

NVM & Node.js Cheat Sheet

1. NVM Installation & Updates

Installation (Linux/macOS)

```
# Installation via curl
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash

# ODER via wget
wget -qO- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash

# Nach Installation: Terminal neu starten oder:
source ~/.bashrc      # für Bash
source ~/.zshrc       # für Zsh
```

Installation (Windows)

```
# nvm-windows verwenden (separates Projekt)
# Download: https://github.com/coreybutler/nvm-windows/releases

# Nach Installation via Installer:
nvm version
```

NVM aktualisieren

```
# Aktuelle Version prüfen
nvm --version

# Update durch Neuinstallation (neueste Version in URL einsetzen)
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash
```

```
# Globale Pakete nach Update migrieren  
nvm reinstall-packages <alte-version>
```

2. Node.js Versionen verwalten

Verfügbare Versionen anzeigen

```
# Alle installierbaren Versionen (sehr lange Liste)  
nvm ls-remote  
  
# Nur LTS-Versionen anzeigen  
nvm ls-remote --lts  
  
# Bestimmte Major-Version filtern  
nvm ls-remote 20          # alle Node 20.x.x Versionen  
nvm ls-remote --lts=iron  # LTS "Iron" (Node 20)  
nvm ls-remote --lts=hydrogen # LTS "Hydrogen" (Node 18)
```

Installierte Versionen anzeigen

```
# Alle lokalen Installationen  
nvm ls  
nvm list  
  
# Aktuelle aktive Version  
nvm current  
node --version
```

Node.js installieren

```
# Neueste Version installieren  
nvm install node
```

```
# Neueste LTS-Version installieren
nvm install --lts

# Spezifische Version installieren
nvm install 20.11.0          # exakte Version
nvm install 20              # neueste 20.x.x
nvm install 18.19.0

# Mit Migration globaler Pakete von anderer Version
nvm install 20 --reinstall-packages-from=18

# Neueste npm-Version gleich mit installieren
nvm install 20 --latest-npm
```

Node.js Version wechseln

```
# Zu bestimmter Version wechseln (nur aktuelle Shell)
nvm use 20.11.0
nvm use 20
nvm use --lts
nvm use node          # neueste installierte

# Standardversion festlegen (für neue Shells)
nvm alias default 20.11.0
nvm alias default lts/*
nvm alias default node

# Temporär andere Version für einen Befehl
nvm exec 18 node app.js
nvm exec 18 npm test
nvm run 18 app.js
```

Node.js deinstallieren

```
# Bestimmte Version entfernen
nvm uninstall 16.20.0
```

```
# Aktuelle Version kann nicht deinstalliert werden
# Erst wechseln, dann deinstallieren:
nvm use 20
nvm uninstall 18
```

Aliase verwalten

```
# Alias erstellen
nvm alias mein-projekt 18.19.0
nvm alias stable 20.11.0

# Alias verwenden
nvm use mein-projekt

# Alias entfernen
nvm unalias mein-projekt

# Alle Aliase anzeigen
nvm alias
```

3. NPM verwalten

NPM Version prüfen & aktualisieren

```
# Aktuelle Version anzeigen
npm --version
npm -v

# NPM auf neueste Version aktualisieren
npm install -g npm@latest

# Bestimmte NPM-Version installieren
npm install -g npm@10.2.0

# NPM mit Node-Installation aktualisieren
```

```
nvm install-latest-npm
```

NPM Cache verwalten

```
# Cache prüfen  
npm cache verify  
  
# Cache komplett leeren  
npm cache clean --force  
  
# Cache-Pfad anzeigen  
npm config get cache
```

Globale Pakete verwalten

```
# Globale Pakete auflisten  
npm list -g --depth=0  
  
# Global installieren  
npm install -g <paket>  
  
# Global deinstallieren  
npm uninstall -g <paket>  
  
# Veraltete globale Pakete anzeigen  
npm outdated -g  
  
# Alle globalen Pakete aktualisieren  
npm update -g
```

Lokale Pakete verwalten

```
# Abhängigkeiten installieren (aus package.json)  
npm install  
npm i
```

```
# Paket als Dependency hinzufügen
npm install <paket>
npm install <paket>@version

# Paket als DevDependency hinzufügen
npm install <paket> --save-dev
npm install <paket> -D

# Paket entfernen
npm uninstall <paket>

# Veraltete Pakete anzeigen
npm outdated

# Alle Pakete aktualisieren (nach semver)
npm update

# Sicherheitslücken prüfen
npm audit

# Sicherheitslücken automatisch beheben
npm audit fix
npm audit fix --force          # auch breaking changes
```

package-lock.json

```
# Exakte Versionen aus lock-file installieren (CI/CD)
npm ci

# Lock-file neu generieren
rm package-lock.json
npm install
```

4. Projekt-Konfiguration

.nvmrc Datei (automatische Node-Version)

```
# .nvmrc im Projektroot erstellen
echo "20.11.0" > .nvmrc

# ODER
echo "lts/*" > .nvmrc

# ODER
node -v > .nvmrc

# Version aus .nvmrc verwenden
nvm use

nvm install # installiert falls nötig
```

Automatisches Wechseln (Shell-Konfiguration)

Für Bash (~/.bashrc):

```
# Am Ende der Datei hinzufügen:
cdnvm() {
    command cd "$@" || return $?
    nvm_path=$(nvm_find_up .nvmrc | tr -d '\n')
    if [[ ! $nvm_path = *[^[:space:]]* ]]; then
        declare default_version;
        default_version=$(nvm version default);
        if [[ $default_version == "N/A" ]]; then
            nvm alias default node;
            default_version=$(nvm version default);
        fi
        if [[ $(nvm current) != "$default_version" ]]; then
            nvm use default;
        fi
    elif [[ -s $nvm_path/.nvmrc && -r $nvm_path/.nvmrc ]]; then
        declare nvm_version
        nvm_version=$(<"$nvm_path"/.nvmrc)
        declare locally_resolved_nvm_version
        locally_resolved_nvm_version=$(nvm ls --no-colors "$nvm_version" | tail -1 | tr -d '\-
```

```

>*' | tr -d '[:space:]')
    if [[ "$locally_resolved_nvm_version" == "N/A" ]]; then
        nvm install "$nvm_version";
    elif [[ $(nvm current) != "$locally_resolved_nvm_version" ]]; then
        nvm use "$nvm_version";
    fi
fi
}
alias cd='cdnvm'
cdnvm "$PWD"

```

Für Zsh (~/.zshrc):

```

# Automatisches nvm use bei Verzeichniswechsel
autoload -U add-zsh-hook
load-nvmrc() {
    local nvmrc_path="$(nvm_find_nvmrc)"
    if [ -n "$nvmrc_path" ]; then
        local nvmrc_node_version=$(nvm version "$(cat "${nvmrc_path}")")
        if [ "$nvmrc_node_version" = "N/A" ]; then
            nvm install
        elif [ "$nvmrc_node_version" != "$(nvm version)" ]; then
            nvm use
        fi
    elif [ -n "$(PWD=$OLDPWD nvm_find_nvmrc)" ] && [ "$(nvm version)" != "$(nvm version
default)" ]; then
        echo "Reverting to nvm default version"
        nvm use default
    fi
}
add-zsh-hook chpwd load-nvmrc
load-nvmrc

```

engines in package.json

```

{
  "name": "mein-projekt",
  "engines": {
    "node": ">=18.0.0 <21.0.0",

```

```
"npm": ">=9.0.0"  
}  
}
```

5. LESS verwalten

Installation

```
# Global installieren (CLI überall verfügbar)  
npm install -g less  
  
# Lokal im Projekt (empfohlen)  
npm install less --save-dev
```

LESS CLI verwenden

```
# Basis-Kompilierung  
lessc styles.less styles.css  
  
# Mit Source Maps  
lessc styles.less styles.css --source-map  
  
# Minifiziert (mit clean-css Plugin)  
npm install -g less-plugin-clean-css  
lessc styles.less styles.css --clean-css  
  
# Bestimmte Pfade für @import  
lessc --include-path=node_modules/src/styles styles.less out.css  
  
# Variablen überschreiben  
lessc --modify-var="primary-color=blue" styles.less styles.css
```

LESS mit npm Scripts

```
{
  "scripts": {
    "less:compile": "lessc src/styles/main.less dist/css/styles.css",
    "less:watch": "nodemon --watch src/styles -e less --exec 'npm run less:compile'",
    "less:minify": "lessc src/styles/main.less dist/css/styles.min.css --clean-css"
  },
  "devDependencies": {
    "less": "^4.2.0",
    "less-plugin-clean-css": "^1.5.1",
    "nodemon": "^3.0.0"
  }
}
```

LESS mit Build-Tools

Mit Webpack:

```
npm install less less-loader css-loader style-loader --save-dev
```

```
// webpack.config.js
module.exports = {
  module: {
    rules: [
      {
        test: /\.less$/,
        use: [
          'style-loader',
          'css-loader',
          'less-loader'
        ]
      }
    ]
  }
};
```

Mit Gulp:

```
npm install gulp gulp-less gulp-clean-css gulp-sourcemaps --save-dev
```

```
// gulpfile.js
const gulp = require('gulp');
const less = require('gulp-less');
const cleanCSS = require('gulp-clean-css');
const sourcemaps = require('gulp-sourcemaps');

gulp.task('less', function() {
  return gulp.src('src/styles/**/*.less')
    .pipe(sourcemaps.init())
    .pipe(less())
    .pipe(cleanCSS())
    .pipe(sourcemaps.write('.'))
    .pipe(gulp.dest('dist/css'));
});

gulp.task('watch', function() {
  gulp.watch('src/styles/**/*.less', gulp.series('less'));
});
```

LESS Version aktualisieren

```
# Global
npm update -g less

# Lokal
npm update less

# Auf neueste Major-Version
npm install less@latest --save-dev
```

6. SCSS/SASS verwalten

Installation

```
# Dart Sass (empfohlen, offiziell)
npm install -g sass

# Lokal im Projekt
npm install sass --save-dev

# Alternative: node-sass (deprecated, aber noch verwendet)
npm install node-sass --save-dev
```

SASS CLI verwenden

```
# Basis-Kompilierung
sass src/styles.scss dist/styles.css

# Mit Source Maps (Standard: an)
sass src/styles.scss dist/styles.css --source-map

# Ohne Source Maps
sass src/styles.scss dist/styles.css --no-source-map

# Komprimiert/Minifiziert
sass src/styles.scss dist/styles.css --style=compressed

# Watch-Modus (einzelne Datei)
sass --watch src/styles.scss:dist/styles.css

# Watch-Modus (Verzeichnisse)
sass --watch src/styles:dist/css

# Mehrere Dateien/Verzeichnisse
sass src/main.scss:dist/main.css src/admin.scss:dist/admin.css

# Load-Path für @use und @import
sass --load-path=node_modules --load-path=src/styles src/main.scss dist/main.css
```

SASS mit npm Scripts

```
{
  "scripts": {
    "sass:compile": "sass src/scss:dist/css",
    "sass:watch": "sass --watch src/scss:dist/css",
    "sass:prod": "sass src/scss:dist/css --style=compressed --no-source-map",
    "sass:lint": "stylelint 'src/scss/**/*.scss'"
  },
  "devDependencies": {
    "sass": "^1.69.0",
    "stylelint": "^15.11.0",
    "stylelint-config-standard-scss": "^11.0.0"
  }
}
```

SASS mit Build-Tools

Mit Webpack:

```
npm install sass sass-loader css-loader style-loader --save-dev
```

```
// webpack.config.js
module.exports = {
  module: {
    rules: [
      {
        test: /\.s[ac]ss$/,
        use: [
          'style-loader',
          'css-loader',
          {
            loader: 'sass-loader',
            options: {
              // Dart Sass verwenden
              implementation: require('sass'),
              sassOptions: {
                outputStyle: 'compressed',
              },
            },
          },
        ],
      },
    ],
  },
}
```

```
    ],  
  },  
  ],  
},  
};
```

Mit Vite (modern, empfohlen):

```
npm install sass --save-dev
```

```
// vite.config.js  
import { defineConfig } from 'vite';  
  
export default defineConfig({  
  css: {  
    preprocessorOptions: {  
      scss: {  
        additionalData: `@use "@/styles/variables" as *;`  
      }  
    }  
  }  
});
```

Mit Gulp:

```
npm install gulp gulp-sass sass gulp-clean-css gulp-sourcemaps --save-dev
```

```
// gulpfile.js  
const gulp = require('gulp');  
const sass = require('gulp-sass')(require('sass'));  
const cleanCSS = require('gulp-clean-css');  
const sourcemaps = require('gulp-sourcemaps');  
  
gulp.task('sass', function() {  
  return gulp.src('src/scss/**/*.scss')  
    .pipe(sourcemaps.init())  
    .pipe(sass().on('error', sass.logError))  
    .pipe(cleanCSS())  
    .pipe(sourcemaps.write('.'))  
    .pipe(gulp.dest('dist/css'));
```

```
});

gulp.task('watch', function() {
  gulp.watch('src/scss/**/*.scss', gulp.series('sass'));
});
```

SASS Linting (Stylelint)

Installation

```
npm install stylelint stylelint-config-standard-scss --save-dev
```

```
// .stylelintrc.json
{
  "extends": "stylelint-config-standard-scss",
  "rules": {
    "selector-class-pattern": null,
    "scss/dollar-variable-pattern": null
  }
}
```

Ausführen

```
npm run stylelint "src/scss/**/*.scss"
npm run stylelint "src/scss/**/*.scss" --fix
```

SASS Version aktualisieren

```
# Global
npm update -g sass

# Lokal
npm update sass

# Migration von node-sass zu dart-sass
npm uninstall node-sass
npm install sass --save-dev

# Code-Anpassungen: @import → @use/@forward
```

7. Nützliche Kombinationen

Komplettes Projekt-Setup

```
# 1. Node-Version festlegen
nvm install 20 --lts
nvm use 20
echo "20" > .nvmrc

# 2. Projekt initialisieren
npm init -y

# 3. SCSS einrichten
npm install sass --save-dev

# 4. package.json Scripts
```

```
{
  "scripts": {
    "dev": "npm run sass:watch",
    "build": "npm run sass:prod",
    "sass:compile": "sass src/scss:dist/css",
    "sass:watch": "sass --watch src/scss:dist/css",
    "sass:prod": "sass src/scss:dist/css --style=compressed --no-source-map"
  }
}
```

Globale Pakete bei Node-Update migrieren

```
# Aktuelle globale Pakete auflisten (zum Dokumentieren)
npm list -g --depth=0 > global-packages.txt

# Neue Node-Version mit Paket-Migration
nvm install 22 --reinstall-packages-from=20

# Oder manuell nach Update
```

```
nvm install 22
nvm use 22
npm install -g less sass typescript nodemon
```

Troubleshooting

```
# NVM nicht gefunden nach Installation
source ~/.bashrc # oder ~/.zshrc

# Berechtigungsprobleme
# NVM installiert in $HOME, sollte keine sudo-Rechte brauchen

# node/npm nicht gefunden
nvm use default
# oder
nvm alias default node

# Paket-Konflikte
rm -rf node_modules package-lock.json
npm cache clean --force
npm install

# SASS Kompilierungsfehler nach Update
npm rebuild sass

# node-gyp Fehler (bei nativen Modulen)
npm install -g node-gyp
node-gyp rebuild
```

Schnellreferenz

| Aufgabe | Befehl |
|-------------------|-----------------------------|
| NVM Version | <code>nvm --version</code> |
| Node installieren | <code>nvm install 20</code> |
| Node wechseln | <code>nvm use 20</code> |

| Aufgabe | Befehl |
|-------------------|--|
| Default setzen | <code>nvm alias default 20</code> |
| NPM aktualisieren | <code>npm install -g npm@latest</code> |
| LESS installieren | <code>npm install -g less</code> |
| SASS installieren | <code>npm install -g sass</code> |
| LESS kompilieren | <code>lessc input.less output.css</code> |
| SASS kompilieren | <code>sass input.scss output.css</code> |
| SASS watch | <code>sass --watch src:dist</code> |

Erstellt: Mai 2026 | NVM v0.39.x | Node.js 18-22 | SASS 1.x | LESS 4.x

Revision #2

Created 2026-05-31 14:06:54 UTC by art10m

Updated 2026-05-31 14:07:43 UTC by art10m