

# NVM - Node Version Manager

---

nvm ist ein Versionsmanager für node.js. Über nvm können mehrere unterschiedliche Node Versionen installiert werden. NVM wird über die Konsole aufgerufen und ist mit verschiedenen Shells kompatibel (sh, dash, ksh, zsh, bash).

## Links

- <https://github.com/coreybutler/nvm-windows> (Win)
- <https://github.com/nvm-sh/nvm> (Mac OS)

## Installation Windows

1. Lade den NVM-Installer über folgenden Link herunter:  
<https://github.com/coreybutler/nvm-windows/releases>
2. Installiere NVM über den [Installer](#)
3. Sollte vorher bereits NodeJS global ohne NVM installiert worden sein, wird empfohlen die vorherige NodeJS Version zu [deinstallieren](#).

## Installation Mac OS

- Installiere NVM über die Konsole mit:

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.35.2/install.sh
```

- Füge den Pfad zum NVM Ordner zur Shell-Config hinzu. ( `~/.bash_profile` , `~/.zshrc` , `~/.profile` , or `~/.bashrc` ).

```
export NVM_DIR="$([ -z "${XDG_CONFIG_HOME-}" ] && printf %s "${HOME}/.nvm" [ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
```

- Aktuelle Informationen zur Installation findet man unter folgendem Link:

<https://github.com/nvm-sh/nvm>

## Erste Schritte

Sobald NVM erfolgreich installiert wurde kann man über nvm in der Konsole eine neue Node Version mit folgendem Befehl installieren:

```
nvm install [NODE VERSIONSNUMMER]
```

Hinweis: Die aktuellste Versionsnummer von NodeJS findet man am einfachsten auf der [NodeJS-Seite](#)

Installierte Node Versionen auflisten

```
nvm list
```

Node Version verwenden/wechseln

```
nvm use [NODE VERSIONSNUMMER]
```

Node Version als Standard einrichten

```
nvm alias default [NODE VERSIONSNUMMER]
```

## Node.js

---

### Definition

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

## Neues Projekt erstellen

---

- Über die Konsole in den Projektordner gehen.

Projektordner wechsel über "change directory" (cd)

```
cd [projektpfad]
```

- Eigene Package.json Datei generieren (nur einmalig bei einem neuen Projekt notwendig.)

```
npm init
```

- Module installieren, die nur für die Entwicklung benötigt werden. (devDependencies)

Befehl	Zweck
<code>npm install Modul@Version</code>	installiere Modul in der Version Version (semantic versioning)
<code>npm install Modul@Version --save</code> <code>npm install Modul@Version -S</code>	Modul wird in der Datei <code>package.json</code> unter dependencies hinzugefügt (Mittlerweile default)
<code>npm install Modul@Version --save-dev</code> <code>npm install Modul@Version -D</code>	Modul wird in der Datei <code>package.json</code> unter devdependencies hinzugefügt
<code>npm install Modul@Version --global</code> <code>npm install Modul@Version -g</code>	Modul wird global installiert. (Von jedem Ordner aus anwendbar)
<code>npm uninstall Modul@Version</code>	entfernt Modul aus <code>node_modules</code>
<code>npm uninstall Modul@Version --save</code>	entfernt Modul aus <code>node_modules</code> und aus der dependencies- Eigenschaft in der <code>package.json</code> .
<code>npm update Modul</code>	Aktualisiert Modul auf die neueste Version
<code>npm update Modul --save</code>	Aktualisiert Modul auf die neueste Version und auch die Versionsnummer der dependencies- Eigenschaft in der <code>package.json</code> .
<code>npm init</code>	erstellt eine package.json-Datei (der Befehl fragt die Eigenschaften ab)
<code>npm init --yes</code>	erstellt eine package.json-Datei (der Befehl setzt Default-Werte ein)
<code>npm list</code>	listet alle installierten Module

	auf.
<code>npm search</code>	Suchbegriff sucht Module, deren Namen den Suchbegriff enthält

# Globale Module

## SASS

---

### Definition

Sass is the most mature, stable, and powerful professional grade CSS extension language in the world.

Sass ist eine CSS-Preprocessor Sprache.

### Liste von verwendeten Modulen

---