Installation für Programmieren in Python

Stand: 19. Oktober 2019

Willkommen zum Praktikum *Python*. Wir machen die praktischen Übungen mit der Skriptsprache Python (`3.6`). Wir verwenden eine halbwegs aktuelle Python-Umgebung mit ein paar üblichen Bibliotheken sowie Visual Studio Code (`code.visualstudio.com`, `1.38`) als Editor mit dem Anspruch eine integrierte Entwicklungsumgebung zu sein.

Sie können auch zu Hause beziehungsweise auf Ihrem Laptop arbeiten. Sowohl mit Linux, MacOSX als auch Windows klappt das gut. Wenn Sie es sehr einfach haben wollen, dann installieren Sie sich einfach nur VirtualBox mit dem vorbereiteten Image, das Sie schon von OOP kennen. Sie müssen dann nur noch Visual Studio Code installieren, dann klappt alles sofort. Alternativ können Sie sich die Einzelteile auf Ihren Linux, MacOSX oder Windows Rechner selbst installieren. Das dauert ein bisschen ist aber durchaus machbar. Infos und aktuelle Links finden Sie unter `www.pbma.de/swt` User `pbma`, Password `pbma19`.

VirtualBox

Wenn Sie VirtualBox (`www.virtualbox.org`), eine kostenlose Virtualisierungsumgebung, verwenden, dann enthält das Image `pb19` schon alles Notwendige auf einem freien Linux-System (Ubuntu `18.04`) und entspricht weitgehend der Umgebung in den Poolräumen. Sie brauchen einen Rechner mit Linux, MacOSX oder Windows, ein 64-Bit Betriebssystem, und mindestens 4 GB RAM (2 GB reicht nicht). Sie finden alle Dateien auf ein paar Festplatten/USB-Sticks, die die Runde machen oder im Pool unter Linux im Netzwerkordner `$HOME/export/Oop/pb19.ova` und im Netzwerkordner `Y:\Oop\pb19.ova` unter Windows. Alternativ finden Sie alle Links zum Download unter

`https://www.pbma.de/swt/pb19`.html

Folgen Sie der Anleitung `InstallOOP.pdf` unter oben genannten Link. Zusätzlich installieren Sie sich noch Visual Studio Code, wie unten unter Linux beschrieben.

Umgebung für Programmieren in Python installieren

Zunächst erläutern wir die Installation von Python und Visual Studio Code auf Windows, Linux und MacOSX. Dann plattformunabhängig wie Sie mit Visual Studio Code in Python einfach programmieren können. Sie können für eine Übungsaufgabe, bei der wir ein Stück C-Code integrieren, auch Ihre vorhandene Installation von OOP weiter verwenden

Windows

Python installieren Sie am besten als ein vollständiges Paket von

`winpython.github.io`

Wir nehmen `WinPython*3.6*Qt*` und entweder 64 Bit oder 32 Bit (je nach Ihrem Win-

dows und C/C++-Compiler), am besten 64 Bit. Wir starten es und installieren es unter `opt`, zum Beispiel

```
C:\opt\WP64-3680
```

Das dauert dann etwas. Wir fügen den Pfad *nicht* zu den Umgebungsvariablen des Systems hinzu, da sich das leider mit `MinGW` oder `Qt` für die C/C++-Programmierung beißen kann (insbesondere könnte der Debugger `gdb` nicht mehr funktionieren) oder eine andere Python-Umgebungen (Anaconda) nicht mehr lauffähig macht. Wenn wir den Umgebungsvariablen des Systems nichts hinzufügen, dann funktioniert alles andere weiter, aber wir müssen dann die Python-Umgebung für die Entwicklung immer manuell auswählen beziehungsweise unserer IDE mitteilen.

Starten Sie nach dem Entpacken den `WinPython Command Prompt` entweder über Start, Programme, WinPython oder gehen Sie in den oben genannten Ordner und klicken Sie darauf. In dem `Cmd`-Fenster geben Sie dann Folgendes ein.

```
> pip install psycopg2
> pip install virtualenv
```

Mehr muss nicht sein. Alle anderen Pakete sind schon vorhanden. Eine Registrierung ist nicht notwendig, aber eventuell sinnvoll, da dann Gruppen zum Klicken angelegt werden.

Installieren Sie sich dann Visual Studio Code (`1.38`) als integrierte Entwicklungsumgebung von

```
https://code.visualstudio.com/Download
```

Wählen Sie den Download unter Windows. Am einfachsten installieren Sie sich den `User Installer` passend (32/64 Bit). Damit kann dann nur der aktuell angemeldete Benutzer arbeiten. Unter Windows sollte man automatisch einen Desktop-Eintrag beziehungsweise einen Eintrag in einem Menu erhalten.

Da Sie Python nicht in den Pfad aufgenommen haben, müssen Sie später den folgenden Eintrag in Ihren `settings.json` machen

```
"python.pythonPath": "C:\\opt\\WP64-3680\\python-3.6.8.amd64\\python.exe",
"code-runner.respectShebang": false,
```

Der zweite Eintrag ist notwendig, da unter Windows alle Pfade (und auch alles andere) ein bisschen anders sind. Details dazu später unter „Arbeiten mit Visual Studio Code"'.

Linux

Python müsste schon installiert sein. Wir verwenden Python 3 mit einigen Paketen. Die folgende Zeilen müssten die fehlenden Pakete installieren.

```
$ sudo apt-get install python3-dev python3-virtualenv \
  python3.6-examples python3-tk python3-pip python3-numpy \
  python3-matplotlib python3-cffi python3-scipy ipython3 \
  python3-httplib2 python3-netifaces python3-pandas \
  python3-bs4 python3-requests python3-six python3-werkzeug \
  python3-psycopg2 python3-reportlab python3-setuptools \
  python3-simpy python3-pil python3-lxml python3-yaml
```

Falls Sie kein `apt-get` haben müssen Sie sich mit der Installationsmethode auf Ihrem System auseinandersetzen. Das geht aber auch nachträglich noch, wenn wir es brauchen.

Installieren Sie sich dann Visual Studio Code (`1.38`) als integrierte Entwicklungsumgebung von

```
https://code.visualstudio.com/Download
```

Wählen Sie den Download unter Linux. Vermutlich kommen Sie mit dem `.deb` Paket für 64 Bit einfach an das Ziel (für die anderen Optionen müssen Sie selbst schauen). Wenn Sie es heruntergeladen haben, dann öffnen Sie ein Terminal und installieren Sie sich `gdebi` zum Installieren und dann Visual Studio Code.

```
$ sudo apt-get install gdebi
$ sudo gdebi ~/Schreibtisch/code*.deb
```

wobei Sie eventuell Ihr Passwort eingeben müssen und alles mit Ja (`y`) bestätigen. Es kann sein, dass die Datei `code ... .deb` auch woanders ist unter `/Downloads` oder `/tmp`, schauen Sie im Firefox bei den Downloads und klicken Sie auf das Ordner Symbol hinter der Datei. Unter Linux sollten Sie auch einen Menu-Eintrag erhalten (unter Entwicklung), von dem Sie dann sich selbst einen Desktop-Eintrag machen können. Sie können die Entwicklungsumgebung immer vom Terminal aus starten mit:

```
$ code
```

Lesen Sie bei „Arbeiten mit Visual Studio Code"'weiter.

MacOSX

Das unter MacOSX installierte Python3 ist zwar aktuell, aber leider nicht nutzbar[1]. Um unter MacOSX ein aktuelles Python zu installieren, bemühen Sie `brew`. Informationen gibt es unter:

```
https://brew.sh
```

Machen Sie eine Konsole auf und geben Sie in einer Zeile Folgendes ein.

```
/usr/bin/ruby -e \
"$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)
```

Danach können Sie einfach Sachen installieren, die fehlen. Auf dem Mac sollte bei `brew` Python3.7* aktuell sein, was wir auch nehmen können.

```
$ brew install python3
```

Achten Sie darauf, dass das richtige `python3` kommt, wenn Sie es starten. Es ist das, was unter `/usr/local/bin/python3` kommt und nicht das unter `/usr/bin` (das ist das vom System). Wenn Sie versehentlich ein System-Python haben, dann verwenden Sie `/usr/local/bin/python3`. Dann installieren oder aktualisieren wir `pip`

```
$ /usr/local/bin/pip3 install --upgrade pip
```

und installieren die übrigen Pakete

[1] Selbst wenn man die meisten Pakete damit zum laufen kriegt scheitert man am Ende am GUI

```
$ /usr/local/bin/pip3 install ipython virtualenv numpy matplotlib cffi scipy
$ /usr/local/bin/pip3 install httplib2 netifaces pandas bs4 requests six
$ /usr/local/bin/pip3 install werkzeug psycopg2-binary
$ /usr/local/bin/pip3 install reportlab setuptools simpy pillow lxml
```

Für manche Aufgaben helfen dann noch ein paar Tools. Mit `brew install pqiv` hat man sogar ein Tool mit dem man Bilder auf dem Mac normal anschauen kann. Zum Beispiel zeigt dann

```
pqiv -t lena.pgm out.pgm
```

die beiden Bilder groß skaliert an und man kann mit der Leertaste zwischen den Bildern wechseln und `q` beendet den Viewer. ImageMagick gibt es mit `brew install imagemagick` (Konsolentool, um Bilder zu konvertieren und bearbeiten). Mit `brew install fortune` gibt es auch das `fortune`-Programm. Die Sprüche sind dann unter `/usr/local/share/games/fortun`

Interessanterweise sollte sich über `brew` auch Visual Studio Code installieren lassen mit

```
$ brew cask install visual-studio-code
```

Wenn nicht, dann eben wieder unter dem Link

```
https://code.visualstudio.com/Download
```

und dann den Download unter `Mac` wählen. Vermutlich findet sich dann das passende Python 3 unter `/usr/local/bin` was uns zu den folgenden Einträgen in `settings.json` führt.

```
"python.pythonPath": "/usr/local/bin/python3",
"code-runner.respectShebang": false,
```

Auch auf dem Mac verzichten wir auf den Shebang, da `/usr/bin/python3` meist nicht existiert oder ein altes System-Python ist. Aber unter den Rechnern im PC-Pool ist es das Richtige.

Auch auf dem Mac sollten Sie einen Menu-Eintrag erhalten. Sie können vermutlich die Entwicklungsumgebung aber auch vom Terminal aus starten mit

```
$ code
```

wenn `brew` seinen Pfad hinzugefügt hat. Ab Catalina wird sofort erkannt das das keine Apple-Software und sich direkt übelst beschwert (kann nicht geöffnet werden, Schadsoftware, an Entwickler wenden, ...), dass nicht Apple-Software ausgeführt wird. Es gibt ein Issue `#74782` was eventuell weiterhelfen kann. Offensichtlich muss nicht Apple-Software "notarisiert" werden, sie können dazu wohl was bei Sicherheit in den Systemeinstellungen finden.

Lesen Sie bei „Arbeiten mit Visual Studio Code"'weiter.

Arbeiten mit Visual Studio Code

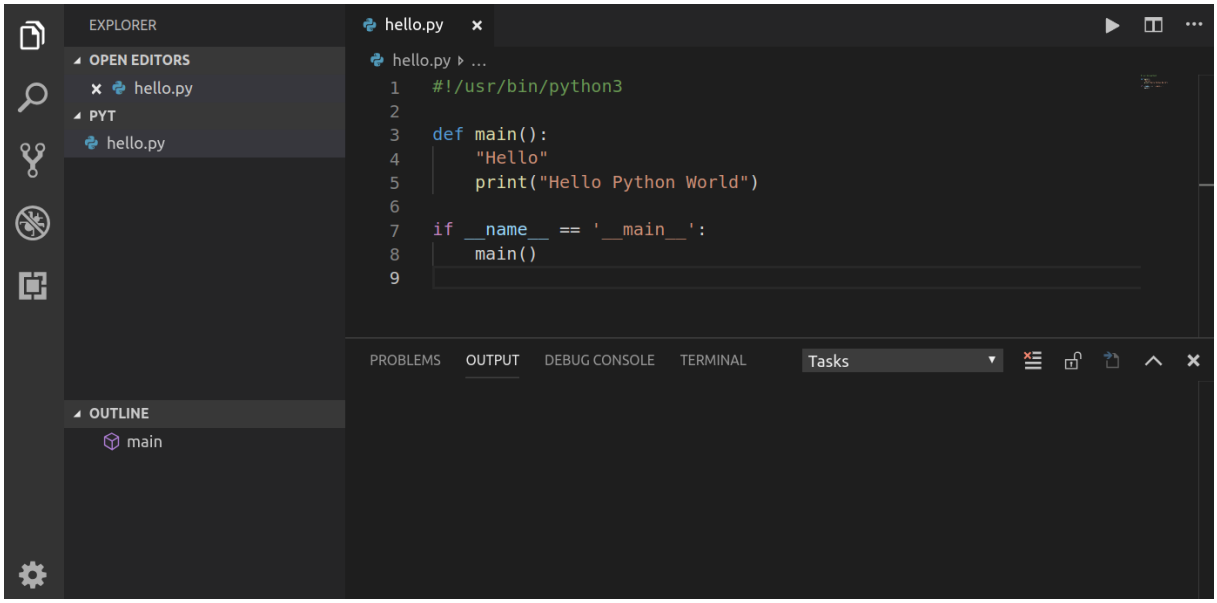
Visual Studio Code ist ein plattformunabhängiger Editor für Programmierer, der die wichtigsten Features einer integrierten Entwicklungsumgebung unterstützt ohne überladen zu wirken.

Übersicht

Nach dem Start sehen Sie auf links sechs Icons.

- Explorer: Die Standardumgebung beim Programmieren, Zugriff auf Ihre Dateien im Ordner, in dem Sie entwickeln
- Suche: Suchen und Ersetzen
- Versionskontrolle: Nutzen Sie die integrierte Git-Umgebung, um Ihre Python- und andere Dateien einzuchecken und zu aktualisieren
- Debug: Der Debugger, geht prima mit Python
- Extensions: Anpassen von Visual Studio Code an die jeweilige Programmierplattform, damit installieren wir gleich ein paar Erweiterungen
- Settings: Einstellungen

Sie können das Willkommensfenster beim Start deaktivieren (auf der Seite links unten) und schließen. Der wichtigste Keyboard-Shortcut ist `Ctrl-Shift-P`, was oben einen Eingabezeile öffnet. Damit können Sie alle Kommandos auch direkt eingeben und ausführen und müssen (erst Mal) keine Shortcuts lernen.



Erweiterungen

Wir gehen auf Erweiterungen und suchen nach `Python`. Der erste Treffer (sortiert nach Popularität) beziehungsweise die Erweiterung von Microsoft selbst ist die richtige. Klicken

Sie auf `Install` und warten sie kurz[2]. Zusätzlich brauchen wir noch `Code Runner` (von Jun Han), das Sie genauso installieren. Das reicht dann schon. Wenn `code` wegen eines `linter`s nervt, dann können Sie das ignorieren.

Jetzt geben Sie `Ctrl-Shift-P` ein und tippen oben in der Zeile `Settings` und wählen die Option

```
Preferences: Open Settings (JSON)
```

aus. Irgendwie können Sie sich sicherlich auch durch alles durch klicken, aber so geht es deutlich einfacher. Visual Studio Code öffnet dann eine Datei mit dem sprechenden Namen `settings.json` in der Sie Ihre Einstellungen eintragen. Das JSON-Format erinnert an eine (ist eine) Map, also eine Abbildung von Schlüssel auf Werte (Eintrag), wobei die Schlüssel immer Strings sind und die Werte boolean, Zahlen, Strings oder wieder Maps. Schlüssel sind durch : von den Werten getrennt, wir haben immer ein Komma , nach einem Eintrag. Eine Map ist von geschweiften Klammern umschlossen. Sie können folgende Einträge hinzufügen.

```
{
    "telemetry.enableTelemetry": false,
    "code-runner.enableAppInsights": false,
    "window.zoomLevel": 2,
    "workbench.startupEditor": "none",
    "python.pythonPath": "python3",
    "code-runner.clearPreviousOutput": true,
    "code-runner.showExecutionMessage": false,
    "code-runner.respectShebang": false,
    "code-runner.executorMap": {
        "python": "$pythonPath -u $fullFileName",
    }
}
```

Die ersten beiden begrenzen den Drang der IDE nach Hause zu telefonieren. Der Zoom-Level erlaubt die Ansicht zu skalieren, die meisten von Ihnen werden 1 statt 2 präferieren, für die Demos nehme ich immer 2 oder 3. Beim `startupEditor` können Sie auch `none` eintragen, dann wird keine leere Datei geöffnet oder Willkommen gezeigt. Spannend ist dann der `pythonPath`. Unter Linux ist der Eintrag `python3` richtig, unter Windows beziehungsweise MacOSX machen Sie hier den Eintrag wie im jeweiligen Abschnitt beschrieben. Die nächsten drei Zeilen konfigurieren die Möglichkeit Ihre Python-Skripte laufen zu lassen. Ziel ist wenig sinnlose Extra-Ausgaben und zumindest für MacOSX und Windows kein Shebang[3]. Die `executorMap` konfiguriert, dass wir das eingestellte Python nehmen und dabei die aktuelle Python-Datei ausführen und ist auf allen Plattformen okay.

[2] wirklich nur kurz, das ist alles schnell und fühlt sich daher angenehm an

[3] Der Shebang `#!` kommt noch in der Vorlesung, für jetzt ist das erst mal nur ein Kommentar.

Hello in Python

Wir arbeiten mit Visual Studio Code immer mit Ordnern, später im Kurs alles in einen Ordner (ohne!) weitere Hierarchie. Um das jetzt auszuprobieren, erstellen wir auf dem Desktop einen neuen Ordner (außerhalb von Visual Studio Code) mit einem einfachen Namen (keine Leerzeichen oder Sonderzeichen) wie zum Beispiel `pyt`. Dann öffnen wir den Ordner von Visual Studio Code aus (File -> Open Folder). Im Explorer sehen wir dann `PYT` und wenn wir über die Zeile gehen, dann kommt ein Icon für ein `New File`, was wir betätigen. Wir geben dann den Namen der neuen Datei `hello.py` direkt unterhalb im Eingabefeld ein. Der Editor öffnet sich und wir schreiben das Folgende hinein.

```
#!/usr/bin/python3

def main():
    "Hello"
    print("Hello Python World")

if __name__ == '__main__':
    main()
```

Beachten Sie, dass in Python das Einrücken des Code **mit 4 Leerzeichen** je Block Teil der Syntax[4] ist. Für ein `hello world` würde uns Zeile 5 mit dem `print` reichen, aber wir geben etwas mehr ein. Sie werden merken, dass Visual Studio Code Ihnen schon (versucht) beim Tippen zu helfen (Intellisense). Eine Vervollständigung oder Details zu einer Vervollständigung fordern Sie an mit `Ctrl-Space`.

Sie können die Datei nun einfach laufen lassen `Ctrl-F5`. Wenn alles klappt, dann öffnet sich unten ein Terminal (gut) und nach anderen Ausgaben erhalten Sie auch das kanonische

```
Hello Python World
```

Sie können das Terminal unter Windows mit `Ctrl-ö` und unter Linux mit `Ctrl-`` (das ` ist ein Backtick, also Shift und neben dem Backspace rechts oben) auf und zu klappen.

Um ein Programm auszuführen, ist es jedoch eleganter den `Code Runner` zu verwenden. Das geht wiederum am einfachsten durch klicken auf das Dreieck (Play Button) rechts oben. Jetzt erscheint (weil wir das so konfiguriert haben) nur noch die eigentlich Ausgabe. Diese nicht mehr im Terminal, sondern nur noch im Output Tab aber auch unten. Der passende Keyboard Shortcut ist `Ctrl-Alt-N`.

Der integrierte Debugger ist auch einfach zu bedienen. Klicken Sie links neben den Zeilennummern in Zeile 5 (der Zeile mit dem `print`, dann erscheint ein roter Punkt, ein Breakpoint. Jetzt klicken Sie auf den Käfer links in der Spalte, dann erscheint oben links grüner Play-Button. Beim ersten Mal müssen Sie sich eine `Debug Configuration` aussuchen. Wir nehmen `Python File`. Wir laufen direkt bis zu Breakpoint und können uns links Variablen anschauen, Watch-Ausdrücke anlegen oder den Aufrufstack sehen. Oben in der Mitte sind die üblichen 5 Knöpfe beim Debugging:

[4] Okay, es ist egal wie viel Leerzeichen (oder ein Tab) aber einheitlich, für uns ist einheitlich 4 Leerzeichen.

Continue (F5), Step Over (F10), Step Into (F11), Step Out, Restart und Stop.

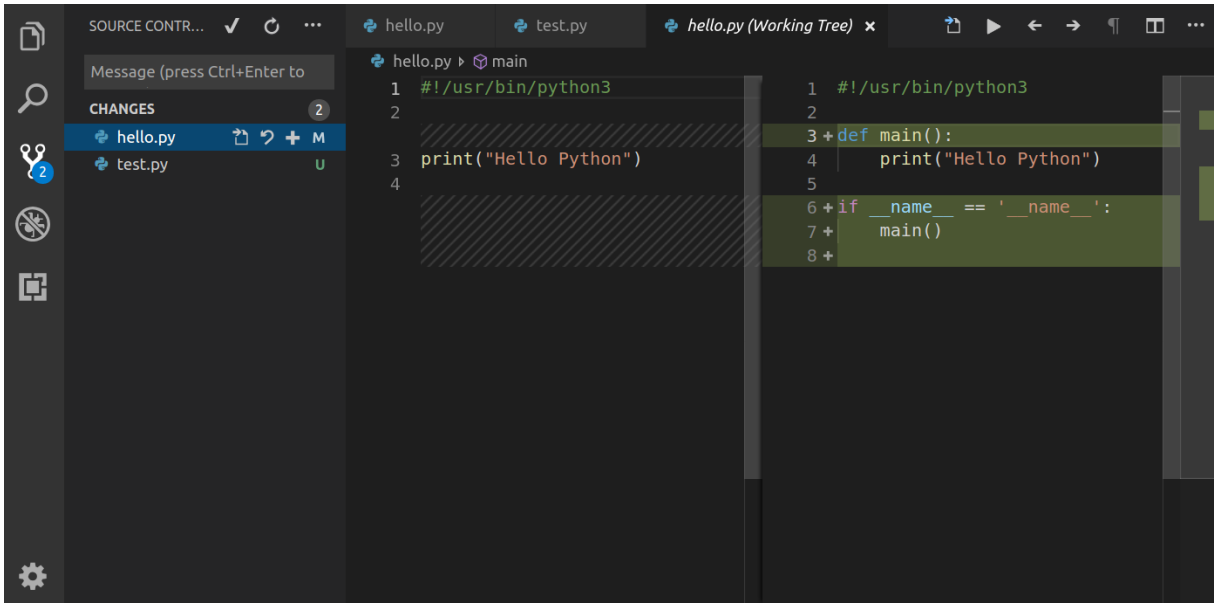
Damit sollten Sie auf jeden Fall schon arbeiten können. Ich habe auf die meisten Screenshots verzichtet, da der von Ihnen so geliebte Dark Mode bei einem Ausdruck nur Toner verschwendet. Im Web gibt es Tonnen von Dokumentation und viele unterschiedliche Videos mit schwankender Qualität, aber um zu sehen wie man etwas mit einer IDE macht ist Video das richtige Medium – lassen Sie sich ruhig inspirieren.

Versionskontrollsystem Git

Wir nutzen für die Veranstaltung ein Git-Repository. Verwenden Sie das, um einfach auf den PCs im Praktikum zu arbeiten, zu Hause auf dem Desktop oder unterwegs auf dem Laptop – immer abwechselnd, welches Gerät auch immer gerade verfügbar ist. Git-Unterstützung ist schon bei Visual Studio Code mit dabei, aber Sie brauchen trotzdem auch eine Git-Installation. Dazu gibt es für jede Plattform entsprechende Dokumentation. Ich empfehle Ihnen initial Ihr Repository manuell auszuchecken, zum Beispiel mit

```
$ git clone https://scm.inftech.hs-mannheim.de/scm/19pythxx
```

wobei Sie `xx` mit der Ihnen zugewiesenen Nummer ersetzen. Wenn Sie in einem Ordner mit Versionskontrolle arbeiten, dann erscheinen in Visual Studio Code entsprechende Icons und Hinweise. Selbst im Explorer sehen Sie Grün `U` für `Unversioned` und Orange `M` für `Modified`. Wenn Sie auf die Versionskontrollansicht gehen, dann sehen Sie alle geänderten Dateien. Bei Klick auf die Datei eine sehr schöne Ansicht der Änderungen.



Über die Plus-Knöpfe können Sie einzelne Dateien oder alle in dem Changes-Block in die Staging-Area übernehmen. Das Häkchen ganz oben committed dann, eine Commit-

Message können Sie direkt eingeben. Für den Push (nicht vergessen!) müssen Sie die drei Punkte bemühen.