# Diverses

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | ZEN Mode |  | Strg + K Z | | Line Copy |  | Strg + Shift + Pfeil hoch / Pfeil runter | | Move Line |  | Alt + Pfeil hoch oder Pfeil runter | | Delete Line |  | Strg + Shift K | | Linke Leiste Ein-/Ausblenden | Explorer | Strg + B | | Untere Leiste Ein-/Ausblenden | Terminal | Strg + J | | Python Script laden ohne auszuführen | Damit habe ich Zugriff auf Methoden usw. | python -i main.py | | Im Dokument suchen |  | Strg + P dann @ | | Im Workspace suchen |  | Strg + P dann # | | Help | Anzeigen der möglichen comands | Strg + P dann ? | |  |  |  |   [Visual Studio Code (Windows) - Setting up a Python Development Environment and Complete Overview - YouTube](https://www.youtube.com/watch?v=-nh9rCzPJ20&list=PLd5CM1FQHzcfLnjzuR5q2gfiBXxsEttKv&index=4&t=1544s) (47:45) |

|  |
| --- |
| Visual Studio Unit Tests Strg + Shift + P -> „Python: Configure Tests“   * Es kommt eine Routine, mit der ich konfigurieren kann. Wenn ich das durch habe sind die entsprechenden Einstellungen in der settings.json sichtbar!!! 🡪 Diese Settings kann ich genauso gut ausschneiden und in die workspace Settings reintun * Jetzt ist eine neue Kachel in der Action Bar auch sichtbar   Strg + Shift + P -> „Python: Discover Tests“   * Jetzt müssten die Tests in der Action Bar Tests Tab zu sehen sein   **„Bug“:** Wichtig war, das ich den Pfad während der Configuration direkt auf den Ordner gesetzt habe wo meine Python Dokumente waren. Sonst hat es rumgeschissen!  **Beispiel:**    Wichtig:   * main.py und test\_main.py liegen im selben Ordner und der Pfadin „Python: Configure Tests“ wurde auf genau diesen Ordner gesetzt; in diesem Fall „Day14“ |

# Visual Studio Code – Get Started

**Settings**

|  |  |
| --- | --- |
| Settings Filter (In Search Bar type in…):   * @modiefied * @ext:ms-python.python * @feature:explorer * @id:workbench.activityBar.visible * @lang:python * @tag:workspaceTrust * …   Hint: if @lang:python is active in Seraching Widget, the settings is configured in the given scope for python. | Show all modified Settings  Specific to an extension through extension ID  Specific to a feature sub group, here File Explorer  Based on setting ID  Based on Language ID  Specific to a system, |

# Visual Studio Code – User Guide

**Basic Editing**

|  |  |
| --- | --- |
| Multi Cursor – Add Cursors | * Alt+Click * Strg+Alt+Up * Strg+Alt+Down |
| Selecting the Word at the Curser | * Strg+D |
| Set multi-cursor on same occurrences | * Strg+Shift+L |
| Box-selecting  (Damit markiere ich genau die Stellen) | * Shift+Alt+Mouse  (Maus ziehen) |
| Auto Save | Settings (User, Worksapce):   * files.autoSave * files.autoSaveDelay   (Einfach diese Texte in die Suche in den Settings kopieren und einfügen) |
| Format Code:  Auto-Format on Saving: | * Shift+Alt+F * Settings:   + editor.formatOnSave   + editor.formatOnType   + editor.formatOnPaste |
| Folding Code Areas – Fold ALL  Folding Code Areas – UnFold ALL  **Folding Code Areas – Fold by Level**  Folding Code Areas – Fold Block Comments | * Strg+K Strg+0 (Null) * Strg+K Strg+J * Strg+K Strg+2 (e.x. Till Level 2) * Strg+K Strg+/ |

# Visual Studio Code – Python

## Tutorial

|  |
| --- |
| Command prompt:   * py -3 --version |
| Select Python Interpreter: (Pfad für Interpreter setzen)   1. ctrl + shift + p 2. „Python: Select Interpreter“ eintippen    1. Wenn man in einem Workspace mit mehreren Ordnern ist kann man dann entweder für ganzen Workspace oder bestimmte Ordner auswählen    2. Wenn man in keinem Workspace oder Ordner ist wird der Pfad für den User bzw. Global (quasi Default gesetzt) 3. Dann den entsprechenden Interpreter auswählen.   In Settings nachsehen: File -> Preferences -> Settings -> TAB: User oder Workspace oder Folder   * In der Suche: „python.pythonPath“ eingeben hier sieht man das der „Python: Python Path“ (python.pythonPath) auf den Pfad des Interpreters gesetzt wurde. |
| Code Ausführen:   * Play Symbol oben rechts, dann wird im Grunde im Terminal „python dateiname.py“ ausgeführt. * Recht Maustaste irgendwo im Editor und „Run Python File in Terminal“ * Codebereich markieren   + Shift + Enter für markierten Code ausführen   + Rechte Maustaste drauf und „“Run Selection/Line in Python Terminal“ * Ctrl+Shift+P und „Python: Start REPL“ Hier kann man dann im Terminal beliebig Python Code eingeben für den aktuellen für das Workspace/Folder ausgewählten Interpreter |
| Debug:   * Wenn ich mein Code im Debug Modus starte kann ich unten neben dem Terminel „Debug Console“ auswählen und hier auch alles mögliche eintippen und abfragen (Anstatt immer Links die Watch Variablen einzugeben) * Während ich im Debug Modus bin kann ich auch über Variablen mit der Maus fahren um die Informationen bezüglich aktuelle Inhalte zu sehen. * Anstatt prints in Code einzubauen Logpoints nutzen |
| Pakete:   * Sind Bibliotheke die man installieren kann * Diese werden typischerweise von PyPi bezogen/installiert (<https://pypi.org/>)   Pakete installieren:   * Ctrl + Shift + P -> „Terminal: Create new integrated Termial: (.myVM) PS C:\Users\guemu\OneDrive\Desktop\Udemy> (.myVM) ist quasi jetzt gewählt und hier installiere ich dann auch das Paket rein * Wenn (.myVM) nicht angezeigt wird dann: .myVM\scripts\activate * Pakete installiert man dann indem man im Terminal z.B. sowas eingibt: python -m pip install matplotlib oder python -m pip install -U matplotlib (was heißt das -u?) * pip selber installieren (glaub für upgrade?): python -m pip install -U pip   Pakete deinstallieren:   * Wenn ich Matplotlib installiere wird die aktuelle Version von Numpy mit installiert. Hier gibt es aber einen Bug sodass ich Downgraden muss   + pip uninstall numpy 🡪 Aktuelle Version deinstallieren   + pip install numpy==1.19.3 🡪 bestimmte Version installieren   Alle installierten Pakete anzeigen:   * pip list   Diese Pakete installieren:   1. pip install autopep8 (Code formatieren) Dann: Strg + Shift + P 🡪 "Format Document" Oder: Strg + Shift + F 2. pip install flake8 (Linter) STRG + Shif + P -> Python: Select Linter 3. .. |

# Workspaces

|  |  |
| --- | --- |
| [Workspaces in Visual Studio Code](https://code.visualstudio.com/docs/editor/workspaces)  [Multi-root Workspaces in Visual Studio Code](https://code.visualstudio.com/docs/editor/multi-root-workspaces) | |
| **Single-Root Workspace (Just a Folder)**  🡪 One Project   * UI Settings: User und Workspace * Locations: in .vscode there is a settings.json   Here my Folder is my Workspace, be not confused because its not named as such one. | **Multi-Root Workspace (Several Folders)**  🡪 Several Projects   * UI Settings: User, Workspace und Folder * Locations: in .vscode there is a settings.json and a .code-workspace   🡪  Here my Wokspace is defined through my .code-workspace Folder. All relevant Projects/Folders are listed here as well as the “integrated” settings.json on this Level. |
| User Settings (Global Settings)   * As Front-End: STRG+P: Preferences: Open User Settings * As Json-File: STRG+P: Preferences: Open Settings (Json) * Location: %APPDATA%\Code\User\settings.json | |
| Single-Root Workspace Settigns   * As Front-End: STRG+P: Preferences: Open Workspace Settings * As Json-File: STRG+P: Preferences: Open Workspace Settings (Json) * Location: <Project\_Folder>\.vscode\settings.json | |
| Multi-Root Workspace Settings   * As Front-End: STRG+P: Preferences: Open Workspace Settings  STRG+P: Preferences: Open Folder Settings * As Json-File: STRG+P: Preferences: Open Workspace Settings (Json)  STRG+P: Preferences: Open Folder Settings (Json) * Location: <Project\_Folder>\.vscode\settings.json  <Project\_Folder>\.vscode\<workspace>.code-workspace   For a Multi-root Workspace, workspace settings (content of settings.json) are located inside the workspace configuration file. | |
| * Default settings -> This scope represents the default unconfigured setting values. * User settings -> Apply globally to all VS Code instances. * Workspace settings -> Apply to the open folder or workspace. * Workspace Folder settings -> Apply to a specific folder of a multi-root workspace. * Language-specific default settings -> These are language-specific default values that can be contributed by extensions. * Language-specific user settings -> Same as User settings, but specific to a language. * Language-specific workspace settings -> Same as Workspace settings, but specific to a language. * Language-specific workspace folder settings -> Same as Workspace Folder settings, but specific to a language. | |
| * Über UI Settings vornehme, dann werden die entsprechenden JSON Einträge erzeugt * JSON Einträge von abstrakter nach spezifischer kopieren und konfigurieren, dann werden die abstrakteren überschrieben. | |

Folder zu Workspace hinzufügen:

* Ctr +shift + P -> „Workspaces: Add Folder to Workspace“
* Merke: Auch wenn ich einen „Überordner“ habe in dem alle Projekte als einzelne Ordner drinnen sind, dann öffne ich das erste Projekt/Ordner und ergänze dann alle anderen Projekt/Ordner mit „Workspaces: Add Folder to Workspace“. Dann „Save Workspace as“ 🡪 Also nicht den Oberordner adden: Das wäre dann kein Multi-Root Workspace mehr, sondern einfach nur ein Workspace, was aber als Folder behandelt wird und nicht als „echtes“ Workspace durchgeht.

Folder aus Workspace rausnehmen:

* Ctr +shift + P -> „Remove Folder from Workspace“

Workspace speichern:

* Ctr +shift + P -> „Workspaces: save Workspace as“
* Wenn ich das mache, dann kann ich eine .code-workspace Datei speichern

Workspace schließen:

* Ctr +shift + P -> „Workspaces: close Workspace“

Workspace öffnen:

* Ctr +shift + P -> „Workspaces: open Workspace“
* Hier kann ich dann eine .code-workspace Datei auswählen und öffnen

Workspace Configuration File:

* Ctrl + Shif + P -> „Workspaces: open workspace configuration file“
* Das selbe wie „Preferences: open workspace settings (json)“

Workspace löschen:

* Im File Explorer .code-workspace Datei finden und den löschen

Weiteres:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Der . in Folders aus dem Screenshot bedeutet: der selbe Ordner wo auch die config Datei (.code-workspace) liegt.. Alle weiteren Folder würden mit Pfad drin stehen 🡪 update: mit dem . meint man auch alle Unterordner, also würden nur alle weiteren Folder mit Pfad hier stehen die nicht in dem Root Ordner sind in dem auch die .code-workspace liegt.

# Virtual Environment

1. Einen Workspace erstellen und die Projektordner assignen / Oder Single-Root Workspace nuten! 🡪 Dann zum root Ordner navigieren.
2. Ctrl + Shift + P -> „Terminal: Create new integrated Termial (In Active Workspace)  
   Bzw. In den Projektordner/Workspaceordner navigieren
3. Über Terminal: Im Terminal Virtuelle Umgebung erstellen: py -3 -m venv .venv
   1. -3 = Python Version
   2. -m = ?
   3. venv = ich möchte eine virtuelle Umgebung erstellen
   4. .venv = Frei wählen -> Der Name meiner virtuellen Umgebung

Über VS Code: STRG + Shift + P -> „Python: Create Environment“

1. Die VM als Standard Interpreter für **Workspace** (workspace.code-workspace):
   1. Ctrl. + Shift + P
   2. „Select: Python Interpreter“
   3. Entire Workspace
   4. Den Interpreter mit (venv) auswählen

VM als Standard für den Workspace(workspace.code-workspace bezieht sich auf Workspace)  
"python.pythonPath": ".deepdive\\Scripts\\python.exe"

1. Die VM als Standard Interpreter für **Ordner** (.vscode\settings.json):
   1. Ctrl. + Shift + P
   2. „Select: Python Interpreter“
   3. Für den Ordner nur
   4. Den Interpreter mit (venv) auswählen

VM als Standard für einen Folder im Workspace (settings.json bezieht sich auf Folder oder Singel-Root Workspace)  
"python.pythonPath": "c:\\Users\\guemu\\OneDrive\\Desktop\\Udemy\\.myVM\\Scripts\\python.exe"

1. Im Terminal Virtuelle Umgebung aktivieren: .myVM\scripts\activate  
   Wenn ich mein Code ausführe muss ich dann immer das sehen -> (.myVM):Ein Bild, das Text, Bildschirm, Screenshot enthält.

   Automatisch generierte Beschreibung  
   Oder: STRG + Shift + P -> „Terminal: Create new Terminal (in active workspace)“