weit verbreitetes Tool zum Editieren von Code. Dies ist zum einen wegen seines lightweight-Ansatzes, da er nicht mit unnötigen Sachen überladen ist und dadurch auch Ressourcen sparen kann, zum anderen das VSC es einem durch den Workspace-Ansatz ermöglicht an mehreren Projekten oder Dateien aus verschiedener Herkunft zu arbeiten. Deshalb ist er vor allem bei vielen Programmieranfängern beliebt, wie die Umfragen von Stakeoverflow aus Kapitel 3 zeigen. Eine große Anzahl an Usern hilft dem Code-Editor sehr, da es auch die Wahrscheinlichkeit erhöht, dass Extensions für die Anwendung geschrieben werden. Das ist ein fundamentaler Baustein des Code-Editors, da diese Anwendung von Erweiterungen lebt. Durch Extensions kann der User sich sein eigenes Tool zusammenbauen, um damit produktiver zu sein. Selbst wenn Extensions fehlen ist der Benutzer in der Lage seine eigenen Erweiterungen für VSC zu schreiben und diese zu Veröffentlichen. Die standardmäßigen Features wie das Debuggin-Tool, IntelliSense und eingebautes Git sind sehr mächtig und helfen dem Programmierer von Anfang an beim editieren und erstellen von Code. Ein weiterer Punkt hierbei ist, dass VSC durch das verwendete Electron Framework auch in den gängigen Internetbrowsern funktionert. Durch das Aufrufen der Webseite https://vscode.dev/ in Edge, Firefox, Chrome oder Safari hat man fast alle Funktionen von VSC die auch die heruntergeladene Version hat als Weboberfläche. Dadurch kann VSC auf mehreren Betreibssystemen wie z.B. MacOS, Windows und Linux benutzt werden.

7.2 Nachteile

Trotz alledem gibt es auch Nachteile beim benutzten von VSC. Die Anwednung lebt in vielen Fällen von Extensions. Diese sind jedoch ein Segen und ein Fluch zugleich, da Extensions die User-Experience in vielen Fällen verbessern können sind sie meist Vorteilhaft für den Endnutzter, jedoch können schlecht geschriebene Extensions oder eine falsche Benutzung von Extensions zum massiven Overhead oder sogar zum Abstürzten des Programmes bzw. im schlimmsten Falle auch zum Absturtz des ganzen Rechners führen kann. Des Weiteren ist auch die Einarbeitungszeit nicht zu vernachlässigen. Programmierer leben davon Code so effizient wie möglich zu schreiben. Dies geschieht meist durch Keybinds, die es einem erleichtern können den Code zu editieren. VSC besitzt eine unglaubliche Anzahl an Key-

binds, sodass es seine Zeit bedarf um die Keybinds zu lernen. Das Problem erweitert sich dadurch, dass die meisten Extensions weitere Keybinds hinzufügen, deshalb kann es eine lange Zeit dauern bis man sich in VSC eingelebt hat, um damit effizient Arbeiten zu können. Ein weiterer Nachteil ist das Fehlen von GUI-Tools. Viele IDEs besitzten GUI-Tools, wie z.B. das Design-Tool von Android Studio, dass einem erlaubt XML-Designs per Drag and Drop zu erstellen, damit Programmierer beim erstellen von Anwendungen mit grafische Oberflächen unterstützt werden. Dies ist in VSC leider nicht der Fall, da dieses Feature fehlt müssen Programmierer beim Benutzten von VSC auf die jeweilige Designsprache ausweichen.

8 Vergleich mit anderen Tools

In diesem Kapitel sollen verschiedene Code Editoren und vergleichbare IDEs zu VSC verglichen werden. Es wird aufgezeigt was für und entgegen dem verglichenen Tool spricht und ob es in spezifischen Use-cases angebrachter ist dies zu verwenden.

8.1 (Adobe) Brackets

Brackets ist ein Open Source Code Editor und ist auf allen gängien Betriebssystem verfügbar. Ursprünglich von Adobe Inc. 2014 veröffentlicht und als Open Source Projekt weitergeführt. Der Schwerpunkt von Brackets in der Web Entwicklung. Es verfügt über ähnliche Features wie Git-Integration, einen Debugger, Erweiterungen bzw. Plugins und weitere gemeinsame Features. Ist jedoch weniger Performant.
Insgesamt ist Brackets sehr ähnlich zu VSC, jedoch weniger Performant. Oft wird sich jedoch VSC anstatt Brackets entschiedne, da Entwickler sich auf die Erfahrung von den Microsoft Entwicklern vertrauen anstatt den Open Source Entwicklern von Brackets

8.2 Notepad++

Notepad+++ ist ein Open Source Code Editor und ist auf nur auf Windows Betriebssystem verfügbar. Notepadd++ wurde erstmal 2003 veröffentlicht und sollte als Alternative zum im Windows integrierten Notepad dienen. Mit Standardfeatures wie Source Code Highlighting und Code Formatting, kann aber mit Erweiterungen mit einem Debugger ausgestattet werden. Notepad++ ist sehr Minimalistisch gehalten.
Der größte Vorteil von Notepad++ ist, dass durch den Minimalismus des Code Editors er extrem Performant ist, jedoch sehr Feature arm. Wenn man nur notwendige und selektierte Plugins installiert, kann man dadurch sehr viel Performance erhalten ohne dabei Funktionalität einzubüßen. Jedoch ist dafür Konfiguration notwendig, welche Zeitaufwendig ist. Hier in Anbetracht dessen entschieden ob der Zeitaufwand den Nutzen rechtfertigt.

8.3 PHPStorm

PHPStorm ist eine IDE für Web-Entwicklung und ist auf allen gängigen Betriebssystem verfügbar. Der Use-Case für die Verwendung von PHPStorm ist die Entwicklung von dynamischen Webanwendungen. Es verfügt über eine Git-Integration und nutzt den XDebug

Debugger, welche extra für PHP-Anwendung entwickelt wurde. Jedoch ist PHPStorm auf die Entwicklung von PHP Anwendungen beschränkt.
Der Preis von 249€ im ersten Jahr, absteigend bis 149€ im dritten Jahr ist ein großer Kostenaufwand. Sollte der Großteil des Projekts, oder der Geschäftsplan, mit der Entwicklung von PHP Anwendungen geplant werden, kann die Entwicklung mit PHP Storm eine große Zeit Ersparnis sein. Besonders durch den XDebug Debugger. Jedoch für kleinere bis mittlere Projekte ist VSC angemessener.

8.4 Sublime Text

Sublime Text ein Code Editor. Ähnlich zu VSC ist er Lightweight und performant, überbietet jedoch VSC in diesen beiden Themen. Beide verfügen ebenfalls über eine Git-Integration, einen Debugger, Erweiterungen bzw. Plugins und weitere gemeinsame Features. Im Gegensatz zu VSC ist Sublime Text jedoch kein Open Source Projekt und er ist kostenpflichtig. Der Preis von 100€ kann in einem sehr großen Projekt Zeit sparen, da Sublime ein schnellerer Code Editor ist. Jedoch muss man abwägen ob die Zeitersparnis im Verhältnis zu den kosten sind.

8.5 Atom

Atom ist ein Open Source Code Editor und ist auf allen gängigen Betriebssystemen verfügbar. Dieser wurde 2014 released von dem selben Entwickler Team wie das von GitHub, und wird dementsprechend auch mit GitHub assoziert, da es eine direkte GitHub anbindung bietet. Eine große Gemeinsamkeit, welche Atom mit VSC hat, ist die Tatsache das beide Code Editoren mit dem Electron Framework arbeiten.Da jedoch Atoms Nutzerzahlen immer weiter sinken, wird der Support für Atom am 15.12.2022 eingestellt. Da es sich in Zukunft um Legacy Software handelt, ist Atom keine Alternative zu VSC.

9 Fazit und Ausblick

VSC ist ein mächtiger Code-Editor der von Microsoft auf dem Electron Framework entwickelt wurde. Durch den leichtgewichtigen Ansatz und dem vielfältigen Anwendungszenario ist er zum beliebtesten Editor geworden. Er ist auf den gängigsten Betriebssystemen einsetzbar und läuft sogar im Browser mit nur sehr wenigen Einschränkungen im Gegensatz zur Desktop-Applikation. Zu dem ist VSC kostenfrei, OpenSource und durch Extension prima erweiterbar. Die Community des Editors ist riesig und veröffentlicht neue Extensions die fast jeden Usecase abdecken. Andere Code-Editoren auf dem Markt müssen in Zukunft neue bahnbrechende Features herausbringen, wenn sie VSC als beliebtesten Editor vom Trohn stoßen möchten.

Quellenverzeichnis

2020, Microsoft Build. *The History of Visual Studio Code*. 2020. (Besucht am 16. 11. 2022).

Microsoft. *Debugging in VSC*. (Besucht am 11. 12. 2022).

– *IntelliSense in VSC*. (Besucht am 13. 12. 2022).

– *Managing Extensions in VSC*. (Besucht am 13. 12. 2022).

– *Source Control with Git in VSC*. (Besucht am 14. 12. 2022).

– *Status Bar in VSC*. (Besucht am 10. 12. 2022).

– *Visual Studio Code | User Interface*. (Besucht am 14. 12. 2022).

Somasegar, Sivaramakrishnan. *Visual Studio 2013 Launch: Announcing Visual Studio Online*. 2013. (Besucht am 16. 11. 2022).

Stack, Overflow. *Stack Overflow Developer Survey 2019*. 2019. (Besucht am 18. 11. 2022).

– *Stack Overflow Developer Survey 2022*. 2022. (Besucht am 18. 11. 2022).

Ehrenwörtliche Erklärung

„Wir versichern, dass die vorliegende Arbeit von uns selbständig und ausschließlich unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt wurde. Alle Stellen, die wörtlich oder annähernd aus Veröffentlichungen entnommen sind, haben wir als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form, auch nicht in Teilen, keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.“

KAPITEL 2, 5.2 und 8 wurden von Suphi Pembe verfasst.
KAPITEL 1, 4, 5.1, 6, 7 und 9 wurden von Waldemar Granson verfasst.

Rot am See 15.12.2022
Ort, Datum


Unterschrift

Ort, Datum

Unterschrift