Git in Visual Studio Code verwenden: pull, stage, commit, push

Diese Zusammenfassung beschäftigt sich mit der Verwendung von Git in Visual Studio Code. Dabei werden im Folgenden, die vier wichtigen Befehle *pull*, *stage*, *commit* und *push* näher erläutert.

Pull:

Nachdem das lokale Visual Studio Code-Programm über den Befehl *Git: Clone* mit der persönlichen Github-URL verbunden und die zugehörige Datei (bestenfalls auf dem Desktop) gespeichert wurde, können vorherige Arbeitsschritte aus der Github-Datei abgerufen und in Visual Studio Code hineingezogen werden. Dafür muss der Benutzer in der linken Zeile auf das Feld *Source Control* (Ctrl+Shift+G), danach auf die drei Punkte (*More Actions…*) gehen und dann den *Pull*-Befehl ausführen (vgl. Abb. 1, grüner Kreis). Hierbei werden nun, wie oben schon erwähnt, bereits absolvierte und gespeicherte Schritte aus dem Github in das lokale Visual Studio Code-Programm gezogen.

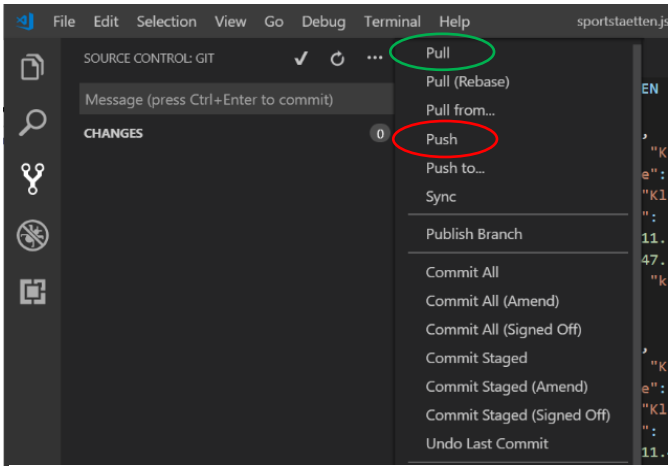

Abbildung 1

Wenn vorher an einem anderen Computer gearbeitet wurde, empfiehlt sich dieser Arbeitsschritt bei jedem Start des Visual Studio Code Programms, um den aktuellen Arbeitsstand zu erhalten und daran weiterarbeiten zu können.

Stage:

Wenn der *Pull*-Befehl erfolgreich ausgeführt wurde, kann die Arbeit beginnen. Neue Texte können in die html-Datei eingefügt oder Skripte umgeschrieben werden. Um diese Änderungen anzuwenden, müssen diese gespeichert und schlussendlich zum Github gepusht werden. Der erste Schritt ist hierbei die Daten in Visual Studio Code zu speichern, dafür muss der Benutzer in der oberen Spalte auf *File* und dort auf *Save File* gehen, oder die Tastenkombination Ctrl+S nutzen. Anschließend zeigt das Feld der *Source Control* an, dass Änderungen gespeichert wurden, diese müssen nun mit dem Befehl *Stage* gesichert werden. Hierfür muss der Benutzer in der linken Zeile auf das Feld *Source Control* (Ctrl+Shift+G) gehen und mit dem „+“ alle Änderungen stagen (vgl. Abb. 2, blauer Kreis), an die Stelle des „+“ tritt nun ein „-“. Mit einem Klick auf das „-“ kann der *stage*-Befehl, wenn gewünscht, wieder zurückgenommen werden.

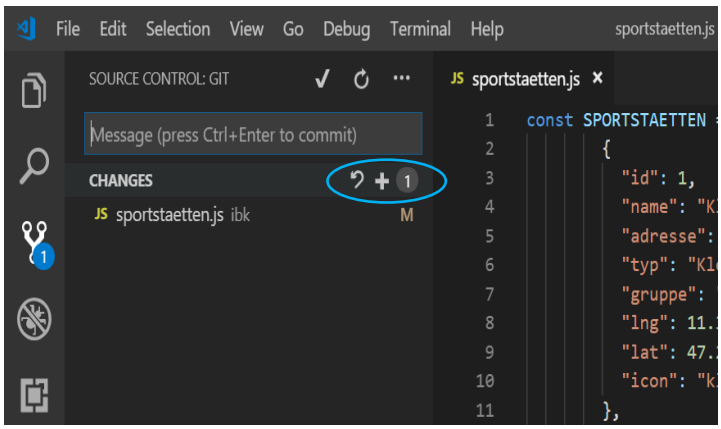

Abbildung 2

Commit:

Auf den *stage*-Befehl folgt nun der *commit*-Befehl, um den nächsten Schritt im Speicherprozess zu behandeln. Nachdem stagen kann eine Nachricht in das obere Feld eingetragen werden (vgl. Abb. 2 „Message (press Ctrl+Enter to commit)), um den Arbeitsschritt im Nachhinein besser nachvollziehen zu können, also bspw. „Aufgabe 4/12“ oder „Änderung des Titelbilds“. Danach müssen noch mit der Tastenkombination Ctrl+Enter die gespeicherten Daten commited werden.

Push:

Nun sind die gewünschten Änderungen in der lokalen Datei gespeichert und können dort auch schon angewendet werden, damit diese online zu sehen sind, muss allerdings noch der Befehl *push* verwendet werden. Damit werden alle Daten, die bereits commited wurden, an die Datei im Github gesendet. Um den Befehl auszuführen, muss ähnlich wie beim *Pull*-Befehl zuerst das Feld *Source Control* (Ctrl+Shift+G), anschließend die drei Punkte (*More Actions…*) ausgewählt und dann den *Push*-Befehl ausgeführt werden (vgl. Abb. 1, roter Kreis).

Kleiner Tipp zum Abschluss: Die Änderungen brauchen eine etwas längere Zeitspanne (zumeist allerdings unter einer Minute) um online abrufbar zu sein, also nicht zu früh verzweifeln, wenn das gewünschte Ergebnis noch nicht online zu sehen ist.