nvm Node version manager

CONCEPTS

nvm is a shell function, not a program: it only exists in shells that source nvm.sh, and it changes PATH of that shell only.

.nvmrc version in project
↓ `nvm use`
nvm shell function
↓ sets PATH
~/.nvm/versions installed Node
↓ first on PATH
node, npm this shell only

Global npm packages belong to one Node version; a new version starts without them.

INSTALL NVM

`curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.8/install.sh | bash`
Install nvm with curl

`nvm` **`--version`**
Show the nvm version

`command -v nvm`
Check that nvm is loaded

`source ~/.bashrc`
Reload the shell profile after install

INSTALL AND USE

`nvm` **`install`** *`<version>`*
Install a Node version

`nvm` **`install --lts`**
Install the newest LTS

`nvm` **`install node`**
Install the latest release

`nvm` **`install`**
Install the version from .nvmrc

`nvm` **`use`** *`<version>`*
Use a version in this shell

`nvm` **`use --lts`**
Use the newest LTS

`nvm` **`use node`**
Use the latest release

`nvm` **`use system`**
Use the system-installed Node

`nvm` **`use default`**
Return to the default version

INSTALL AND USE (cont.)

`nvm` **`use`**
Use the version from .nvmrc

`nvm` **`install --lts=`***`<name>`*
Install a named LTS line

`nvm` **`install`** *`<version>`* **`--latest-npm`**
Upgrade npm to the newest it supports

`nvm` **`install`** *`<version>`* **`--reinstall-packages-from=`***`<old>`*
Install and copy global packages

INSPECT VERSIONS

`nvm` **`ls`**
List installed versions

`nvm` **`ls-remote`**
List all installable versions

`nvm` **`ls-remote --lts`**
List LTS versions only

`nvm` **`current`**
Show the active version

`nvm` **`which`** *`<version>`*
Print the path of that node binary

`nvm` **`uninstall`** *`<version>`*
Remove an installed version

ALIASES AND DEFAULTS

`nvm` **`alias default`** *`<version>`*
Set the version new shells start with

`nvm` **`alias default node`**
Default to whatever is latest

`nvm` **`alias default lts/*`**
Default to the newest LTS

`nvm` **`alias`** *`<name>`* *`<version>`*
Create your own alias

`nvm` **`alias`**
List all aliases

RUN WITHOUT SWITCHING

`nvm` **`run`** *`<version>`* *`<script.js>`*
Run node under that version

`nvm` **`exec`** *`<version>`* *`<command>`*
Run any command under that version

RUN WITHOUT SWITCHING (cont.)

`nvm` **`run --lts`** *`<script.js>`*
Run node under the newest LTS

nvm run and exec fall back to the active Node when no .nvmrc is found; always pass a version in scripts.

.NVMRC AND PACKAGES

`echo "lts/*" > .nvmrc`
Pin the project to the newest LTS

`node -v > .nvmrc`
Pin the project to the running version

`nvm` **`install && nvm use`**
Install then use the pinned version

`nvm` **`reinstall-packages`** *`<version>`*
Copy that version's global packages here

.nvmrc contents

Value	Meaning
`20`	Any installed or installable 20.x
`20.11.1`	That exact release
`lts/*`	Newest LTS
`lts/iron`	A named LTS line
`node`	Latest release

nvm looks for .nvmrc upward through parent directories and exits with status 127 if it finds none. Lines starting with # are comments.

SHELL INTEGRATION

`export NVM_DIR="$HOME/.nvm"`
Where nvm lives

`[ -s "$NVM_DIR/nvm.sh" ] && . "$NVM_DIR/nvm.sh"`
Load nvm in your shell profile

`[ -s "$NVM_DIR/nvm.sh" ] && . "$NVM_DIR/nvm.sh" --no-use`
Load nvm without activating the default

Slow shell startup is usually nvm.sh being sourced eagerly. Add --no-use, or load it lazily on first use of nvm, node or npm.