



Vue JS

Introducció a VueJS

Aina Palacios

Aina Palacios

- Enginyera de Telecomunicacions especialitzada en Audiovisuals
- Màster en Tecnologies Avançades especialitzada en deep learning en Multimèdia!
- Experiència en programació web i machine learning.
- Mentora a IT Academy de **Vuejs**
- Mentora a IT Academy de **DataScience**



<https://www.linkedin.com/in/ainapc/>



ainaPali#2617



Què és VueJS?

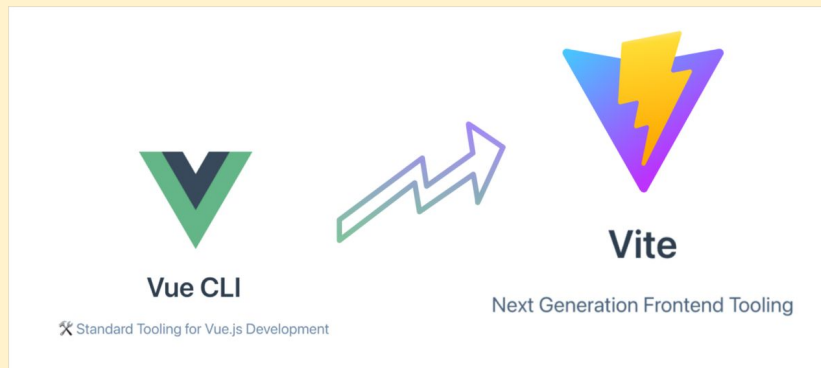


- Llibreria *open source* de **JavaScript** per desenvolupar UI (interfície d'usuari)
- En un Modelo-Vista-Controlador seria la **Vista**, necessita altres components per desenvolupar una pàgina web.
 - Angular està més orientat a controlador-vista
 - React també està orientat a l'usuari
- Basat en **components**
- S'utilitza en empreses com:
 - Nintendo
 - Adobe
 - Netflix
 - ...



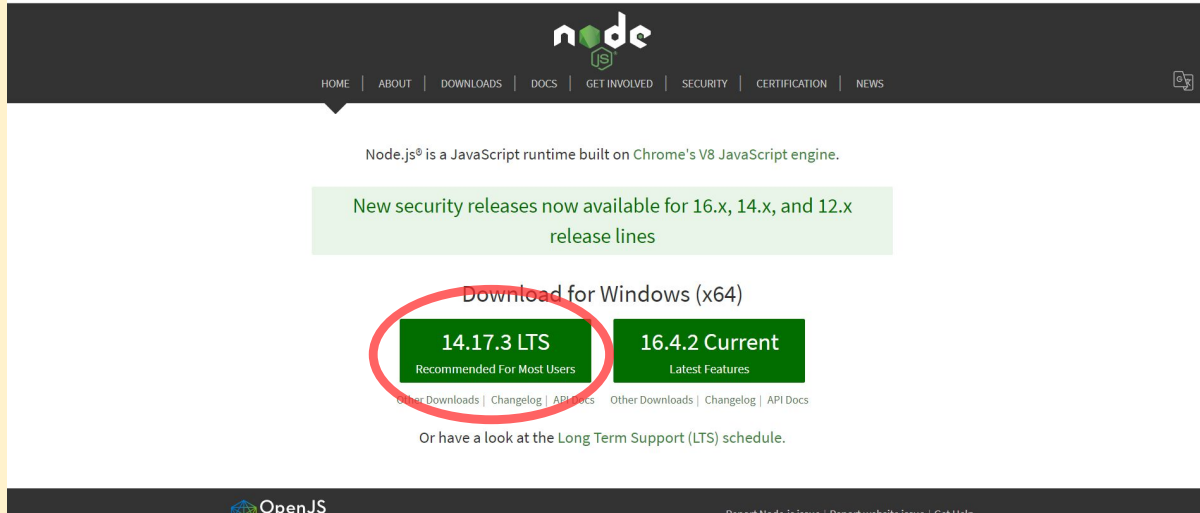
Ecosistema

- VueJS consta de diferents ecosistemes per desenvolupar tasques concretes
 - **Vue** -> Core o núcli de Vue, on hi ha les funcions principals
 - **VueCLI** -> Assistent per crear aplicacions VUE en un terminal o entorn
 - **Vue Router** -> Sistema per crear i gestionar rutes URL
 - **Vuex** -> Gestor d'estats -> Centralitzar els estats de la aplicació



Node JS

- Node.js is an open source, cross-platform, server environment that uses JavaScript on the server
- Install NodeJS: www.nodejs.org/es



Node JS

- S'ha instal·lat bé?
 - Obrir terminal o CMD, i escriure “**node -v**”, si apareix un número, es que tot està correcte. (NVM)

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

Microsoft Windows [Versión 10.0.19042.1052]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\usuario\Desktop\cursoReact\todos>
C:\Users\usuario\Desktop\cursoReact\todos>node -v
v10.15.3

C:\Users\usuario\Desktop\cursoReact\todos>|
```

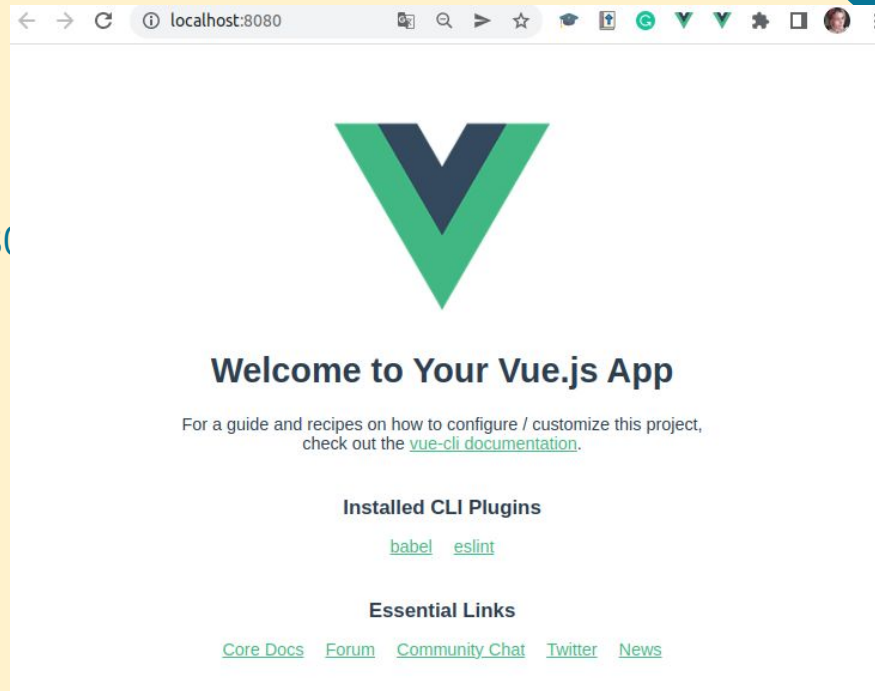
Creació del primer projecte VueJS!

1. Obrim Visual Studio Code i fem un nou Terminal
2. **`npm install -g @vue/cli`**
3. **`vue create yourProjectName`**



Obrir i iniciar el projecte

1. Al terminal: **cd yourProjectName**
 - a. La comanda *cd* ens porta dins del directori que acabem de crear
2. Al terminal: **npm run serve**
3. Obrim el Chrome i posem: localhost:8080



Ja ho tens!!

Primera web creada!



Comandandes bàsiques

- **npm run start**
 - Per iniciar el server en local, podem veure els canvis a localhost:3000
- **npm build**
 - Compila el projecte i crea els arxius necessaris per fer un “deploy” en producció. Aparaixarà tot a la carpeta “build”
- **npm test**
 - Fa tots els test
- **npm install / npm i**
 - Instal·la, reinstal·la o actualitza totes les dependències de package.json

GitHub



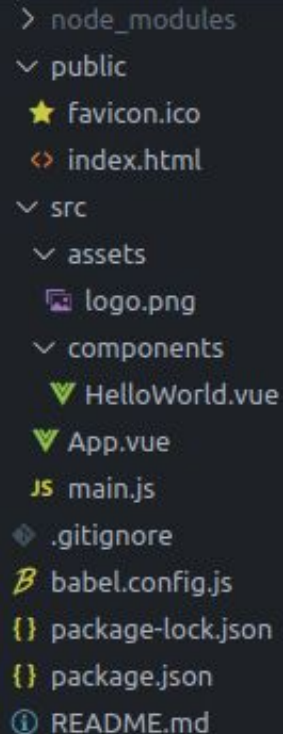
<https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

- **git clone NomRepositori** -> Clonar Repo
- **git status** -> Estat del git
- 1. **git add NomFitxer** -> Afegir
- 2. **git commit -m "Missatge"**
- 3. **git push**
- **git checkout nomBranca/nomCommit** -> anar a la branca o commit que vulguem

Errors: Si ens posa que no podem modificar la branca perquè tenim conflictes, podem fer **git stash** seguidament del git checkout que vulguem.

Més info: <https://learngitbranching.js.org/>

File structure (Estructura d'arxius)



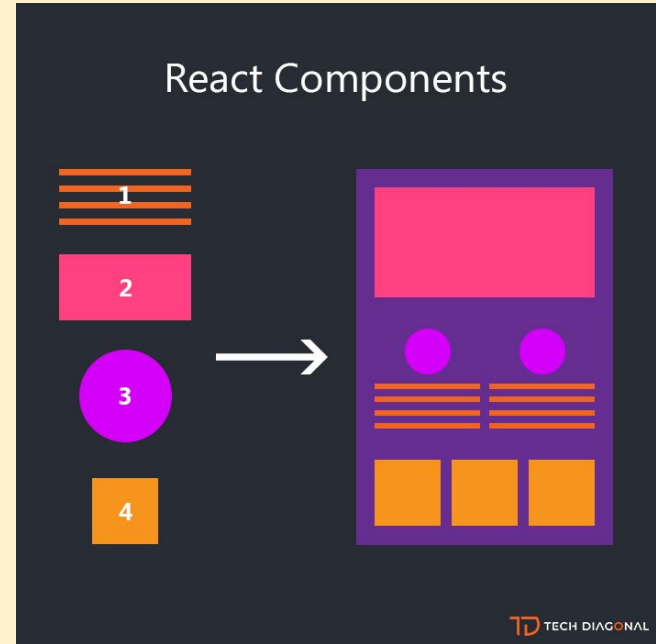
A screenshot of a file explorer interface with a dark background. It shows a hierarchical tree of files and folders. The 'public' folder is expanded, showing 'favicon.ico' and 'index.html'. The 'src' folder is also expanded, showing 'assets' (with 'logo.png'), 'components' (with 'HelloWorld.vue' and 'App.vue'), and 'main.js'. At the root level, there are files for '.gitignore', 'babel.config.js', 'package-lock.json', 'package.json', and 'README.md'.

- > node_modules
- ✓ public
 - ★ favicon.ico
 - <> index.html
- ✓ src
 - ✓ assets
 - 🖼 logo.png
 - ✓ components
 - 📄 HelloWorld.vue
 - 📄 App.vue
 - 📄 main.js
- 💎 .gitignore
- 📄 babel.config.js
- {} package-lock.json
- {} package.json
- 📄 README.md

- **main.js** és l'arxiu principal del projecte
- **App.js**
- **package.json**
- **node_modules**
- **.gitignore**

Components!

- Els Components permeten separar la UI (Interfície d'usuari) en peces independents reutilitzables i pensar en cada component de manera aïllada.
- Els trens grans frameworks de JS estàn pensats en components. (Vue, React i Angular).
- Nombrarem "Pare" a la view o pàgina que contingui els components, que seràn nombrats com a "Fills".
- Els components han de tenir la lògica implementada en ells mateixos, han de ser independents del Pare.



Single-File Components

```
App.vue
src > App.vue > {} "App.vue" > template
1 <template>
2   
3   <HelloWorld msg="Welcome to Your Vue.js App"/>
4 </template>
5
6 <script>
7   import HelloWorld from './components/HelloWorld.vue'
8
9   export default {
10     name: 'App',
11     components: {
12       HelloWorld
13     }
14   }
15 </script>
16
17 <style>
18 #app {
19   font-family: Avenir, Helvetica, Arial, sans-serif;
20   -webkit-font-smoothing: antialiased;
21   -moz-osx-font-smoothing: grayscale;
22   text-align: center;
23   color: #2c3e50;
24   margin-top: 60px;
25 }
26 </style>
27
```

- **Template** -> HTML
- **Script** -> import dels components
 - Export default -> Nom, data, methods...
- **Style**



Importar un CSS Framework

- Normalment, es sol importar un css Framework ja creat. Conté els seus propis estils, amb variables globals fàcilment editables. Els més comuns són:
 - Bootstrap / Bootstrap-vue
 - Tailwind
- També ens podem trobar amb una Template ja designada.



Importar Bootstrap

1. Al terminal podem posar: **npm i bootstrap**

```
JS main.js
import { createApp } from 'vue'
import App from './App.vue'
import 'bootstrap'
import 'bootstrap/dist/css/bootstrap.min.css'

createApp(App).mount('#app')
|
```

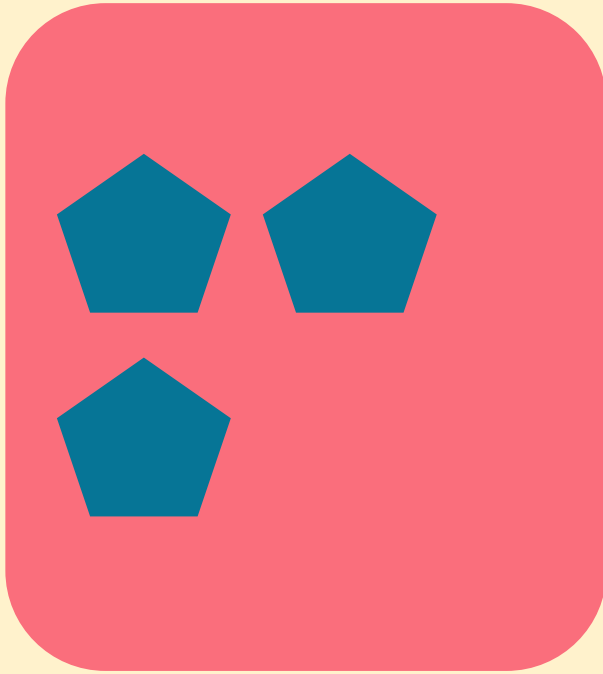
Com passar data entre components

Pare a fill	Fill a pare
Props	Passar callbackFunction com a prop



Props

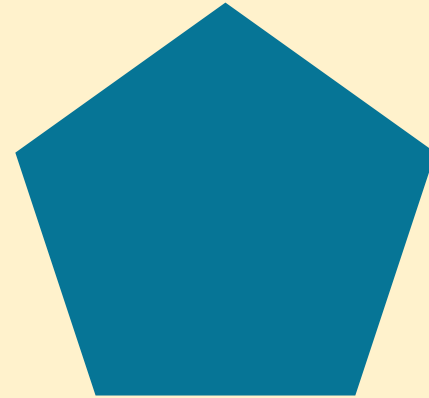
Pare



Props



Component



```

▼ Card.vue U      ▼ App.vue M X
src > ▼ App.vue > {} "App.vue" > template > div.container.row
1  <template>
2    
3    <div class="container row">
4      <Card/>
5    </div>
6  </template>
7
8  <script>
9    import Card from '@components/Card.vue'
10   export default {
11     name: 'App',
12     components: {
13       Card
14     }
15   }
16 </script>
17

```

```

▼ Card.vue U X
src > components > ▼ Card.vue > {} "Card.vue"
1  <template lang="">
2    <div class="card col-4">
3      <div class="card-header">Títol</div>
4      <div class="card-body">Body</div>
5    </div>
6  </template>
7  <script>
8    export default {
9      name: "Card"
10   }
11 </script>
12 <style lang="">
13
14 </style>

```

```
▼ Card.vue U    ▼ App.vue M X
src > ▼ App.vue > {} "App.vue" > style > #app
  1  <template>
  2    
  3    <div class="d-flex justify-content-center row">
  4      <Card :titolProp="titolPare" :bodyProp="bodyPare"/>
  5    </div>
  6  </template>
  7
  8  <script>
  9    import Card from '@components/Card.vue'
 10    export default {
 11      name: 'App',
 12      components: {
 13        Card
 14      },
 15      data() {
 16        return{
 17          titolPare: "Titol del pare",
 18          bodyPare: "Body del pare"
 19        }
 20      }
 21    }
 22  </script>
```

```
▼ Card.vue U ×  ▼ App.vue M
src > components > ▼ Card.vue > {} "Card.vue"
1  <template lang="">
2    <div class="card col-4">
3      <div class="card-header">{{titolProp}}</div>
4      <div class="card-body">{{bodyProp}}</div>
5    </div>
6  </template>
7  <script>
8    export default {
9      name: "Card",
10     props: ["titolProp", "bodyProp"]
11   }
12 </script>
13 <style lang="">
14
15 </style>
```



Titol del pare

Body del pare

Parlar de Fill a Pare: Callback Functions

```
▼ Card.vue U X  ▼ App.vue M
src > components > ▼ Card.vue > {} "Card.vue"
1  <template lang="">
2    <div class="card col-4">
3      <div class="card-header">{{titolProp}}</div>
4      <div class="card-body">
5        <p>{{bodyProp}}</p>
6        <button v-on:click="hanFetClick">
7          Click me!
8        </button>
9      </div>
10   </div>
11 </template>
12 <script>
13   export default {
14     name: "Card",
15     props: ["titolProp", "bodyProp"],
16     methods: {
17       hanFetClick() {
18         this.$emit('fillClick');
19       }
20     }
21   }
22 </script>
```

```
▼ Card.vue U    ▼ App.vue M X
src > ▼ App.vue > {} "App.vue"
1  <template>
2    
3    <div class="d-flex justify-content-center row">
4      <Card :titolProp="titolPare"
5        |   v-bind:bodyProp="bodyPare"
6        |   v-on:fillClick="contador++"
7      />
8    <h1>{{contador}}</h1>
9    </div>
10 </template>
11
12 <script>
13 import Card from '@components/Card.vue'
14 export default {
15   name: 'App',
16   components: {
17     Card
18   },
19   data() {
20     return{
21       titolPare: "Titol del pare",
22       bodyPare: "Body del pare",
23       contador: 0
24     }
25   }
26 }
27 </script>
```



Titol del pare

Body del pare

Click me!

5

Methods

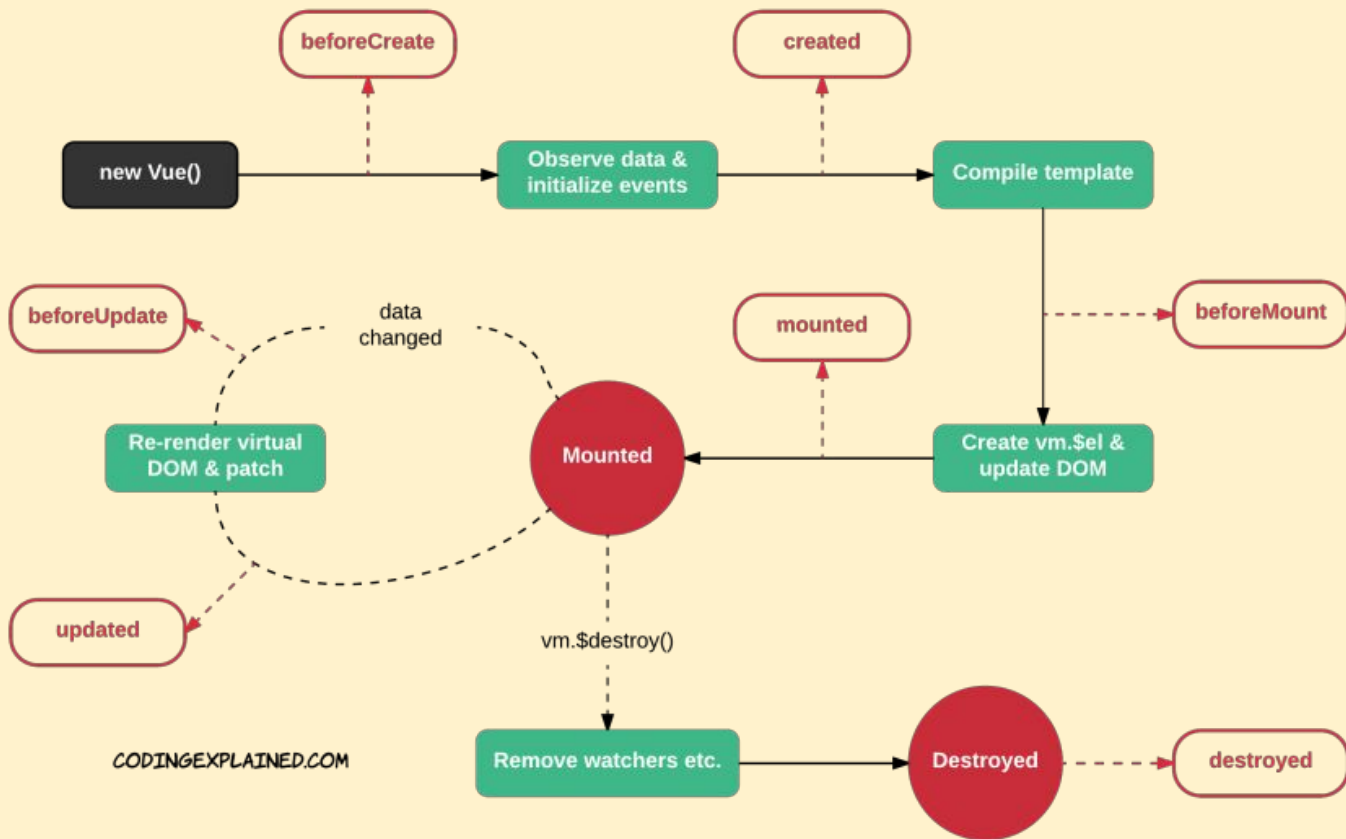
```
<template lang="">
  <div class="card col-4">
    <div class="card-header">{{titolProp}}</div>
    <div class="card-body">
      <p>{{bodyProp}}</p>
      <button v-on:click="hanFetClick">
        Click me!
      </button>
    </div>
  </div>
</template>
<script>
export default {
  name: "Card",
  props: ["titolProp", "bodyProp"],
  methods: {
    hanFetClick() {
      this.$emit('fillClick');
    }
  }
},
```

Watch vs Computed

```
▼ Card.vue U    ▼ App.vue M X
src > ▼ App.vue > {} "App.vue" > script > [🔍] default > 🔗 components > 🔗 Card
1  <template>
2    
3    <div v-if="show" class="d-flex justify-content-center row">
4      <Card :titolProp="titolPare"
5        |   v-bind:bodyProp="bodyPare"
6        |   v-on:fillClick="contador++"
7      />
8    <h1>{{frase}}</h1>
9    <h2>{{modificarFrase}}</h2>
10  </div>
11  <button @click="show= !show">Show!</button>
12
13 </template>
14
```

```
▼ Card.vue U    ▼ App.vue M X
src > ▼ App.vue > {} "App.vue" > script > [🔍] default > 🔗 computed > 🔗 modificarFrase
16 import Card from '@components/Card.vue'
17 export default {
18   name: 'App',
19   components: {
20     Card
21   },
22   data() {
23     return{
24       titolPare: "Titol del pare",
25       bodyPare: "Body del pare",
26       contador: 0,
27       show: false,
28       frase: "El contador és 0"
29     }
30   },
31   watch: {
32     contador(){
33       this.frase = "El contador en watch és " + this.contador;
34     }
35   },
36   computed: {
37     modificarFrase(){
38       return "El contador en computed és " + this.contador;
39     }
40   }
}
```

LiveCycle



LiveCycle

```
export default {  
  name: "Card",  
  props: ["titolProp", "bodyProp"],  
  methods: {  
    hanFetClick() {  
      this.$emit('fillClick');  
    }  
  },  
  created() {  
    console.log('created')  
  },  
  unmounted() {  
    console.log('unmounted')  
  }  
}  
</script>  
<style lang="">
```