



Github repository:

<https://github.com/vuejs/vue>



Chen Xida

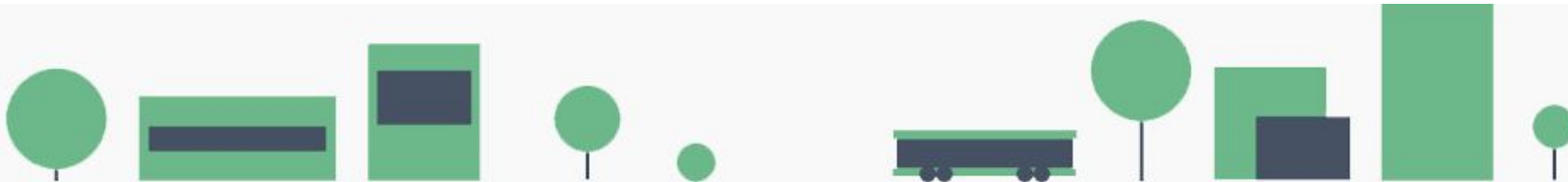
1217780

Stojkovic Danilo

1222399

# Breve descrizione

- Vue.js è un framework per lo sviluppo di applicazioni web
- Si basa sull'architettura Model-View-ViewModel
- Nasce dall'idea di offrire un ambiente più leggero di quello offerto da angularJS
- Prima versione: febbraio 2014
- Versione attuale: 3.2.24





# ISSUE TRACKING SYSTEM:



# Apertura issues

Per aprire una issue si deve usare:

<https://new-issue.vuejs.org/?repo=vuejs/core#>

I am opening an issue for

vuejs/core (Vue 3)

This is a

Bug Report

Feature Request

Please make sure to file the issue at appropriate repo.

Issue title

Version

3.2.36

Check if the issue is reproducible with the latest stable version of Vue.

Link to minimal reproduction

If possible, use [the SFC Playground](#) to provide a reproduction. If it requires an actual build setup, you can use [StackBlitz](#) or provide a GitHub repo.

[What is a \*minimal reproduction\*, and why is it required?](#)

Please do not just fill in a random link. We will close your issue if you do that.

# Apertura issues

Steps to reproduce

What do we need to do after opening your repro in order to make the bug happen? Clear and concise reproduction instructions are important for us to be able to triage your issue in a timely manner. Note that you can use [Markdown](#) to format lists and code.

What is expected?


What is actually happening?


Any additional comments? (optional)


e.g. some background/context of how you ran into this bug.

Preview

# Labels

 Labels

 Milestones

 Search all labels

33 labels		Sort ▼
1.x		3 open issues and pull requests
backlog		
browser quirks		19 open issues and pull requests
bug		42 open issues and pull requests
contribution welcome		12 open issues and pull requests
dependencies	Pull requests that update a dependency file	1 open issue or pull request
discussion		30 open issues and pull requests
feat:compiler		4 open issues and pull requests
feat:ssr		13 open issues and pull requests

# Milestones


0 Open ✓ 14 Closed		Sort ▼
2.0	<div><div></div></div> 100% complete 0 open 4 closed	
Closed on 3 Nov 2016 ⌚ Last updated over 5 years ago		
1.0.0-alpha	<div><div></div></div> 100% complete 0 open 17 closed	
Closed on 6 Nov 2015 ⌚ Last updated over 6 years ago		
1.0.0-beta	<div><div></div></div> 0% complete 0 open 0 closed	
Closed on 6 Nov 2015 ⌚ Last updated over 6 years ago		
1.0.0	<div><div></div></div> 100% complete 0 open 1 closed	
Closed on 6 Nov 2015 ⌚ Last updated over 6 years ago		
0.11	<div><div></div></div> 100% complete 0 open 7 closed	
Closed on 26 Aug 2015 ⌚ Last updated almost 7 years ago		
v0.9.0	<div><div></div></div> 100% complete 0 open 2 closed	
Closed on 25 Feb 2014 ⌚ Last updated over 8 years ago		


# Board


2.7   
Updated on 17 Mar

 Filter cards


7 To do

 Investigate attribute setting strategy: set as DOM property by default and fallback to attributes  
Added by yyx990803


 feat(props): allow defining a required prop as null  
2 of 13 tasks  
#9358 opened by posva  
discussion feature request improvement  
semver:minor

 feat: add origin prop for <transition-  
#8424

1 In progress

 chore: Improve npm package specification  
4 of 13 tasks  
#12501 opened by annwb

1 Done

 feat(types): expose ElementAttributesProperty for TSX  
4 of 13 tasks  
#9379 opened by andoshin11  
semver:minor



# Reportistica

Nei report vengono specificati i seguenti campi:

- Breve descrizione
- Tipo di cambiamento (Bugfix, Feature)
- La pull request crea rotture nel progetto? (se si fornire ulteriori informazioni)
- La pull request soddisfa i seguenti requisiti (spuntare le opzioni soddisfatte es: (è richiesto sul ramo di dev v2.x, passa tutti i test, etc...))
- Se si aggiunge una nuova feature, motivare in modo convincente
- Altre informazioni

# Gestione rilasci

Le release sono gestite direttamente dagli sponsor principali del progetto.

Le versioni sono contrassegnate dai tags ed inoltre le modifiche sono descritte dettagliatamente in:

[vue/blob/mainvue/blob/main/CHANGELOG.md](https://github.com/vuejs/vue/blob/main/CHANGELOG.md)

Fino ad ora sono state create 214 release

La latest version della repo è 2.6.14

Releases 214

 v2.6.14 **Latest**  
on 7 Jun 2021

# Gestione contribuzioni

Per contribuire al progetto si deve seguire la guida descritta in:  
<https://github.com/vuejs/vue/blob/main/.github/CONTRIBUTING.md>

ed è suddiviso in:

- Issue Reporting Guidelines

<https://new-issue.vuejs.org/?repo=vuejs/core#>

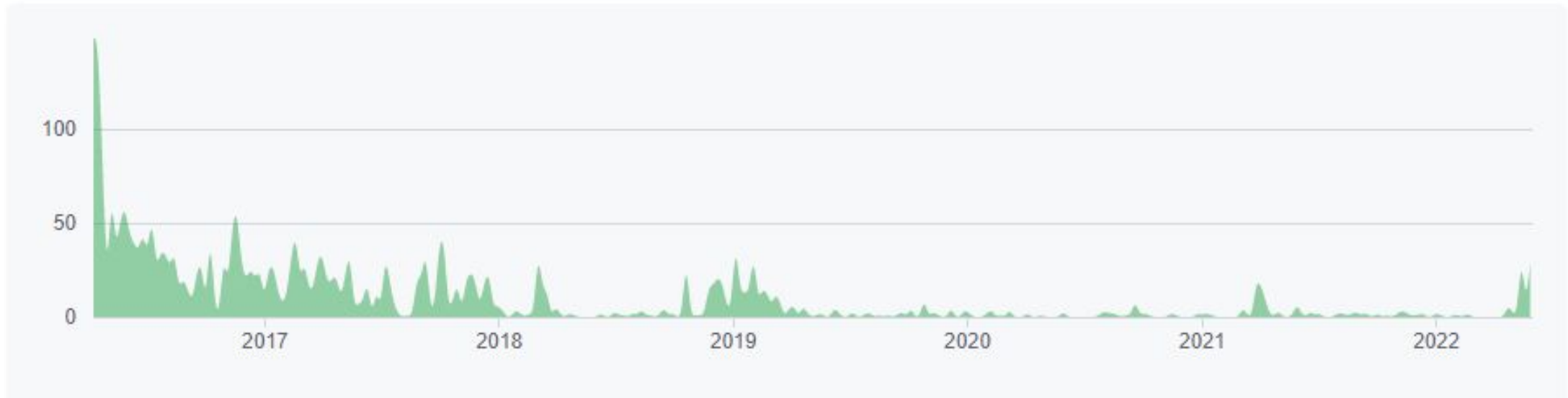
- Pull Request Guidelines (regole da seguire, ad es: fornire test case se si aggiungono feature)
- Development Setup (uso del prodotto)
- Project Structure (suddivisione file in un progetto Vue)

# Gestione contribuzioni

Apr 10, 2016 – Jun 2, 2022

Contributions: Commits ▼

Contributions to main, excluding merge commits and bot accounts





# VERSION CONTROL SYSTEM



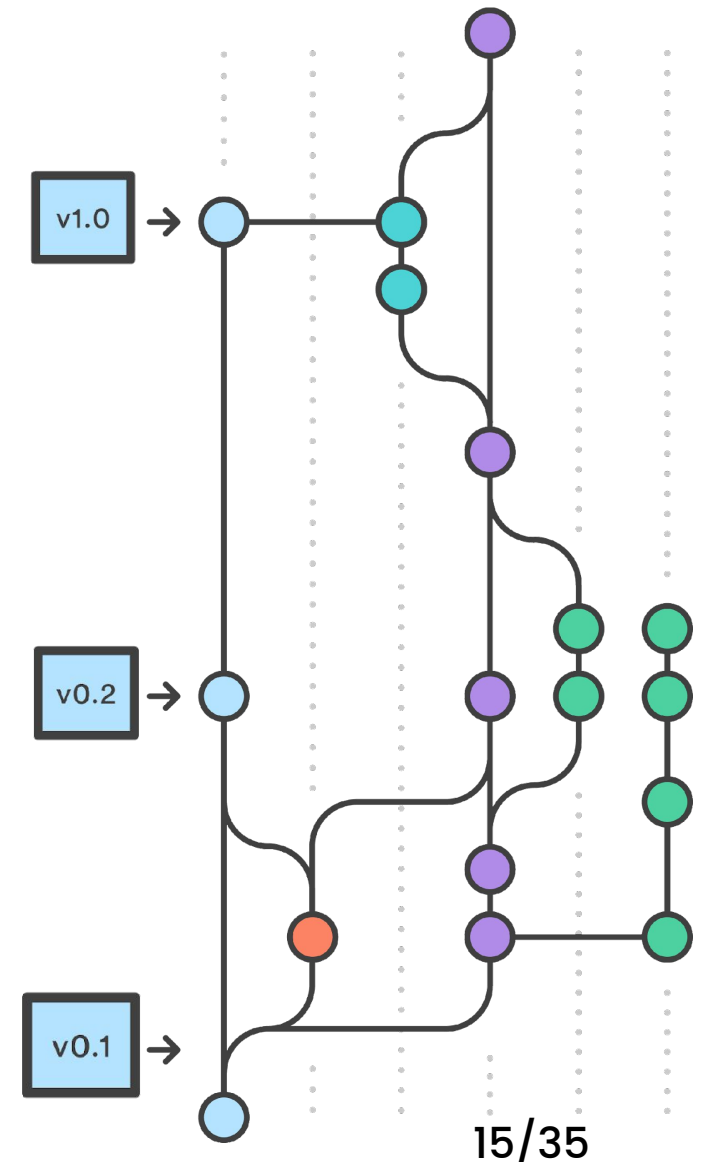
E' stato usato Git nel progetto di Vue e ha queste caratteristiche:

- Fornisce la possibilità di creare diversi branch (rami) a seconda delle necessità del progetto (main, features, develop, release, etc..)
- È un DVCS, quindi permette storicizzazione e condivisione in due momenti separati
- È collegato con IssueTrackingSystem di GitHub attraverso i messaggi scritti nel commit, ad esempio con «Close #numeroIssue» è possibile spostare le issue in stato di Done nella Board del progetto.

Link download: <https://git-scm.com/downloads>

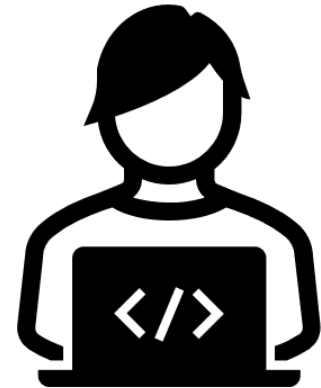
# Workflow

- Si suppone che un workflow basato su gitflow possa essere adatto a questo progetto, ossia riportare sul ramo di **develop** ogni qualvolta che una feature viene completata nel ramo di **feature**.
- Quindi una volta che si decide il rilascio della versione, si passa al ramo di **release** per modifiche minori ed infine rilasciare in produzione nel ramo **master**.
- Così come è descritto anche nelle linee guida del progetto è bene NON sottoscrivere PR al ramo master.
- È inoltre necessario superare i test in locale con «npm test» prima di fare la PR, dato che fallirebbe la build automatica di GitHub Actions.



# Steps per contribuire al progetto

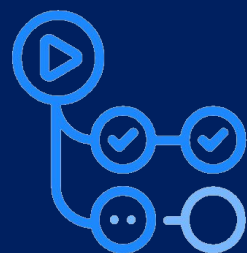
1. Leggere le linee guida del progetto
2. Aprire una issue ed aspettare la conferma
3. Forkare la repository
4. Effettuare le modifiche (bugfix o feature)
5. Superare i test localmente
6. Creare una pull request
7. Aspettare l'approvazione degli amministratori per il merge







# CONTINUOS INTEGRATION & BUILD AUTOMATION



Github Actions



# Continuous integration

Lo strumento utilizzato per la Continuous Integration è GitHub Actions: <https://docs.github.com/en/actions>

GitHub Actions è uno strumento che permette di configurare la pipeline di integration attraverso la configurazione di file.yml creando un workflow componibile in base all'esigenza dell'organizzazione.

In genere sono configurati per automatizzare build, test e report.

Inoltre la peculiarità di questo strumento è la disposizione di Actions dal marketplace, che permettono di importare configurazioni anche molto complesse.

Nel progetto sono presenti due workflow principali:

<https://github.com/vuejs/vue/actions/workflows/release-tag.yml>

<https://github.com/vuejs/vue/actions/workflows/ci.yml>

# Build automation

## Release-tag.yml

```
23 lines (21 sloc) | 601 Bytes
Raw Blame    

1  on:
2    push:
3      tags:
4        - 'v*' # Push events to matching v*, i.e. v1.0, v20.15.10
5
6  name: Create Release
7
8  jobs:
9    build:
10     name: Create Release
11     runs-on: ubuntu-latest
12     steps:
13       - name: Checkout code
14         uses: actions/checkout@master
15       - name: Create Release for Tag
16         id: release_tag
17         uses: yyx990803/release-tag@master
18     env:
19       GITHUB_TOKEN: ${ secrets.GITHUB_TOKEN }
20     with:
21       tag_name: ${ github.ref }
22       body: |
23         Please refer to [CHANGELOG.md](https://github.com/vuejs/vue/blob/main/CHANGELOG.md) for details.
```

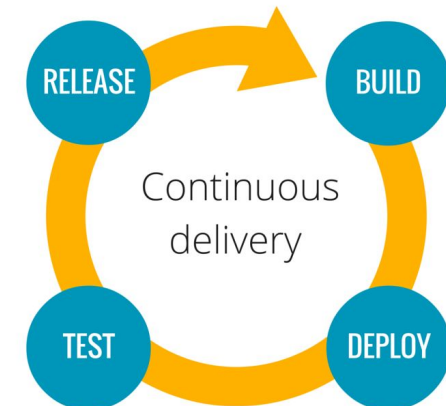
# Build automation

## ci.yml

```
1  name: 'ci'
2  on:
3    push:
4      branches:
5        - main
6    pull_request:
7      branches:
8        - main
9  jobs:
10   unit-test:
11     runs-on: ubuntu-latest
12     steps:
13       - uses: actions/checkout@v2
14
15       - name: Install pnpm
16         uses: pnpm/action-setup@v2
17
18       - name: Set node version to 16
19         uses: actions/setup-node@v2
20         with:
21           node-version: 16
22           cache: 'pnpm'
23
24       - run: pnpm install
25
26       - name: Run unit tests
27         run: pnpm run test:unit
```

# CI-Strumento utilizzato

- Il progetto utilizza la piattaforma di continuous integration CircleCI (<https://circleci.com/>).
- La pipeline CI di Vue.js si può trovare al link:
- <https://app.circleci.com/pipelines/github/vuejs/vue?branch=dev>.



# CI-Pipelines

- `weekly_regression_test`:  
utilizzato per eseguire i test di regressione settimanalmente;
- `install-and-parallel-test`:  
esecuzione dei test in parallelo.



vue 898	▶ <span>Success</span>	<code>weekly_regression_test</code>	dev 65a333f	24d ago	1m 48s	
vue 897	▶ <span>Success</span>	<code>install-and-parallel-test</code>	pull/12539 2396f10 Update README.md	25d ago	2m 37s	



# TESTING

# Testing

Per il testing, il progetto utilizza le librerie:

Jasmine, Karma e Vitest

<https://jasmine.github.io/>

<https://karma-runner.github.io/latest/index.html>

<https://vitest.dev/>

- Sono tutti contenuti nell'apposita cartella `/test`
- Vengono eseguiti dalla build automation nei momenti di push o pull request

```
test/  
├── e2e/  
├── helpers/  
├── transition/  
└── unit/
```





# Test statici

- Il test statici vengono eseguiti attraverso il comando **pnpm run ts-check**
- Le regole sono contenute nell'apposito file **tsconfig.json**

## package.json

```
"ts-check": "tsc -p tsconfig.json --noEmit",  
"ts-check:test": "tsc -p test/tsconfig.json --noEmit",
```

## tsconfig.json

```
"allowJs": true,  
"noImplicitAny": false,  
"noImplicitThis": false,  
  
"noUnusedLocals": true,  
"experimentalDecorators": true,  
"resolveJsonModule": true,  
"esModuleInterop": true,  
"removeComments": false,
```

# Test di unità

- Suddivisi nelle cartelle \features e \modules
- Seguono il pattern AAA (Arrange Act Assert)

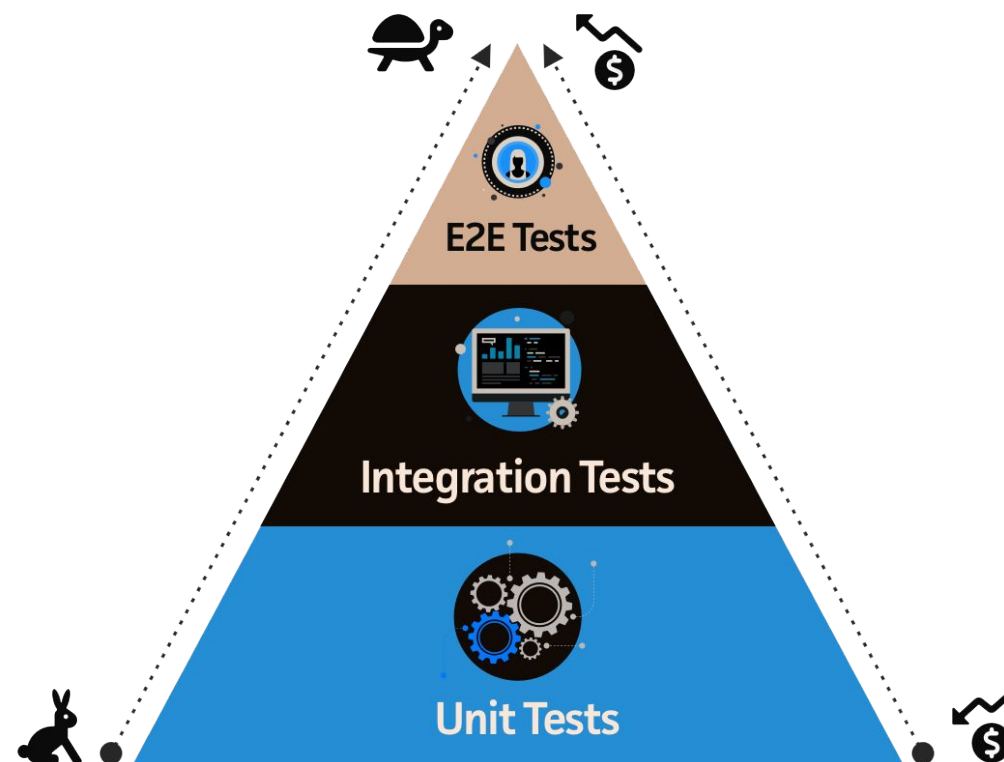
```
it('handle division', () => {  
  const vm = new Vue({  
    data: { a: 2 },  
    template: `<div>{{ 1/a / 4 | double }}</div>`,  
    filters: { double: v => v * 2 }  
  }).$mount()  
  expect(vm.$el.textContent).toBe(String(1 / 4))  
})
```



# Test End-to-End

I test e2e si trovano nell'omonima cartella test/e2e.

I test e2e cercano di simulare l'uso del prodotto vero e proprio. Nonostante il costo maggiore di questa tipologia di test sono anche quelli che forniscono il maggior valore aggiunto.



# Smoke Test

Si ipotizza che uno strumento adatto agli smoke test di questo progetto potrebbe essere la libreria Nightmare.js.

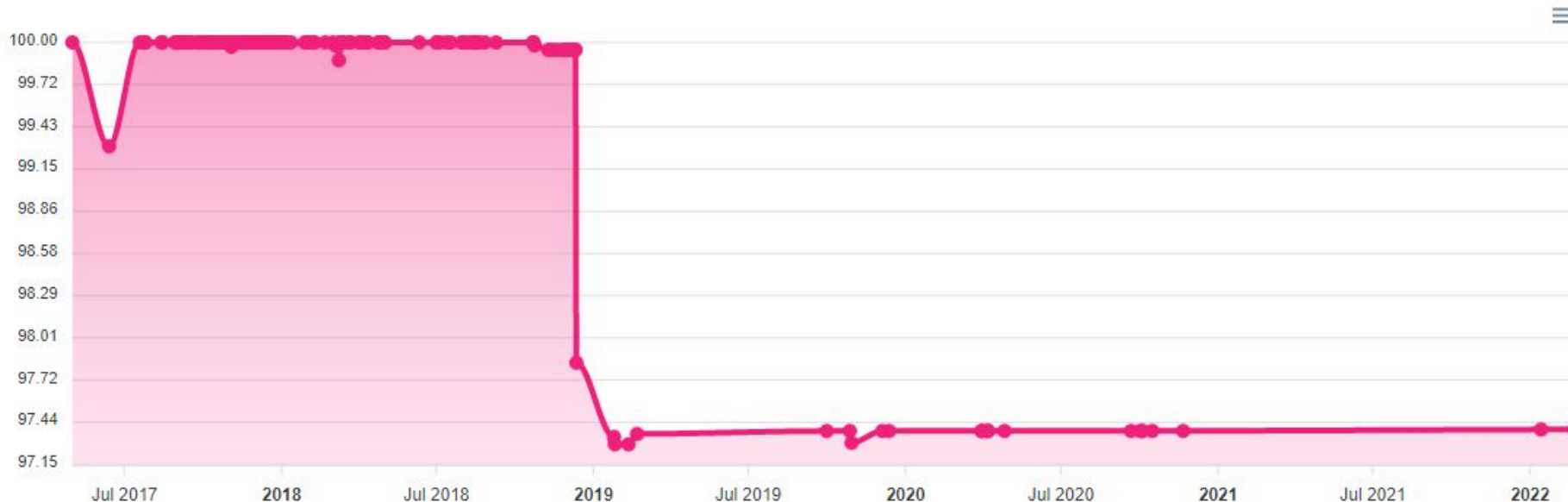
In questo modo si possono testare le funzionalità di base del progetto a costi ridotti, riducendo significativamente anche il tempo di build, seguendo la filosofia del Fail-Fast.

<https://github.com/segmentio/nightmare>



# Copertura

- Il progetto attualmente ha una copertura che si aggira sul 97%, nonostante ciò non si può affermare nè l'eshaustività di tali test, nè la loro qualità.
- Lo strumento utilizzato è Codecov:



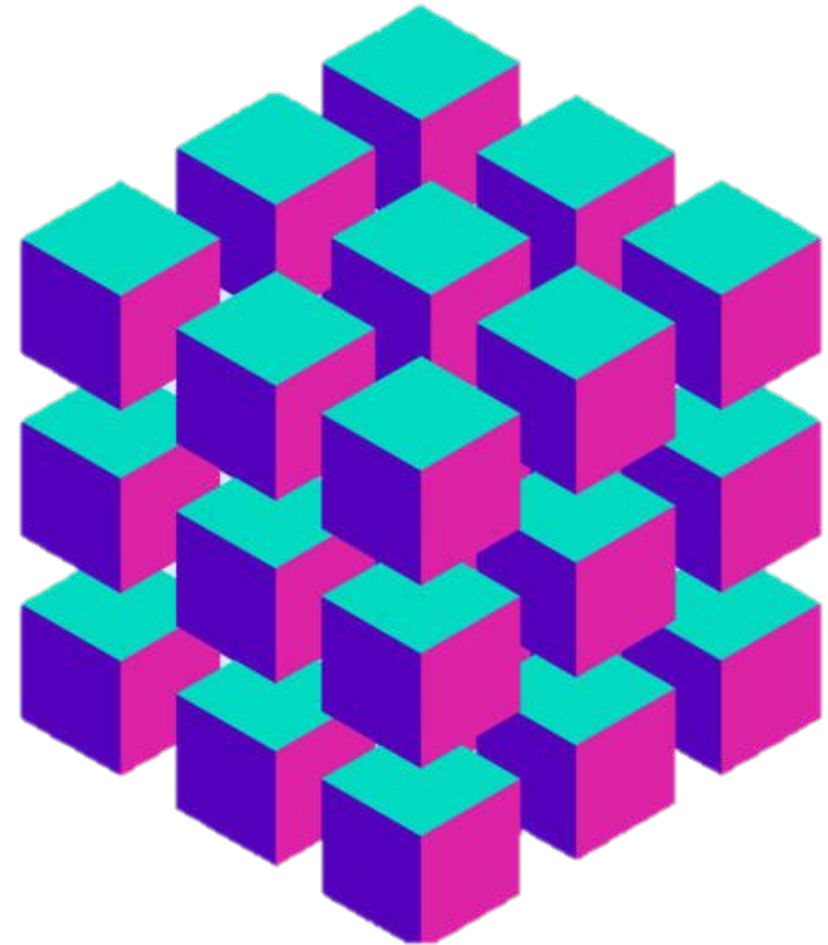


# ARTIFACT REPOSITORY & DEPLOYMENT

# AR-Strumento utilizzato

Il progetto utilizza il gestore di pacchetti e dipendenze npm.

[www.npmjs.com](https://www.npmjs.com)



# AR-Strumento utilizzato

“The free npm Registry has become the center of JavaScript code sharing, and with more than one million packages, the largest software registry in the world.”

npm publish (si deve possedere un account npm con email verificata)

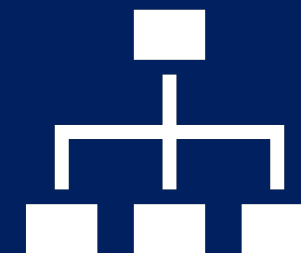
<https://docs.npmjs.com/updating-your-published-package-version-number>

npm install nomepacchetto.

Nel nostro caso:

npm install -g @vue/cli  
vue create nomeprogetto.





# CONFIGURATION MANAGEMENT TOOLS

# Configuration Management Tools

Non è stata trovata nessun orchestratore e file di configurazione particolari a causa della natura del progetto:

- Assenza di server da configurare.
- Il prodotto è una libreria (non un applicativo) che si poggia su Node e non su macchine reali.

Per questa ragione ad esempio non si usano strumenti come Docker (non si sfruttano i vantaggi dei container).

infatti la configurazione del progetto è stato gestito con la combinazione degli strumenti di Github + npm.

**Grazie per l'attenzione**