Damit viele neue User wissen was Github ist und was Github kann, hier mal eine kurze Beschreibung

FAQ Github (Deutsch)

Einleitung

Was ist eigentlich git und GitHub (www.github.com)

GitHub ist sozusagen eine **Web-Anbindung** für die Versionsverwaltung "git"

Eine Versionsverwaltung dient dazu, den Verlauf eines Projektes zu protokollieren, indem automatisch alle Änderungen gespeichert werden und gegen die ursprüngliche Version verglichen werden können ("diff")

In der Regel werden solche Versionierungs-Systeme in der Software-Entwicklung angewandt, können aber auch für alle anderen Arten von dynamisch veränderten Projekten gebraucht werden, z.B. Bücher, Fahrpläne, Musik-Kompositionen u.v.m.

Bei Projekten mit mehreren oder gar vielen Entwicklern kann GitHub die Schnittstelle bilden zwischen den einzelnen Beteiligten, die jeweils lokal die Versionen mit git verwalten
Außerdem bietet es eine grafische Oberfläche und viele auf die Teamarbeit bezogene Zusatzfunktionen

Zu **Git** sollte man noch erwähnen, dass git ein Tool unter Linux ist um Versionskontrolle zu betreiben. Github dagegen ist eine Plattform für Git Repositories
Beides ist also nicht das gleiche.
Git ist das (Kommandozeilen)-Tool und Github ist der Service für Projekte, die git benutzen

Git kann unter Ubuntu (z. B.) via*:* `sudo apt-get install git` installiert werden
Wie man git dann korrekt verwendet ist im folgenden verlinkten Buch ziemlich gut erklärt:

git-scm.com/book/de/v1

Git ist auch für alle Plattformen verfügbar: git-scm.com/download

Es gibt etliche andere Tools, die man unter verschiedenen Betriebssystemen für Github verwenden kann, z.B. Github Desktop
Smartgit ist zum Beispiel auch eines. Man darf aber nicht vergessen, dass viele GUI Tools nicht ALLE Funktionalitäten von git bieten. Die Kommandozeile ist die einzige Option um alle Möglichkeiten auszuschöpfen. Klar, manches wird man nie brauchen, aber es dennoch immer wichtig, dass man das wenigstens mal erwähnt

Inhaltsverzeichnis

1.Was ist Github

Github ist eine Entwickler Plattform, auf der viele User gemeinsam an einem Project arbeiten können
Hier liegen meistens die Daten von Repositories, Addons usw.
Es können aber alle Arten von Daten hochgeladen werden

Wichtig ist noch zu erwähnen, das Euer Github „öffentlich“ ist, dass heiß jeder kann es sehen
Will man das nicht, so muss man bezahlen

Es gibt für den PC die Anwendung Github Desktop. Damit können die Daten auf dem PC bearbeitet werden und dann hochgeladen werden

Hilfe beim Arbeiten mit Github gibt es hier: https://help.github.com/
Eine kurz Einführung in Github findet Ihr hier: https://guides.github.com/introduction/flow/

2.Wer kann auf Github arbeiten

Jeder User, der sich auf www.github.com anmeldet, kann auf /mit Github arbeiten.
Github ist kostenlos („öffentlich“). Es können eine Vielzahl an Daten hochgeladen werden
Es ist immer ersichtlich, wer die Daten bearbeitet hat und was genau geändert wurde

3.Wie kann ich ein Project erstellen

Dazu auf Deiner Githubseite anmelden

1. In der Menüleiste auf „Repositories“ klicken
2. Dann rechts auf „New“ klicken
3. Nun bei „Repository name“ den Namen Eures Projects eingeben
   Public (öffentlich) anklicken oder wenn Ihr bezahlen wollt dann privat
   Und unbedingt „Initialize this repository with a README“ anklicken
4. Dann „Create Repository“ klicken und das Project ist angelegt

https://help.github.com/articles/create-a-repo/

4.Wo sehe ich die Änderungen

Besucht eine beliebige Githubseite, z.B. https://github.com/kodinerds/

Das „blau“ geschrieben sind die angebotenen Repositories oder Addons
Klickt ihr auf eines drauf, ist der Inhalt dieses Repos/Addons zu sehen

Hier findet ihr nun 3 Spalten:
Links: die Datei Mitte: die Änderungen an der Datei rechts: zuletzt bearbeitet

Mit einem Klick in der mittleren Spalte, seht Ihr die Änderungen
Rot bedeutet gelöscht, Grün bedeutet neu hinzugefügt

5.Wie kann ich an einem Project mitarbeiten wenn ich keine Schreibrechte/Berechtigung für diese Github Seite habe

Da ist eigentlich ganz einfach
Wenn Ihr keine Schreibrechte/Berechtigung für diese Seite habt, so müsst Ihr von dieser Github Seite ein „Fork“ machen
Das ist sozusagen eine 1:1 Kopie, welche in wenigen Sekunden auf Eurer Github Seite erscheint

Um einen „Fork“ zu erzeugen, klickt Ihr einfach rechts oben auf das „Fork“ Symbol

Nun könnt Ihr an dieser Seite (welche sich ja nun auf Eurem Github befindet) Änderungen durchführen und diese dann beim Entwickler einreichen

https://help.github.com/articles/fork-a-repo/

6.Wie kann ich Änderungen (Pull Requests) beim Entwickler einreichen

Das Einreichen einer Änderung bezeichnet man als „Pull Requests“
Das bedeutet, dass die Änderungen, welche Ihr auf Eurer Github Seite (auf dem „Fork“) gemacht habt, direkt auf der Github Seite des Entwickler erscheint, in der Kategorie „Pull Requests“ (am oberen Rand in Github)

Der Entwickler kann diese „Einreichung“ annehmen oder ablehnen.
Nimmt er sie an, so wird diese (meistens) so übernommen wie ihr sie erstellt habt

https://help.github.com/articles/creating-a-pull-request-from-a-fork/

7.Wie kann ich ein „PullRequest“ annehmen

Die eingereichten Änderungen eines Users bei Euch, seht Ihr in der Menüleiste „Pull Requests“
Wählt den „Pull“, welchen Ihr bearbeiten bzw. mit Eurem Project „mergen“ (zusammenfügen) wollt

Wenn die eingereichte Branche nicht mit Eurer Branche übereinstimmt (welche zusammen geführt werden sollen), so könnt Ihr das ganz einfach noch anpassen:

Compare changes
Compare changes across branches, commits, tags, and more below

Hier steht darunter:
Base: master Base: master……

Das linke ist die Branche von Euch, wohin der „PullRequest“ eingefügt wird
Das rechte die Branch des anderen Users

Ändert die linke Seite so, wir Ihr es benötigt (master, nightly, beta usw…)

Rechts am Rand steht „Edit“. Nach einem Klick darauf, könnt Ihr Eure Branch anpassen

https://help.github.com/articles/changing-the-base-branch-of-a-pull-request/

Bei „This Branch has no conflicts with the base branch“ könnt Ihr nun auf den grünen Knopf „Merge Pull Requests“ klicken

Zum Schluss klickt Ihr unten auf „Confirm Merge“

https://help.github.com/articles/merging-a-pull-request/

8.Wie bekomme ich Schreibrechte/Berechtigung, bzw. wie kann ich einem User Rechte erteilen für ein Github Project

Diese Berechtigung, kann nur der Inhaber der Github Project Seite erteilen

Dazu wir das jeweilige Project ausgewählt, am oberen Rand auf „Settings“
Dann ein Klick auf „Collaboartos“

Jetzt steh auf der rechten Seite: „Search by username, full name or email address“
Hier sucht ihr den User, welchen Ihr Berechtigen wollt

Dieser bekommt dann eine E-Mail mit dem Aktivierungslink. Sobald dieser angenommen wird, hat der User Schreibrechte (nur) für diese eine Project

Bekommt der User die E-Mail nicht, so kann ihm der Github Inhaber den Link auch so geben
Steht unter „Collaborators“, neben dem eben angelegten User: „Copy invite Link“

https://help.github.com/articles/inviting-collaborators-to-a-personal-repository/

9.Wie können Daten hochgeladen werden bzw. neu erstellt werden

Zum Hochladen wählt ihr ein Project aus und klickt auf „Upload Files“

Die maximale Größe pro Upload hierbei beträgt 25MB.
Wollt ihr eine größere Datenmenge hochladen, so benötigt ihr Github Desktop. Damit können maximal 100MB pro Upload hochgeladen werden

https://help.github.com/articles/adding-a-file-to-a-repository/

Zum Erstellen einer Datei klickt Ihr auf „Create new file“
Auch hier für benötigt Ihr die oben erwähnte Berechtigung (außer es ist Euer Github)

https://help.github.com/articles/creating-new-files/

10.Wie kann ich eine „Branch“ erstellen

Dazu öffnet Ihr das Project (Repo, Addons usw.)
Dann wählt Ihr das Feld „Branch“

Es öffnet sich nun ein (weißes) Textfeld
In diese Feld könnt Ihr nun den Namen einer neuen Branch eingeben

https://help.github.com/articles/creating-and-deleting-branches-within-your-repository/

11.Wie lösche ich eine“Branch“

Wenn das Project geöffnet ist, findet Ihr eine Menüleiste die folgendes beinhaltet:
commits, branches, releases, contributors, GPL-3.0

Hier wählt ihr „branches“

Wählt die zu löschende Branche und klickt auf den Mistkübel

Die „Master Branche“ ist die default (Standard) Branche. Um diese löschen zu können, muss vorher eine neu default branch festgelegt werden (change default branch)

12.Wie kann ich ein Project löschen

Dazu besucht ihr Eure Github Seite, und wählt das Project welche gelöscht werden soll
Klickt oben auf „Settings“

Ganz nach unten Scrollen und „Delete this Repository“

In dem sich nun öffnenden Fenster muss der genaue Name des zu löschenden Projects eingegeben werden und bestätigen

https://help.github.com/articles/deleting-a-repository/

13. Github Historie (aufrufen), Wie bekommt man alte Versionen

Von jedem Github Project, kann eine Historie aufgerufen werden. Hier sind auch bereits gelöschte Dateien zu finden

Dazu öffnet man Github und das entsprechende Project

In der Menüleiste findet Ihr: commits, branches, releases, contributors, GPL-3.0

Hier den Menüpunkt „Commits“ anklicken

Auf der nun angezeigten Seite, sieht man, wer/wann welche Änderung gemacht hat

Ganz unten kann man weiter zurückgehen in der Zeit, durch Klick auf „Older“

Über die blauen Felder auf der rechten Seite, kann man zu genau dieser Änderung springen:

<>: Springt zum Repo/Addon und zeigt es so an, wie es mal war. Kann dann auch genauso heruntergeladen werden

ddf12nhg: beim Klick auf eine Buchstaben/Zahlenkombination, springt man direkt in die geänderte Datei von damals. Rot bedeutet gelöscht, Grün hinzugefügt
Mit einem Klick auf „Browse files“ kommt man zum Repo/Addon von damals

Eine super genaue Anleitung dazu und wie man nach Dateien suchen kann, findet ihr hier: Link

Im folgendem hab ich die Anleitung aus obigem Link mal hier eingefügt:

Wie bekommt man alte Versionen?
Vorraussetzung: Repo liegt bei Github

Man sucht sich das Datum wann es die Version gab die man haben will
im Fall von Youtv:
[Release] Youtv
Hab ich geschrieben werds am Freitag rausnehmen. Der Freitag war dann der 11 November, als
Brauch man was vor diesem Datum

Nun geht man auf die Github Seite des Repos
github.com/kodinerds/repo/

Da gibt es eine Knopf "[ANZAHL] Commits"

Den drückt man

Jetzt sieht man wer was geändert hat.....

Unten kann man auf „Older“ drücken

Dann wird in der Browser Adressleiste, eine URL wie folgt angezeigt:

github.com/kodinerds/repo/comm…dddc6fe1e863fd3e97c154+34

Wenn es viele Commits gibt, wird man beim Rückwerts springen Wahnsinnig.
Deshalb einfach eine große Zahl statt der 34 Ganz hinten in der URL eingeben

Schauen ob es passt: je älter= größere Nummer, je Neuer=kleinere Nummer

Das macht man solange bis man die richtige Zahl hat in dem Beispiel: 42461

github.com/kodinerds/repo/comm…c6fe1e863fd3e97c154+42461

Der Oberste ist 9.November also liegt es vor dem Datum
Dann die Überschrift anklicken in dem Fall "Autoupdate"

Jetzt Zeigt er die Änderungen (grün/rot) an (Inhalte)
Das ist egal, wir brauchen Links oben den Knopf "Browse Files"

Nun sieht man das Repo von damals

Bein unserem Beispiel
github.com/kodinerds/repo/tree…5671053c6e92c2b241a9ee99f

Nun findet man das addon plugin.video.youtv geht in das Verzeichnis

github.com/kodinerds/repo/tree…9ee99f/plugin.video.youtv

Dort findet man das zipfile, das klickt man an:

github.com/kodinerds/repo/blob…in.video.youtv-0.4.27.zip

Dann auf Download klicken und man hat das Addon ....

14. Was ist eine .md Datei und wie kann man diese bearbeiten

Eine Datei mit der Endung *.md*, bedeutet, dass es sich um eine *MarkDown* Datei handelt

Dieses Dateiformat ist das übliche für Github

Diese Datei kann entweder direkt auf Github erzeugt und bearbeitet werden, oder mit einem anderen Programm, wie zum Beispiel mit dem Online Editor „Stackedit“: https://stackedit.io/

Dieser Editor ist echt ein super Programm. Links wird da der Text geschrieben und rechts sieht man gleich das Ergebnis
Die Datei kann dann als MarkDown oder HTML Exportiert werden
Für PDF muss man bezahlen
Mit http://html2pdf.com/ kann aber eine HTML Datei in PDF Convertiert werden, jedoch nicht immer komplett Fehlerfrei

Dateien, welche mit Stackedit oder anderen Programmen erstellt und auf Github hochgeladen werden, sollten immer auf Richtigkeit (Formatierung, Absätze usw.) überprüft werden
Github ist sehr empfindlich wenn es um MarkDown Dateien geht, welche nicht direkt auf Github erstellt und bearbeite wurden

Die Formatierungen in MarkDown Dateien sind anders als in Word und Co.
So zum Beispiel wir in MarkDown ein Wort Fett geschrieben, wenn man es wie folgt schreibt:
fett
Es gibt unzählige dieser Schreibvarianten, dazu einfach die Google Suche benützen

15. Was ist eine addons.xml.md5 Datei und wie kann man diese bearbeiten

Eine *addons.xml.md5* Datei wird auch als Checksumme/Prüfsumme bezeichnet

Eine *addons.xml.md5* Datei hat üblicherweise 33 Stellen und besteht aus einer Buchstaben-Zahlenkombination

Dabei ist der Bereich der Buchstaben von **a-f** und bei Zahlen von **0-9** und sieht z.B. so aus:
4607ca7e6e26cb91b5e98c334ca0e252

Diese Datei befindet sich in jedem Repository

Wenn Entwickler Änderungen am Repo oder anderen Github Dateien (Addons) machen, und dann eine addon.xml ändern, so ist auch eine Anpassung der *addons.xml.md5* Datei notwendig

Die Aufgabe dieser Datei ist es, Kodi zu sagen, dass es ein Update (Aktualisierung) gibt
Kodi prüft im Hintergrund beim Start, installierte Repositorys auf Updates

Eigentlich sollte die Checksumme/Prüfsumme berechnet werden, z.B. mit Note++:
1. Öffne Note++
2. Werkzeuge – MD5- Erstelle aus Dateien
3. Wähle Dateien zur Erstellung des MD5-Hashwerts
   Suche nun die addon.xml des gewünschten Addons
4. Im unteren Feld wird nun die Checksumme/Prüfsumme angezeigt

In der Praxis genügt es jedoch, die Datei zu erstellen (33Stellen) und zu verändern

16. Github Konto als "Organistaion" betreiben

Es ist möglich, ein Github Konto in eine Organisation umzuwandeln

Dazu muss man wissen, dass dieser Vorgang NICHT rückgängig gemacht werden kann.
Es erscheint aber vor der Umwandlung noch einmal eine Warnung

Der Vorteil einer Github Organisation ist, dass jedem User eigene Rechte zugeteilt werden können

https://help.github.com/en/articles/repository-permission-levels-for-an-organization

So können auch User Repos erstellen, löschen, neue User einladen usw, dies wäre z.B. mit dem größten Recht Admin möglich

Nur der Eigentümer kann die Organisation löschen

Ebenso können Chats geführt werden und die Teams können in eigene Teams erstellen um an Projekten zu arbeiten

Für die Sicherheit der Organisation kann eine 2 Faktor Authentifizierung eingerichtet werden

Hier wird der Unterschied zwischen User Konto und Organisation erklärt:

https://help.github.com/en/articles/differences-between-user-and-organization-accounts

Um eine Organisation zu erstellen werden 2 Github Konten benötigt:

Ein Konto welches dann in eine Organisation umgewandelt wird und ein zweites Konto für den Eigentümer

Bei dem Konto welches in eine Organisation umgewandelt wurde, kann man sich dann nicht mehr mit Benutzername und Passwort anmelden!!

Der große Vorteil ist einfach, dass User eben unterschiedliche Rechte bekommen können und nicht immer auf den Repo Eigentümer warten müssen

Ein Nachteil kann sein, wenn die 2 Faktor Authentifizierung nicht verwendet wird, dass User mit Schreibrechten ganze Repos löschen können.
Dies könnte zum Problem werden, wenn z.B. ein Passwort geklaut oder Konto gehakt wurde

Aber Grundsätzlich ist es eine gute Möglichkeit für Teamworks

17. Wie kann eine Github Änderung einfach rückgängig gemacht werden

Die einfache Möglichkeit Änderungen in einer Branche rückgängig zu machen heiß "revert"

Es gibt verschiedene Möglichkeiten etwas Rückgängig machen zu wollen, z.B. ein *Commit* in einer Branche oder einen *Pull Request* usw.

Pull Request rückgängig machen:

https://help.github.com/en/articles/reverting-a-pull-request

Wird von einer eigenen Branche, in eine andere eigene Branche ein „*Merge Commit*" gemacht und wird diese Änderung rückgängig gemacht, so entsteht eine neue zusätzliche Branche mit dem Vorwort *revert* gefolgt vom Branchename (revert-Branchename)

Direkt auf Github kann ein *Merge Commit* wie folgt rückgängig gemacht werden:

Dazu das Github Project öffnen
Auf PullRequests klicken
Closed klicken
Suchen was rückgängig gemacht werden soll und anklicken
Nun befindet man sich in der Rubrik *"Conversation"*
Bis zum Ende nach unten scrollen

Hier ist der *Revert* Knopf zu finden. Diesen klicken und die Änderung rückgängig machen

Soll das Ganze über Github Desktop rückgängig gemacht werden so stehen hier die Infos dazu:

https://help.github.com/en/desktop/contributing-to-projects/reverting-a-commit

18. Github Desktop (für Windows) verwenden

Das Programm kann von hier herunter geladen werden: https://desktop.github.com/

Der Vorteil von diesem lokalen Programm ist, dass die Ansicht im Explorer erfolgt, übersichtlicher und besser zu Handhaben ist

Nach der Installation startet Github Desktop, das Programm ist in Englisch

Öffne File – Options – Accounts

Hier mit Eurem Github Konto anmelden

Im Anschluss fügt die gewünschten Repos in das Programm ein:
File – Clone Repository

Nun wird ein Fenster geöffnet, wo das Github zu sehen ist (Your repositories), hier das gewünschte Repo auswählen

Es erfolgt nun die Synchronisierung von Github auf den lokalen PC

Sobald dies beendet ist, ist das Project unter Current repository (links oben am Rand) zu finden

Gewünschtes Project auswählen

Im rechten Fenster kann nun unter “View the files in your repository in Explorer“ durch klick auf “Show in Explorer“, der lokale Project Ordner geöffnet werden

Jetzt können Änderungen gemacht werden und im Anschluss einfach den Explorer schließen

In Github Desktop werden nun im linken Fenster die Änderungen angezeigt

Um diese auf Github hochzuladen muss in dem Feld “Summary“ ein Text eingegeben werden (z.B. die Addon Versionsnummer).
Das Feld “Description“ (Beschreibung) ist optional und muss nicht ausgefüllt werden, kann aber gemacht werden

Im Anschluss “Commit to master“ klicken

Das Project wird nun für das Hochladen auf Github vorbereitet.

Am oberen Rand wird nun anstelle von “Fetch origin“, “PullRequest“ angezeigt

Klicken und der Upload Vorgang startet

Mit “Fetch origin“, kann bei jedem Project immer der aktuelle Github Stand lokal heruntergeladen werden

Hat ein Project mehr als eine Branche (z.B. –master, -nightly usw.), so kann diese am oberen Rand bei “Current branch“ ausgewählt werden

Hier ist es dann Wichtig, bevor Änderungen gemacht werden, die Taste “Fetch origin“ zu drücken
Somit werden dann aus der gewählten Branch die Daten heruntergeladen