TransRapport MVP - macOS Installation Guide

Diese Anleitung löst die bekannten macOS-spezifischen Installationsprobleme mit HDF5-Abhängigkeiten und PyTables.

🍎 macOS-spezifische Herausforderungen

TransRapport MVP verwendet PyTables für Datenverarbeitung, was HDF5-Header-Dateien benötigt. Auf macOS führt dies häufig zu Fehlern wie:
- `'H5public.h' file not found`
- `Could not find a local HDF5 installation`
- PyTables Installation schlägt fehl

Diese Anleitung bietet bewährte Lösungen für Intel- und Apple Silicon (M1/M2) Macs.

📋 Voraussetzungen

Hardware

- **macOS**: 10.14 oder neuer (empfohlen: macOS 12+)
- **RAM**: Mindestens 4GB (8GB empfohlen)
- **Speicherplatz**: 4GB frei verfügbar
- **Mikrofon**: USB-Mikrofon oder eingebautes Mikrofon

Software

- **Homebrew**: Paketmanager für macOS
- **Xcode Command Line Tools**: Für Kompilierung
- **Python**: 3.8 oder neuer

🚀 Schritt-für-Schritt Installation

Schritt 1: Homebrew installieren

Falls noch nicht installiert:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Nach der Installation:

```
# Für Apple Silicon (M1/M2)
echo 'eval "$(/opt/homebrew/bin/brew shellenv)"' >> ~/.zshrc
eval "$(/opt/homebrew/bin/brew shellenv)"

# Für Intel Macs
echo 'eval "$(/usr/local/bin/brew shellenv)"' >> ~/.zshrc
eval "$(/usr/local/bin/brew shellenv)"
```

Schritt 2: Xcode Command Line Tools installieren

```
xcode-select --install
```

Schritt 3: Python und Abhängigkeiten installieren

```
# Python installieren
brew install python

# HDF5 installieren (kritisch für PyTables)
brew install hdf5

# FFmpeg für Audio-Verarbeitung
brew install ffmpeg

# Weitere System-Abhängigkeiten
brew install pkg-config
```

Schritt 4: HDF5-Umgebungsvariablen setzen

Für Apple Silicon (M1/M2) Macs:

```
export HDF5_DIR=/opt/homebrew/opt/hdf5
export CPPFLAGS="-I/opt/homebrew/opt/hdf5/include"
export LDFLAGS="-L/opt/homebrew/opt/hdf5/lib"
```

Für Intel Macs:

```
export HDF5_DIR=/usr/local/opt/hdf5
export CPPFLAGS="-I/usr/local/opt/hdf5/include"
export LDFLAGS="-L/usr/local/opt/hdf5/lib"
```

Dauerhaft in der Shell speichern:

```
# Für Apple Silicon
echo 'export HDF5_DIR=/opt/homebrew/opt/hdf5' >> ~/.zshrc
echo 'export CPPFLAGS="-I/opt/homebrew/opt/hdf5/include"' >> ~/.zshrc
echo 'export LDFLAGS="-L/opt/homebrew/opt/hdf5/lib"' >> ~/.zshrc

# Für Intel
echo 'export HDF5_DIR=/usr/local/opt/hdf5' >> ~/.zshrc
echo 'export CPPFLAGS="-I/usr/local/opt/hdf5/include"' >> ~/.zshrc
echo 'export LDFLAGS="-L/usr/local/opt/hdf5/lib"' >> ~/.zshrc

# Shell neu laden
source ~/.zshrc
```

Schritt 5: TransRapport MVP herunterladen

```
# Zum gewünschten Verzeichnis navigieren
cd ~/Documents

# Repository klonen oder Archiv entpacken
# git clone <repository-url>
# oder Archiv entpacken nach ~/Documents/transrapport_mvp
```

Schritt 6: Python Virtual Environment erstellen

```
cd ~/Documents/transrapport_mvp

# Virtual Environment erstellen
python3 -m venv venv

# Aktivieren
source venv/bin/activate

# Pip aktualisieren
pip install --upgrade pip setuptools wheel
```

Schritt 7: Abhängigkeiten installieren

Option A: Automatische Installation (empfohlen)

```
# Zuerst die macOS-spezifische requirements.txt verwenden
pip install -r requirements_macos.txt

# PyTables separat installieren mit HDF5-Pfad
HDF5_DIR=$HDF5_DIR pip install tables==3.9.2
```

Option B: Manuelle Installation bei Problemen

```
# Basis-Abhängigkeiten installieren
pip install -r requirements_macos.txt

# PyTables mit expliziten Pfaden installieren
pip install --no-cache-dir --force-reinstall tables==3.9.2
```

Option C: Conda als Alternative (bei anhaltenden Problemen)

```
# Conda installieren falls nicht vorhanden
brew install --cask miniconda

# Conda-Umgebung erstellen
conda create -n transrapport python=3.11
conda activate transrapport

# HDF5 und PyTables über conda installieren
conda install hdf5 pytables

# Restliche Abhängigkeiten über pip
pip install -r requirements_macos.txt
```

Schritt 8: Installation testen

```
# Python-Imports testen
python -c "import tables; print('PyTables:', tables.__version__)"
python -c "import pyqtgraph; print('pyqtgraph:', pyqtgraph.__version__)"
python -c "import PyQt6; print('PyQt6 installiert')"

# Vollständiger Test
python -c "
import tables
import pyqtgraph
import PyQt6
import librosa
import faster_whisper
print('✅ Alle kritischen Abhängigkeiten erfolgreich importiert!')
"
```

Schritt 9: Anwendung starten

```
# Virtual Environment aktivieren (falls nicht aktiv)
source venv/bin/activate

# Anwendung starten
python main.py
```

🔧 Fehlerbehebung

Problem: “H5public.h file not found”

Lösung:

```
# HDF5 neu installieren
brew reinstall hdf5

# Umgebungsvariablen neu setzen
export HDF5_DIR=$(brew --prefix hdf5)
export CPPFLAGS="-I$(brew --prefix hdf5)/include"
export LDFLAGS="-L$(brew --prefix hdf5)/lib"

# PyTables neu installieren
pip uninstall tables
pip install --no-cache-dir tables
```

Problem: PyQt6/pyqtgraph Installation schlägt fehl

Lösung:

```
# Qt über Homebrew installieren
brew install qt6

# PATH für qmake setzen
export PATH="$(brew --prefix qt6)/bin:$PATH"

# PyQt6 neu installieren
pip uninstall PyQt6 PyQt6-Qt6 PyQt6-sip pyqtgraph
pip install PyQt6 pyqtgraph
```

Problem: Apple Silicon (M1/M2) Architektur-Konflikte

Lösung:

```
# Prüfen der HDF5-Architektur
file $(brew --prefix hdf5)/lib/libhdf5.dylib

# Sollte "arm64" zeigen, nicht "x86_64"
# Falls x86_64: Homebrew für ARM neu installieren

# Python-Architektur prüfen
python -c "import platform; print(platform.machine())"

# Sollte "arm64" zeigen
```

Problem: “tables” Import schlägt fehl

Lösung:

```
# Detaillierte Fehlermeldung anzeigen
python -c "import tables" -v

# Oft hilft Neuinstallation mit expliziten Pfaden
pip uninstall tables h5py
HDF5_DIR=$(brew --prefix hdf5) pip install --no-binary=h5py h5py
HDF5_DIR=$(brew --prefix hdf5) pip install --no-binary=tables tables
```

Problem: Performance-Probleme

Lösung:

```
# Prüfen ob native ARM-Binaries verwendet werden
python -c "
import numpy
import scipy
print('NumPy:', numpy.__version__)
print('SciPy:', scipy.__version__)
"

# Bei Bedarf neu installieren für ARM
pip uninstall numpy scipy
pip install --no-cache-dir numpy scipy
```

🎯 Optimierungen für macOS

Desktop-Verknüpfung erstellen

Erstellen Sie eine Datei `TransRapport.command`:

```
#!/bin/bash
cd ~/Documents/transrapport_mvp
source venv/bin/activate
python main.py
```

Ausführbar machen:

```
chmod +x TransRapport.command
```

Automatisches Aktivieren der Umgebung

Fügen Sie zu `~/.zshrc` hinzu:

```
# TransRapport MVP Umgebung
alias transrapport='cd ~/Documents/transrapport_mvp && source venv/bin/activate &&
python main.py'
```

Mikrofon-Berechtigungen

1. **Systemeinstellungen → Sicherheit & Datenschutz → Mikrofon**
2. Terminal und Python zu erlaubten Apps hinzufügen
3. Bei Bedarf Anwendung neu starten

📊 Systemanforderungen prüfen

```
# System-Info
system_profiler SPSoftwareDataType SPHardwareDataType

# Python und Abhängigkeiten
python --version
pip list | grep -E "(tables|pyqtgraph|PyQt6|h5py|numpy)"

# HDF5-Installation prüfen
brew list hdf5
h5dump --version 2>/dev/null || echo "h5dump nicht verfügbar"
```

🔄 Updates und Wartung

Abhängigkeiten aktualisieren

```
# Virtual Environment aktivieren
source venv/bin/activate

# Homebrew-Pakete aktualisieren
brew update && brew upgrade hdf5 python

# Python-Pakete aktualisieren
pip install --upgrade pip
pip install --upgrade -r requirements_macos.txt
```

Bei macOS-Updates

Nach macOS-Updates können Neukompilierungen nötig sein:

```
# Xcode Command Line Tools aktualisieren
sudo xcode-select --install

# Kritische Pakete neu installieren
pip uninstall tables pyqtgraph
pip install --no-cache-dir tables pyqtgraph
```

🆘 Support und Alternativen

Conda als vollständige Alternative

Falls pip-Installation weiterhin Probleme bereitet:

```
# Miniconda installieren
brew install --cask miniconda

# Neue Umgebung erstellen
conda create -n transrapport python=3.11
conda activate transrapport

# Alle Abhängigkeiten über conda
conda install pytables pyqt pyqtgraph numpy scipy librosa
pip install faster-whisper  # Falls nicht über conda verfügbar
```

Docker als letzte Option

Für komplexe Umgebungen:

```
# Docker Desktop für Mac installieren
# Dockerfile mit Ubuntu-Base verwenden
# Alle Linux-Installationsschritte ausführen
```

📝 Bekannte Limitationen

- **PyTables**: Erfordert HDF5-Header, kann bei Updates brechen
- **PyQt6**: Gelegentliche Wheel-Probleme auf neuen macOS-Versionen
- **Apple Silicon**: Einige Pakete haben noch keine nativen ARM-Builds
- **Xcode-Updates**: Können Neukompilierung erfordern

✅ Erfolgreiche Installation verifizieren

Nach erfolgreicher Installation sollten folgende Tests funktionieren:

```
# Basis-Test
python -c "
import tables
import pyqtgraph as pg
import PyQt6
import librosa
import faster_whisper
import numpy as np
print('✅ Alle Abhängigkeiten erfolgreich geladen!')
"

# GUI-Test
python -c "
from PyQt6.QtWidgets import QApplication
import pyqtgraph as pg
app = QApplication([])
win = pg.GraphicsLayoutWidget()
win.show()
print('✅ GUI-Test erfolgreich!')
app.quit()
"
```

Bei anhaltenden Problemen: Dokumentieren Sie die genaue Fehlermeldung, macOS-Version, und Hardware (Intel/Apple Silicon) für gezielten Support.

Version: 1.0
Letzte Aktualisierung: September 2025
Getestet auf: macOS 12-14, Intel & Apple Silicon