Verwendung GIT mit Unity

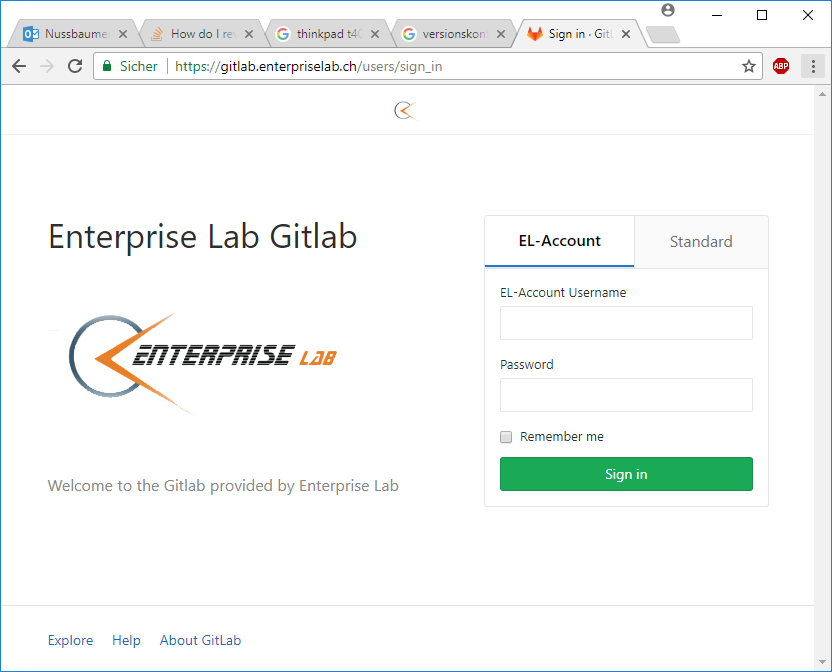
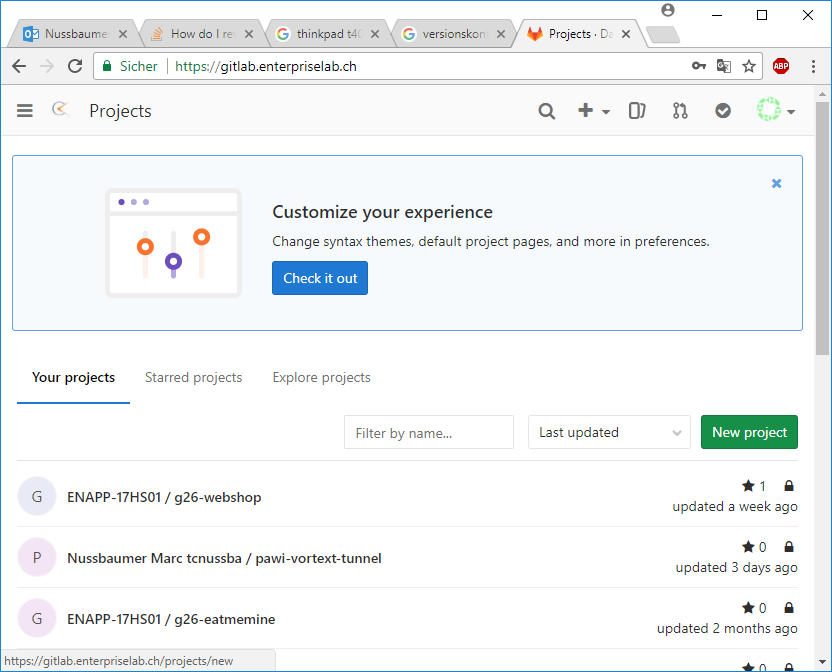
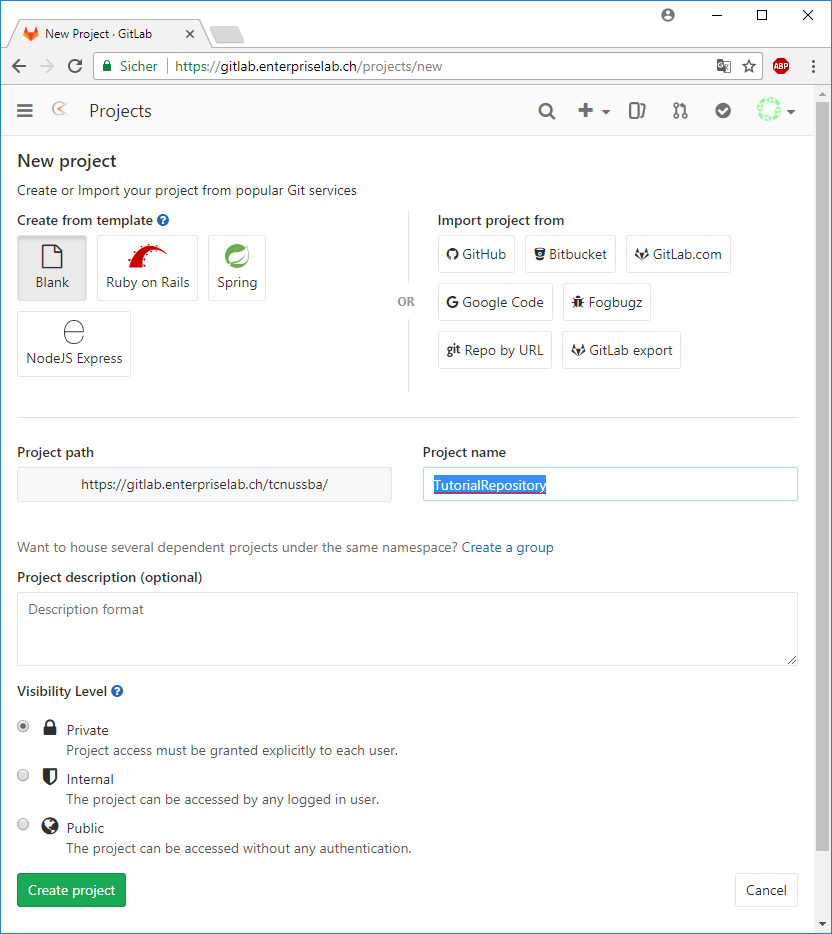
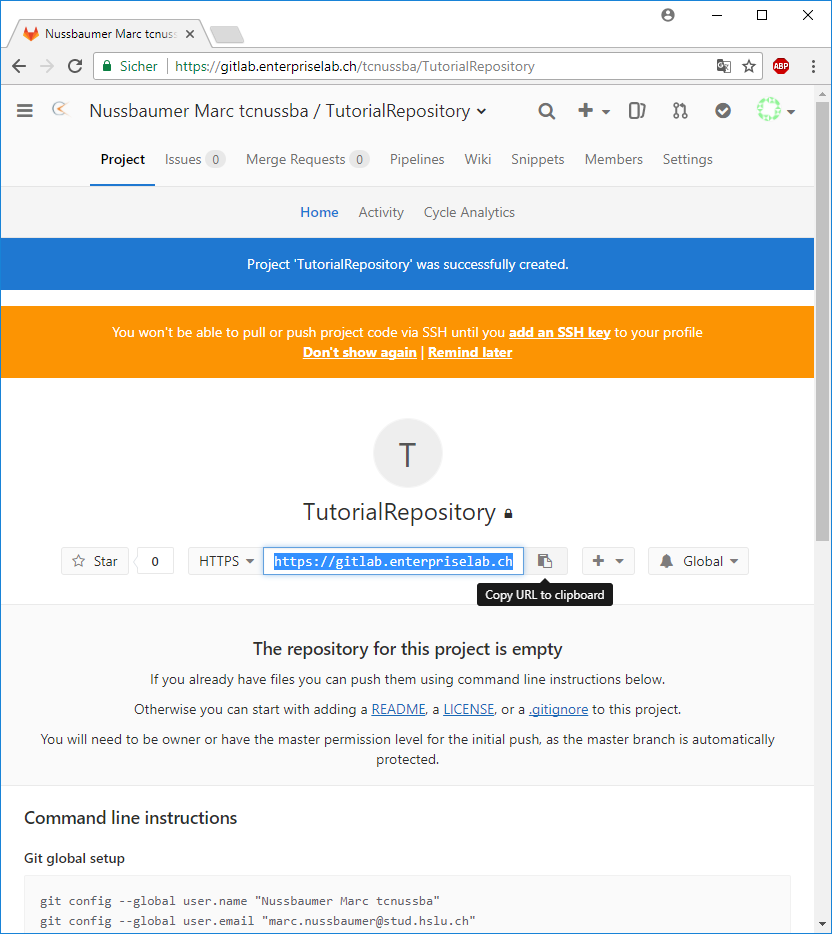
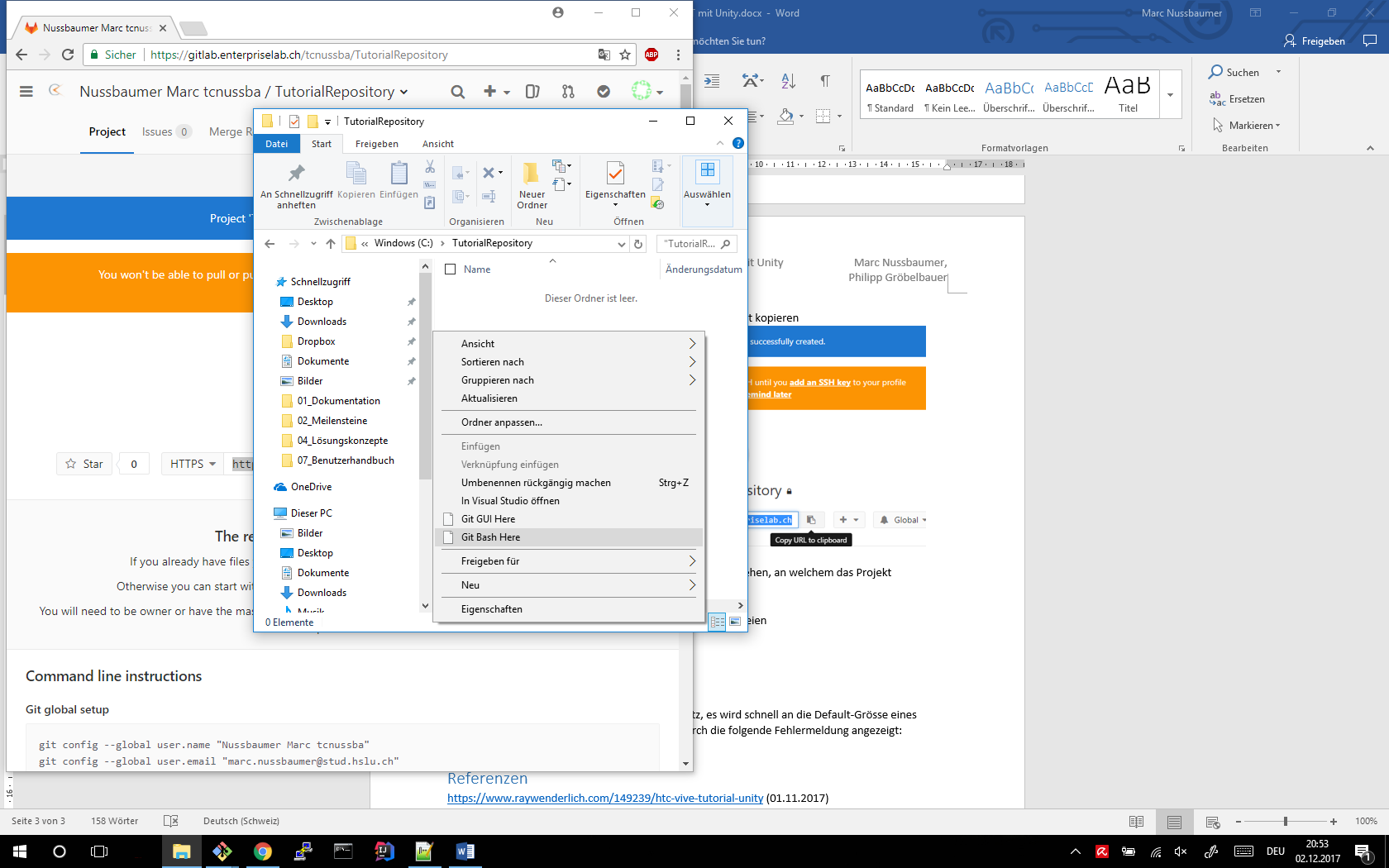
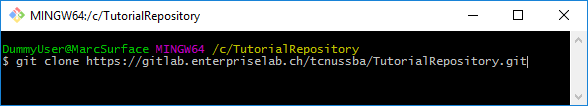
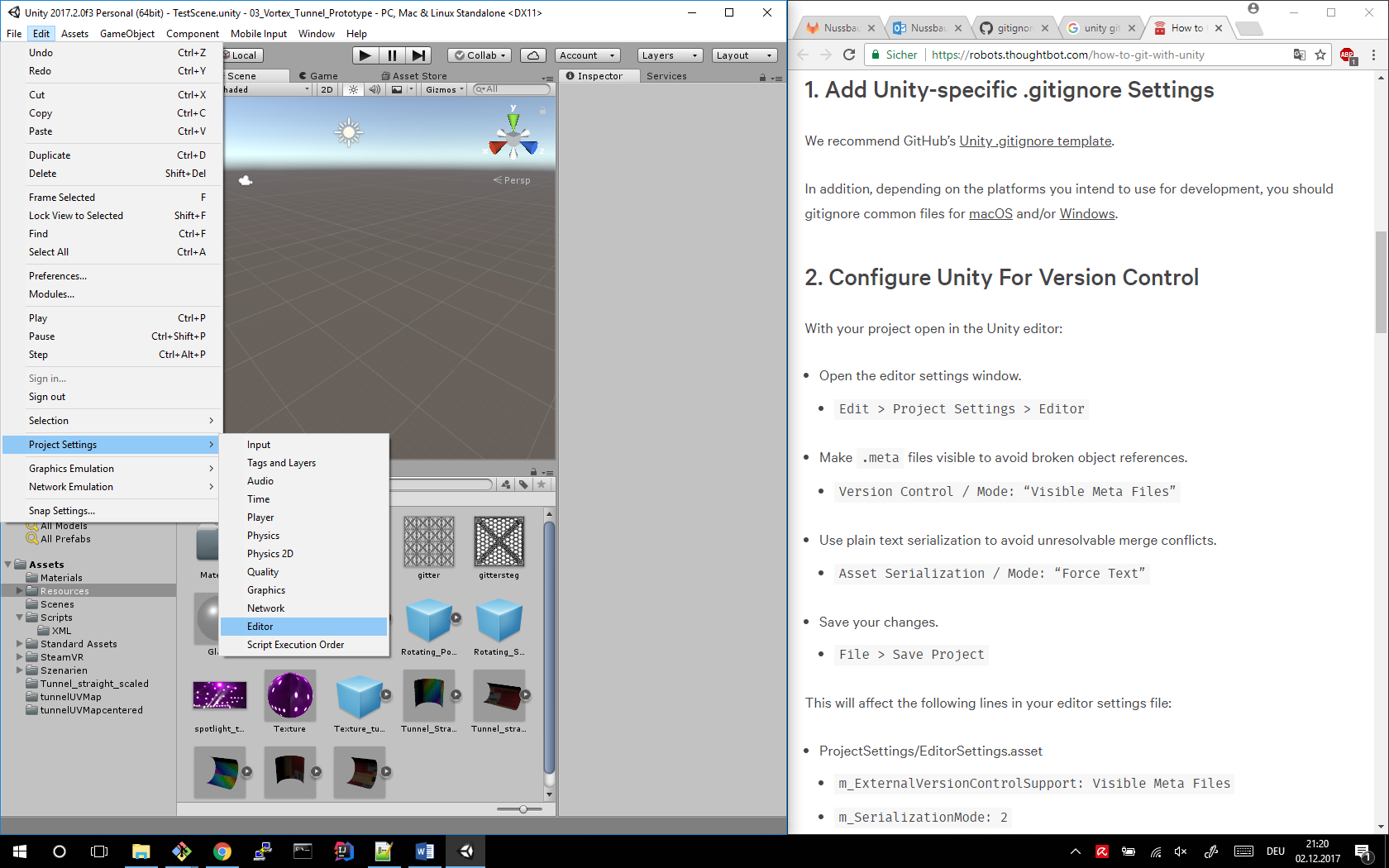
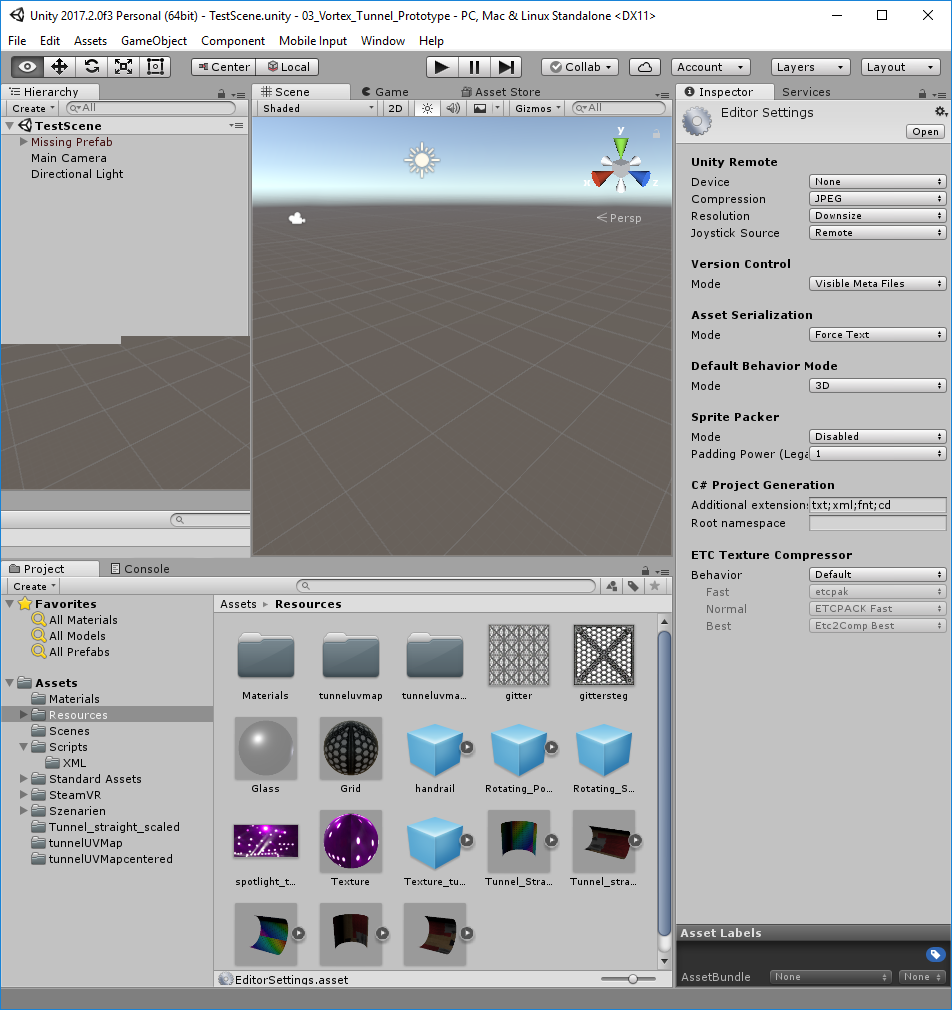
# Benötigte Vorkenntnisse

* Grundlagen Verwendung von Versionskontrollsysteme
* Grundlagen Verwendung GIT über Shell
* Grundlagen Verwendung Unity
* Verwendung von Gitlab

# Ziele

* Verwaltung eines Unity-Projektes in GIT

# Anleitung

1. Erstellung eines neuen Repository  
   Grundsätzlich ist jede Plattform welche GIT anbietet geeignet, in diesem Tutorial verwenden wir Gitlab. Dies da so momentan die Schulprojekte betreut werden.  
   1. Bei der Gitlab-Plattform mit deinem Benutzer anmelden
   2. Anschliessend unter «Your Projects» den Punkt «New Project» wählen  
      
   3. Die Standard-Einstellungen so belassen und nur den Projekt-Namen im Feld «Project Name» auf den gewünschten Namen anpassen. Anschliessend auf «Create project» klicken  
      
2. Initialisierung des Projektes
   1. Im neu erstellten Repository die URL wie folgt kopieren  
      
   2. Anschliessen die Git Bash Konsole im Ordner öffnen, in welchem das Projekt später liegen soll  
      
   3. Anschliessend das leere Repository herunterladen per  
      
3. Erstelle oder Kopiere ein Unity-Projekt in den neu erstellten Ordner
   1. Öffnen die Editor-Settings des Unity-Projektes  
      
   2. Setze unter «Version» den «Mode» auf «Visible Meta Files» und unter «Asset Serialization» den «Mode» auf «Force Text»  
      

Dies ist notwendig, damit Unity diese Änderungen nicht in notwendigen Projektdateien anpasst und somit zu Versionskonflikten führt.

1. Erstellung einer .gitignore-Datei für Unity-Projektdateien
   1. Erstelle eine Text-Datei mit Namen gitignore.txt
   2. Kopiere folgenden Inhalt in diese Textdatei

[Ll]ibrary/

[Tt]emp/

[Oo]bj/

[Bb]uild/

[Bb]uilds/

Assets/AssetStoreTools\*

# Visual Studio 2015 cache directory

.vs/

# Autogenerated VS/MD/Consulo solution and project files

ExportedObj/

.consulo/

\*.csproj

\*.unityproj

\*.sln

\*.suo

\*.tmp

\*.user

\*.userprefs

\*.pidb

\*.booproj

\*.svd

\*.pdb

Physics2DSettings.asset

DynamicsManager.asset

# Unity3D generated meta files

\*.pidb.meta

# Unity3D Generated File On Crash Reports

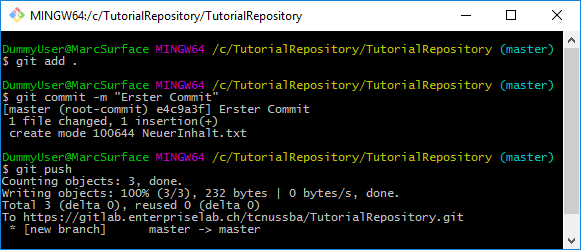
sysinfo.txt

# Builds

\*.apk

\*.unitypackage

* 1. Benenne die Datei zu «**.gitignore.**» um. Ja Punkt-gitignore-Punkt, dadurch wird die Datei zu «.gitignore». Ja Windows ist hier komisch.

1. Upload von Änderungen
   1. Im Ordner die Git-Bash öffnen und folgende Befehle eingeben, wobei «Erster Commit» eine beliebige Nachricht für den Commit sein kann  
      

# Mögliche Stolpersteine

* Unity-Projekte benötigen viel Speicherplatz, es wird schnell an die Default-Grösse eines Gitlab-Repository gestossen. Dies wird durch die folgende Fehlermeldung angezeigt:

error: unpack failed: unable to create temporary object directory

To https://gitlab.enterpriselab.ch/tcnussba/pawi-vortext-tunnel.git

 ! [remote rejected] master -> master (unpacker error)

* Falls die .gitignore-Datei nicht verwendet wird so gibt es bei jedem Push eines anderen Benutzers ein Konflikt. Dies passiert weil Unity sehr viele temporäre lokale Dateien erzeugt.
* Sind die Editor-Settings nicht wie im Tutorial beschrieben gesetzt so ändert Unity Projekt-Assets was zu Konflikten führt beim Push/Pull.

# Referenzen

<https://github.com/github/gitignore/blob/master/Unity.gitignore> (02.12.2017)  
<https://robots.thoughtbot.com/how-to-git-with-unity> (02.12.2017)