



TECHDAY 

# *DESCUBRE* VUE



ORGANIZADO POR  
LA COMUNIDAD  
FRONTEND



**Héctor Gonzalo**  
**COMUNIDAD JS**



ORGANIZADO POR  
LA COMUNIDAD  
FRONTEND

# Cerrando el círculo - "State of the art"

# **1. Testing en Vue**

## **2. Nuxt**

## **3. Vite**

# 1. Testing en Vue

--

**Tipos de tests**

**Estándares en Vue**

# Tipos de tests

- Tests unitarios / componentes
- Test de integración
- Tests E2E (end to end) / funcionales
- Visual testing

# Tests unitarios

- Cada unidad elemental testeable del proyecto se somete a pruebas
- Objetivo: Aislar código, probarlo y determinar que funciona cómo se intencionó que lo hiciera
- Librería estándar en Vue:
  - Vue testing library

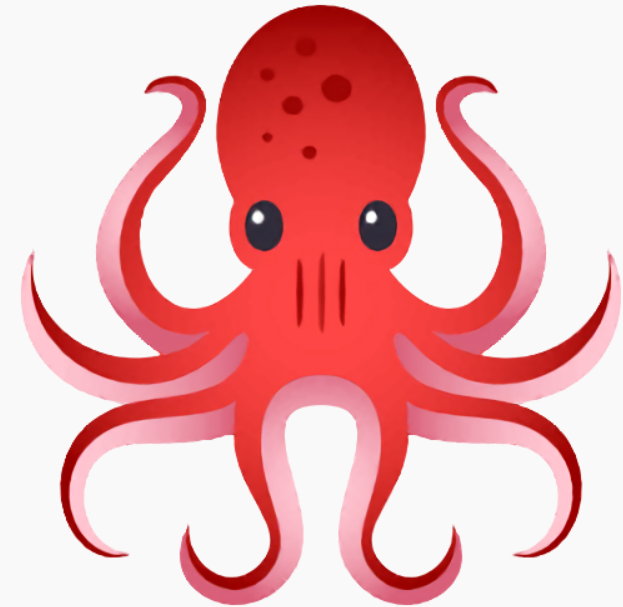
# Tests de integración

- Los módulos individuales se combinan y testean en grupo
- Objetivo: Evaluar si la aplicación cumple los requisitos funcionales
- Los test de integración toman de entrada módulos que han pasado test unitarios, los agrega y les ejecuta los test definidos para el agregado
- Librería estándar en Vue:
  - Vue testing library



# Vue Testing library

- Conjunto de librerías y frameworks bajo una capa de abstracción que expone una API y una filosofía
- Orientado a Vue
- Basado en DOM testing library, Vue Test Utils y Jest



# Tests E2E / Test funcionales

- La aplicación se testea entera desde el punto de vista de un usuario real
- Objetivo: Cobertura de partes que las pruebas unitarias y las pruebas de integración no cubren
- Se suele emplear un robot que "utiliza" la app y verifica que todo funciona correctamente
- Librería estándar en Vue:
  - Cypress

# Cypress

- Escribimos test que se ejecutan en una ventana del navegador
- Podemos analizar paso a paso
- Muy visual

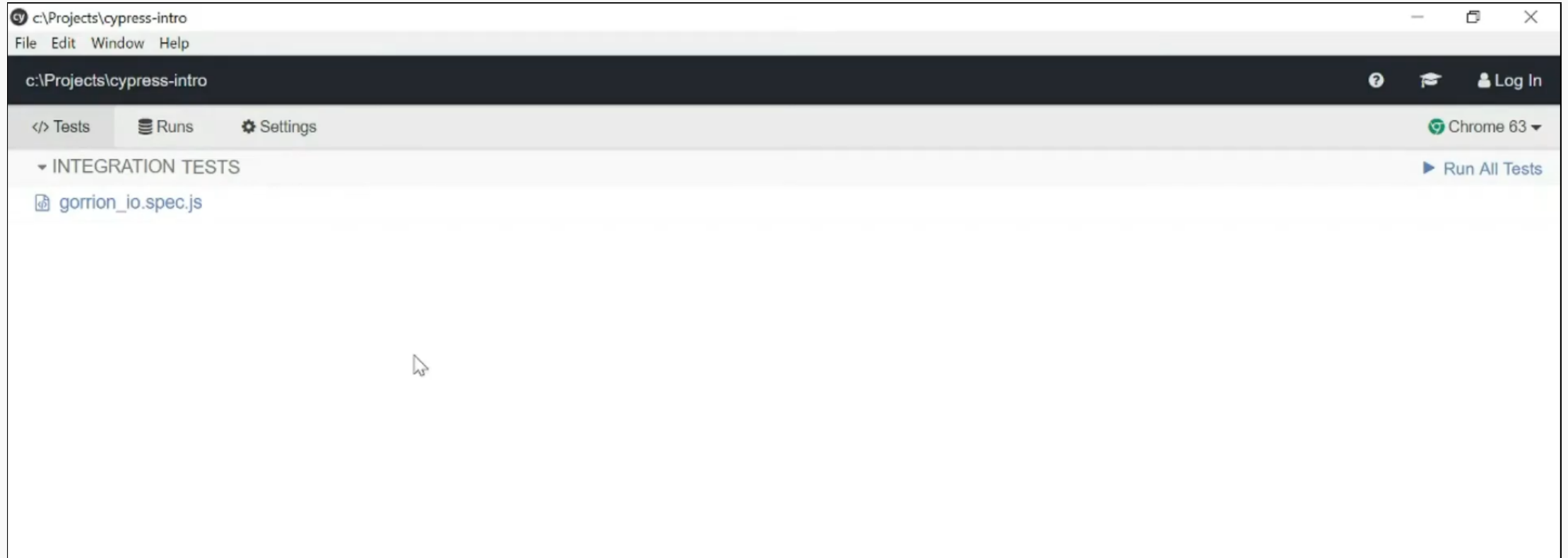


# Cypress - Ejemplo

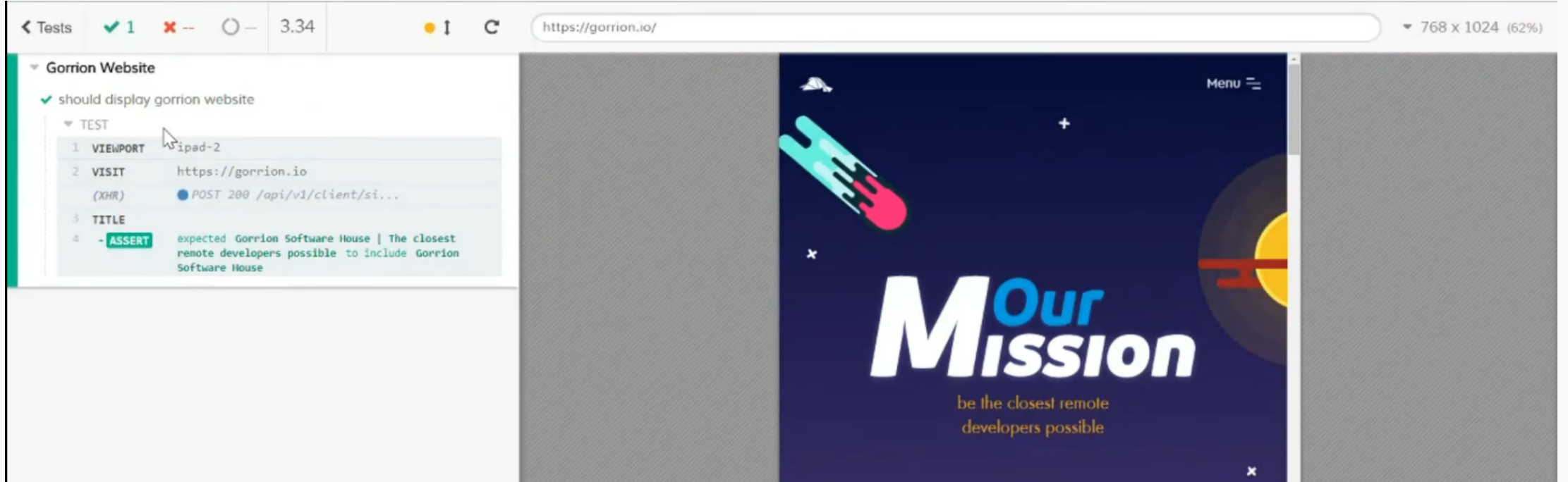
JS gorrior\_io.spec.js x

```
1 describe('Gorrion Website', () => {  
2     it('should display gorrior website', () => {  
3         cy.viewport('ipad-2')  
4         cy.visit('https://gorrior.io')  
5         cy.title().should('include', 'Gorrion Software House')  
6     })  
7 })
```

# Cypress - Ejemplo



# Cypress - Ejemplo



The screenshot displays the Cypress test runner interface. On the left, the 'Tests' panel shows a test suite 'Gorrion Website' with a single test 'should display gorrion website'. The test is marked as failed (red X). The test steps are listed below:

- 1 VIEWPORT iPad-2
- 2 VISIT https://gorrion.io
- (XHR) POST 200 /api/v1/client/si...
- 3 TITLE
- 4 - ASSERT expected Gorrion Software House | The closest remote developers possible to include Gorrion Software House

The main window shows the website being tested, which has a dark blue background with a stylized rocket and the text 'Our Mission' and 'be the closest remote developers possible'. The browser address bar shows 'https://gorrion.io/' and the window size is 768 x 1024 (62%).

# Tests Visuales

- Modalidad de test paralela al desarrollo y al resto de tests
- Objetivo: Evaluar la presencia de bugs visuales (distintos a los funcionales)
- Evalúan el output de la aplicación y lo comparan con lo especificado en su diseño
- Librería estándar en Vue:
  - Backstop JS

# Backstop JS

- Útil en escenarios en los que cambiamos algo visual y queremos cerciorarnos de no haber "roto" nada (visualmente)
- Ej: Cambiamos la propiedad de color de una clase y afectamos a varios elementos. (miles de entradas en un blog)








- Lo que hace por debajo es abrir las vistas en varios viewports (dimensiones de dispositivo), hace capturas de pantalla y las compara
- Al final del proceso nos muestra un reporte con los cambios que detecta entre versiones
- Podemos especificar el % de cambio

BackstopJS Report


backstop\_default

 BackstopJS

all


9 passed

1 failed




filename: backstop\_default\_EOM\_Homepage\_0\_document\_0\_phone.png

REFERENCE




TEST




filename: backstop\_default\_EOM\_Homepage\_0\_document\_1\_tablet.png

REFERENCE



TEST




all

9 passed

1 failed

Filter tests with search...



January 2018

Categories

- Unimplemented

Meta

- Log in
- Privacy Policy
- Comments Policy
- WordPress.org

Post Archives

- July 2018
- June 2018
- May 2018
- April 2018
- March 2018
- February 2018

Calendar

- August 2018

Calendar

- Call me at 1 502 867 5228

January 2018

Categories

- Unimplemented

Meta

- Log in
- Privacy Policy
- Comments Policy
- WordPress.org

Post Archives

- July 2018
- June 2018
- May 2018
- April 2018
- March 2018
- February 2018

Calendar

- August 2018

Calendar

- Call me at 1 502 867 5228

filename: backstop\_default\_EOM\_Contact\_Us\_Page\_0\_document\_0\_phone.png

REFERENCE

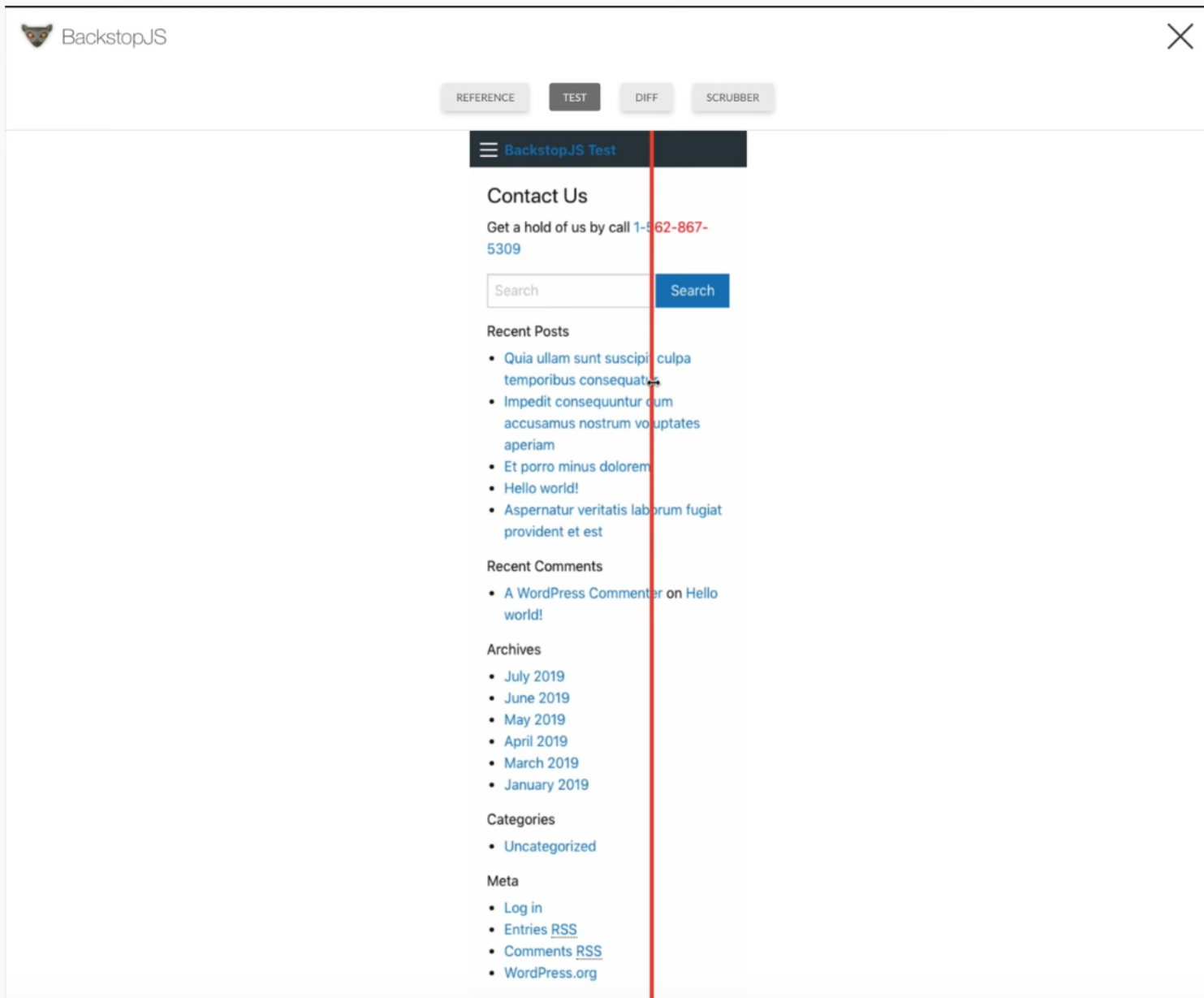
TEST

DIFF

filename: backstop\_default\_EOM\_Contact\_Us\_Page\_0\_document\_1\_tablet.png

REFERENCE

TEST





## 2. Nuxt.js

pron: /nʌkst/

# Introducción

- Framework para crear aplicaciones Vue.js
- Facilita a los desarrolladores el uso de las últimas tecnologías de una manera sencilla y organizada.
- Soporte "out of the box":
  - Server Side Rendering (SSR)
  - Static Site Generation (pre-rendering)
  - Meta tags
  - Code splitting
  - etc

# Server Side Rendering (SSR)

- Soluciona el problema de tratar con optimización SEO y meta tags
- Este problema surge porque no tenemos nada de contenido de nuestra app hasta que no se ejecuta js
- Con SSR: El servidor renderizará (generará el html de respuesta) y lo servirá al cliente o al crawler
- También mejora rendimiento

# Static Sites Generation

- Pre-rendering
- Generación de los html previa (renderizado)
- Beneficios de SSR y hosting gratuito
- Ideal en aplicaciones con pocas páginas (p.ej una página personal)



# Code Splitting

- Repartición del código js en varios archivos
- Nuxt creará un archivo .js para cada página
- Si nuestra app tiene 100 componentes pero la homepage sólo usa 10, en su .js no necesitamos los otros 90
- Gestión correcta de las dependencias

# Conclusión

- Nuxt nos facilita el uso de estas tecnologías ahorrándonos tiempo de investigación e implementación propia
- Elimina el riesgo de reinventar la rueda
- Habilitamos estas tecnologías simplemente activándolas en el archivo de configuración



## 3. Vite

pron: /vit/

- Conjunto de herramientas de desarrollo y construcción (Build Tool)
- Compite con webpack
- Mismo creador que Vue
- Descrita como "Next Generation Frontend Tooling"
- Nombrado así porque en francés significa "rápido"

# Incluye

- Dev server
- Herramienta para generar bundles de producción

# ¿Qué pretende resolver?

- El problema de velocidad de feedback en tiempo de desarrollo
- Vue Cli (con webpack), en proyectos grandes puede llegar a tardar hasta un minuto en "refrescar".
- Esto se produce por la necesidad de hacer bundles de código, problema que se arrastra desde antes de la existencia de soporte de los ESModules en los navegadores
- Si eliminamos el empaquetamiento de estos bundles y aprovechamos el soporte de los ESModules, ganaremos velocidad

# Además...

- Agnóstico de framework
- Introduce una nueva forma de gestionar dependencias, llegando incluso a hacer pre-bundles de dependencias pesadas para reducir llamadas http
- Dentro de un proyecto vite puedes enlazar a archivos .ts directamente y Vite los transpila automáticamente(usando ESBuild) (sin type-checking)

- Cómo ESBuild está escrito en GoLang, es 30x transpilando Ts que si lo haces con Ts...
- Se toma la decisión técnica de que el aumento de rendimiento merece la pena y se delega el chequeo de tipados en el programador, vía la ayuda de tsconfig en el IDE.



- `npm init @vitejs/app`
- En el paso de selección de framework podremos elegir entre varios de ellos, y sus versiones en js o ts

```
Héctor Go An@W11-MBP MINGW64 ~/Documents/EXPERIMENTS
```

```
$ npm init @vitejs/app
```

```
Need to install the following packages:
```

```
  @vitejs/create-app
```

```
Ok to proceed? (y) y
```

```
@vitejs/create-app is deprecated, use npm init vite instead
```

```
✓ Project name: ... hello-vite
```

```
✓ Select a framework: » vanilla
```

```
✓ Select a variant: » vanilla
```

```
Scaffolding project in C:\Users\Héctor Go An\Documents\EXPERIMENTS\hello-vite...
```

```
Done. Now run:
```

```
  cd hello-vite
```

```
  npm install
```

```
  npm run dev
```

- El npm install se ejecuta muy rápido, ya que sólo ha de instalar vite y sus dependencias
- Pero ... lo mejor es lo que tarda en levantar el servidor

```
vite v2.3.7 dev server running at:
```

```
> Local: http://localhost:3000/
```

```
> Network: use `--host` to expose
```

```
ready in 187ms.
```

## 3.1 - Proyectos a partir de Vite

- Vitest
- Plugin para módulos federados (vite-plugin-federation)

# Vitest



- Herramienta de testing
- Destaca por su velocidad
- Cobertura de varios tipos de test
- Uso del entorno de desarrollo Vue

# vite-plugin-federation



- Implementación del concepto módulos federados de webpack
- Orquestación de micro-frontends (y más)
- Un proyecto que se compone de varios "builds" independientes entre ellos



TECHDAY 

# *DESCUBRE* VUE



ORGANIZADO POR  
LA COMUNIDAD  
FRONTEND