

# Git

## ... eine Einführung

# Timeline

1. Einführung
2. Commits und Referenzen

# Was ist Version Control?

Bietet:

- ▶ Protokoll der Änderungen an den Daten
- ▶ Ermöglicht dadurch auf beliebige alte Versionen zuzugreifen
- ▶ Unterschiedliche Entwicklungszweige
- ▶ Geteilter Zugriff auf die Daten

# Umsetzung in Git

- ▶ Git ist dezentral, es benötigt *keinen* Git-Server, insbesondere keinen Host wie GitHub, GitLab, ...
- ▶ Die Protokollierung der Änderungen geschieht durch sog. **commits**
- ▶ **commits** sind identifizierbar über ihren Hash und haben *eindeutige* Eltern!
- ▶ Die Struktur ist ein „gerichteter azyklischer Graph“

# Logbeispiel

Feature: add wlc\_view\_get\_instance()  
Merge pull request #227 from kozec/xdg-positioner  
Updated example to use xdg-positioner and handle popup menus correctly. Ref Cloudef/wlc#210  
Fully implemented xdg-positioner and added getters for stored data. Fixes Cloudef/wlc#210  
Implemented skelet of xdg-positioner, responds to requests, but stores no data.  
Turned non-implemented xdg\_cb\_popup\_grab into warning  
Merge pull request #222 from LeonhardKoenig/prq.WM\_CLASS5  
Use WM\_CLASS5 class instead of instance  
Merge pull request #224 from Hummer12007/sysmacros  
Include sys/sysmacros.h for major/minor(3) (Linux)  
just use class\_total\_len as 2nd param  
If class property only holds one value, use it  
scratch  
Use WM\_CLASS5 class instead of instance  
Use WM\_CLASS5 class instead of instance  
Merge pull request #219 from SirCmpwn/redshift-fix  
Fix error in precondition  
Merge pull request #216 from SirCmpwn/redshift  
Add gamma control API  
Merge pull request #211 from myfreeweb/master  
Fix XWayland and security on FreeBSD  
CMake: s/WLPROTO/WAYLANDPROTOCOLS/  
CMake: Fix FindWaylandProtocol.cmake  
README: wayland-protocols 1.7(+) is now required  
README: Update remark about wayland-protocols  
CMake: Move FindPackage(WaylandProtocols) to root  
v0.0.7 Bump version to 0.0.7  
surface: Fix bad use of wlc\_size\_max  
xdg-shell: Partially implement v6  
v0.0.6 Bump version to 0.0.6  
submodule: Update wayland-protocols  
Merge pull request #197 from zandrmartin/flip-pixel-geometry  
flip vertical coordinates in read\_pixels()  
Merge pull request #192 from kozec/x11-growing-fix  
Fix: Inform X11 windows about border size.  
output: Set backend surface only after creation  
cmake: Constant for LINUX doesn't exist  
Revert "Use a fallback dir, when XDG\_RUNTIME\_DIR is unset"  
keymap: Invalidate fd on release  
keymap: Zero out structure on release  
Merge pull request #182 from yohanesu75/freebsd  
Changes to enable building on FreeBSD. Need to confirm still builds and works as expected on Linux!  
Merge pull request #184 from Snirkimmington/mention-way-cooler  
Include way-cooler in README  
Merge pull request #176 from Hummer12007/xdg-env  
Use a fallback dir, when XDG\_RUNTIME\_DIR is unset  
v0.0.5 Bump version to 0.0.5

- ▶ Zeiger auf den aktuellen commit: **HEAD**
- ▶ Man alle Vorgängercommits die zum aktuellen Status der Daten führen erreichen!
- ▶ Andere Entwicklungszweige sind einfach weiterer solcher Zeiger, sog. **branches**.
- ▶ „master“ ist ein typischer Name der Hauptbranch *aber nicht vorgegeben!*

# Tags

Also für jedes Release / jeden Zustand den man „festhalten“ möchte eine neue Branch? – Nein, es gibt **tags**.

- ▶ Referenz auf einen bestimmten **commit**
- ▶ Hat einen Namen (bspw. Versionsnummer v3.14)
- ▶ Es gibt „lightweight“ Tags und „annotated“ Tags

## Tags II

### Lightweight

- ▶ Besteht rein aus Referenz und Name
- ▶ Keinerlei weitere Informationen wie Name des Autors und Zeitpunkt

### Annotated

- ▶ Speichert zusätzlich Metainformationen, bspw. auch Beschreibung
- ▶ Sinnvoll für größere Releases, bspw. um relevante Änderungen zu listen

# Remotes: Arbeiten mit Online-Repositories

Bisher: Entwickler arbeitet alleine auf seinem Rechner, Code bleibt lokal.

Möchten: Code Teilen: **remotes**



# Remotes: Arbeiten mit Online-Repositories

**Bisher:** Entwickler arbeitet alleine auf seinem Rechner, Code bleibt lokal.

**Möchten:** Code Teilen: **remotes**

Remote:

- ▶ Benötigt Server (bspw. `git.imp.fu-berlin.de`, GitHub, GitLab)
- ▶ Es können mehrere remotes eingetragen werden
- ▶ **pull/push**: Änderungen holen/veröffentlichen

# Divergente Entwicklung: Mergen

Szenario:

1. Mehrere Entwickler ändern Datei **A** in ihren lokalen Repos **I1** und **I2**

# Divergente Entwicklung: Mergen

Szenario:

1. Mehrere Entwickler ändern Datei **A** in ihren lokalen Repos **I1** und **I2**
2. Entwickler 1 veröffentlicht Änderung auf gemeinsamer Remote **r**

# Divergente Entwicklung: Mergen

Szenario:

1. Mehrere Entwickler ändern Datei **A** in ihren lokalen Repos **I1** und **I2**
2. Entwickler 1 veröffentlicht Änderung auf gemeinsamer Remote **r**
3. Entwickler 2 versucht ebenfalls zu **pushen**: Schlägt fehl, da „Geschichte“ divergiert ist / seine lokale Version nicht mehr „aktuell“ ist

## Divergente Entwicklung: Mergen

Szenario:

1. Mehrere Entwickler ändern Datei **A** in ihren lokalen Repos **I1** und **I2**
2. Entwickler 1 veröffentlicht Änderung auf gemeinsamer Remote **r**
3. Entwickler 2 versucht ebenfalls zu **pushen**: Schlägt fehl, da „Geschichte“ divergiert ist / seine lokale Version nicht mehr „aktuell“ ist
4. Muss vorher **pullen**: Die Änderungen auf dem Server werden in die lokale Kopie integriert

## Divergente Entwicklung: Mergen

Szenario:

1. Mehrere Entwickler ändern Datei **A** in ihren lokalen Repos **I1** und **I2**
  2. Entwickler 1 veröffentlicht Änderung auf gemeinsamer Remote **r**
  3. Entwickler 2 versucht ebenfalls zu **pushen**: Schlägt fehl, da „Geschichte“ divergiert ist / seine lokale Version nicht mehr „aktuell“ ist
  4. Muss vorher **pullen**: Die Änderungen auf dem Server werden in die lokale Kopie integriert
- ! *WICHTIG*: Die Reihenfolge der Änderungen auf der Remote sind fest und dürfen nicht mehr geändert werden

# Divergente Entwicklung: Mergen

Szenario:

1. Mehrere Entwickler ändern Datei **A** in ihren lokalen Repos **I1** und **I2**
  2. Entwickler 1 veröffentlicht Änderung auf gemeinsamer Remote **r**
  3. Entwickler 2 versucht ebenfalls zu **pushen**: Schlägt fehl, da „Geschichte“ divergiert ist / seine lokale Version nicht mehr „aktuell“ ist
  4. Muss vorher **pullen**: Die Änderungen auf dem Server werden in die lokale Kopie integriert
    - ! **WICHTIG**: Die Reihenfolge der Änderungen auf der Remote sind fest und dürfen nicht mehr geändert werden
- ⇒ Lokale Änderungen werden (automatisch) angepasst und „angehängt“

# Mergeconflicts: Überschneidende Änderungen

Problem: Gleiche Datei wurde geändert.

Original:

```
1 the quick brown
2 fox jumps ovver
3 the lazy dog
4
```

Entwickler 1:

```
1 The quick brown
2 fox jumps ovver
3 the lazy dog
4
```

Entwickler 2:

```
1 the quick brown
2 fox jumps over
3 the lazy dog
4
```

► Textdatei & Änderung in verschiedenen Zeilen

⇒ Git kann idR. automatisch Änderungen zusammenführen



# Referenzen I

- ▶ Git project.  
Git Referenz  
<https://git-scm.com/docs>
- ▶ Scott Chacon, Ben Straub.  
Pro Git E-Book  
<https://git-scm.com/book/en/v2>
- ▶ Wikipedia Autoren.  
Artikel der englischen Wikipedia  
<https://en.wikipedia.org/wiki/Git>