



Simulación completa de  
la experiencia de usuario



Manuel López Camarena  
Abril, 2021  
AT Sistemas S.A.

AT 202101191211

# Interfaz

Estado de los tests

The screenshot shows a web browser window titled 'blogs\_application-actions' with the address 'http://localhost:8888/#/'. The browser is split into two panes. The left pane displays a test runner interface for 'TodoMVC'. It shows a summary of test results: 1 passed (green checkmark), 0 failed (red X), and 0 pending (grey circle). Below this, a table lists the test steps:

Type	Function	Alias(es)	# Calls
spy-1	inform	inform	1

The right pane displays the 'todos' application. It has a title 'What needs to be done?' and a list of three items: 'buy some cheese', 'feed the cat', and 'book a doctors appointment'. Below the list, it shows '3 items left' and three filter buttons: 'All', 'Active', and 'Completed'. At the bottom of the application, there is a footer that reads 'Double-click to edit a todo', 'Created by petehunt', and 'TodoMVC'.

Cronología

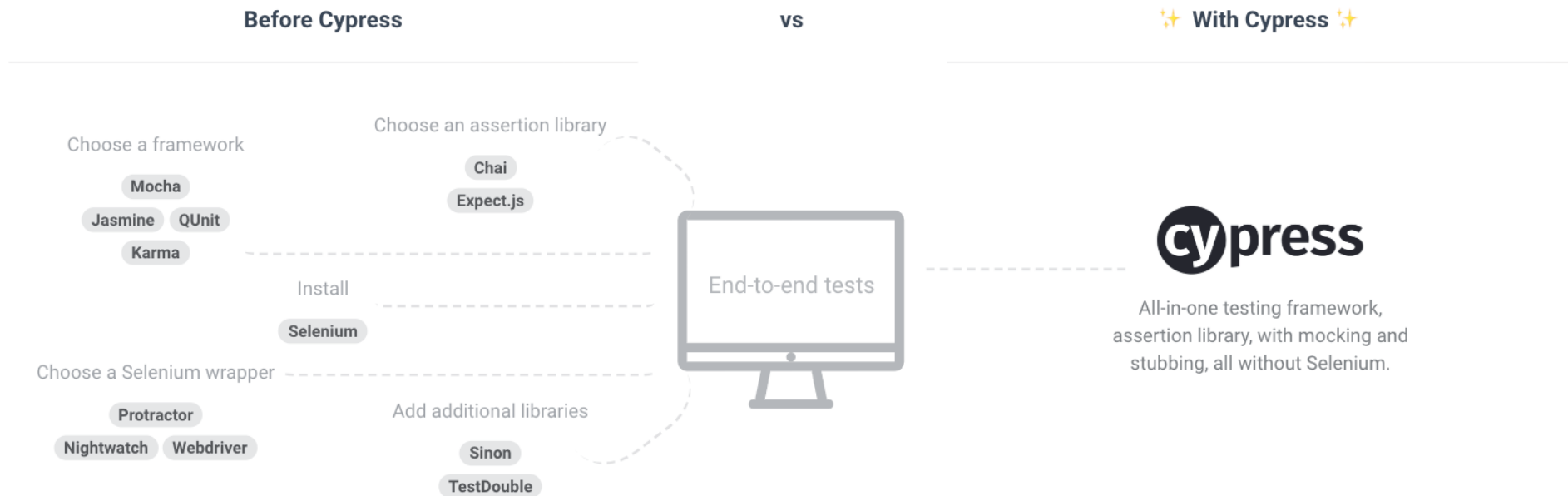
Vista previa app

# Navegación por snapshot en la cronología de ejecución

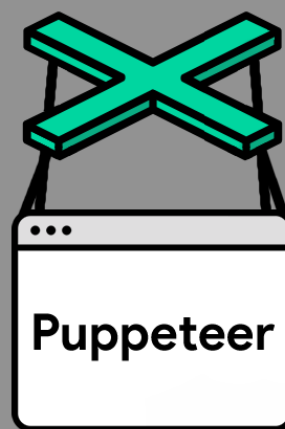
The screenshot displays a web browser window at `http://localhost:5000/`. On the left, a 'Form test' is shown with a table of test steps. A hand cursor points to step 8, which is highlighted in green and labeled 'Click!'. A large black arrow points from this step to the right, where a form is visible. The form contains fields for 'Name' (Molly), 'Email' (molly@dev.dev), and a 'Your message' text area containing 'Mind you if I ask some silly question?'. A 'SEND' button is at the bottom of the form.

Step	Method	URL/Action
1	VISIT	/
2	GET	form
3	GET	input[name="name"]
4	- TYPE	Molly
5	- ASSERT	expected <input#name> to have value Molly
6	GET	input[name="email"]
7	- TYPE	molly@dev.dev
8	- ASSERT	expected <input#email> to have value molly@dev.dev
9	GET	textarea
10	- TYPE	Mind you if I ask some silly question?
11	- ASSERT	expected <textarea#message> to have value Mind you if I ask some silly question?

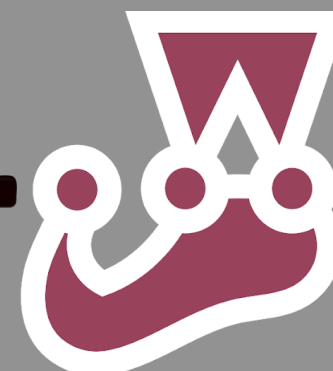
# Cypress, todo en uno.



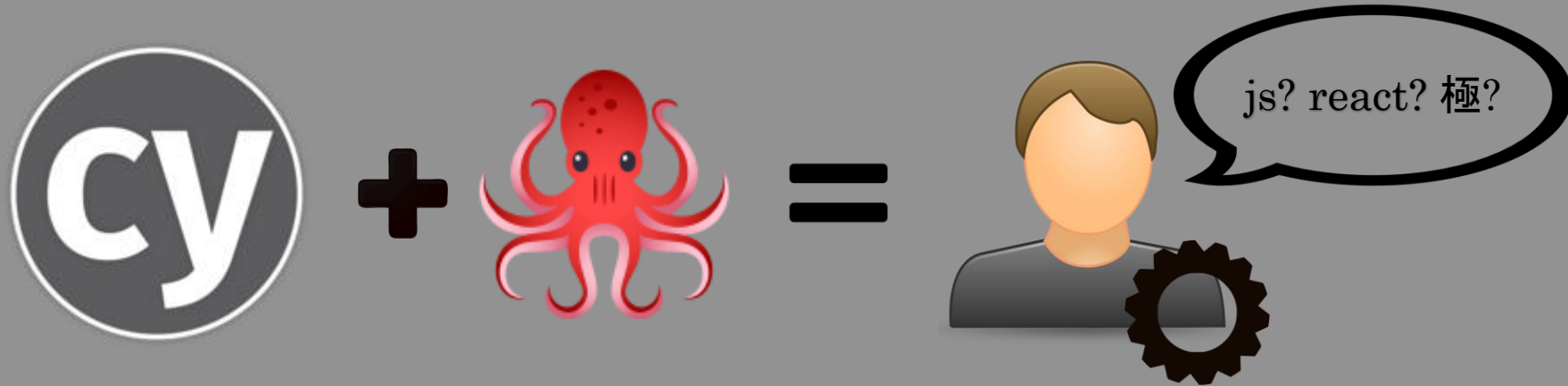
End-to-end testing. Cypress como alternativa a otras librerías.



+



# Integración con Testing Library.

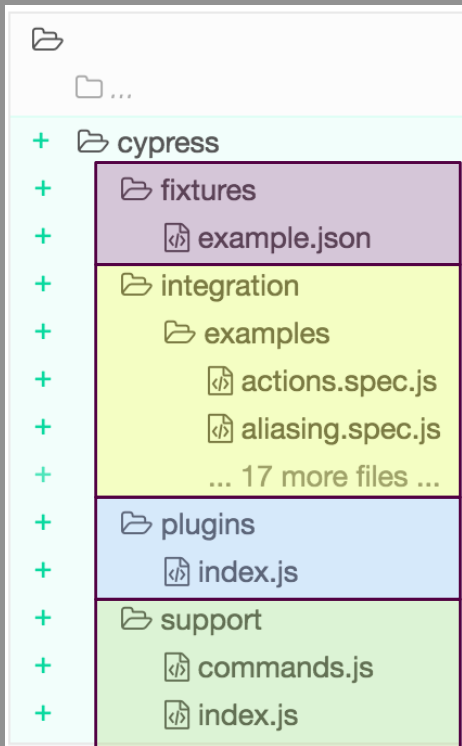


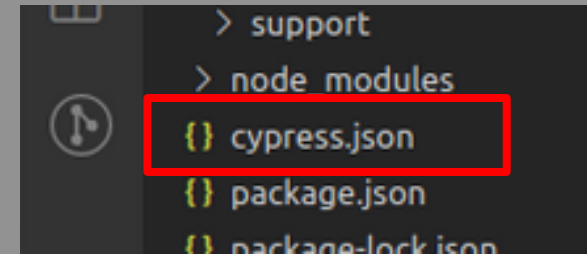
```
cy.findByRole('button', { name: /Jackie Chan/i }).click()
cy.findByRole('button', { name: /Button Text/i }).should('exist')
cy.findByRole('button', { name: /Non-existing Button Text/i }).should(
  'not.exist'
)
cy.findByLabelText(/Label text/i, { timeout: 7000 }).should('exist')

// findAllByText _inside_ a form element
cy.get('form')
  .findByText('button', { name: /Button Text/i })
  .should('exist')
cy.findByRole('dialog').within(() => {
  cy.findByRole('button', { name: /confirm/i })
})
```

Experiencia de usuario real  
y completa,  
se puede testear casi sin  
saber Javascript,  
el código es similar a  
escribir en inglés, con  
cierta nomenclatura de  
programación

# Estructura y variables de entorno.

	
+ fixtures	Carga de datos fijos/mock
+ example.json	
+ integration	Tests
+ examples	
+ actions.spec.js	
+ aliasing.spec.js	
... 17 more files ...	
+ plugins	Manipulación del comportamiento interno de Cypress
+ index.js	
+ support	Creación de comandos propios o integración de comandos de otras librerías (Testing library)
+ commands.js	
+ index.js	



```
{
  "host": "veronica.dev.local",
  "api_server": "http://localhost:8888/api/v1/"
}
```



```
Cypress.env() // {host: 'veronica.dev.local', api_server: 'http://localhost:8888/api/v1/'}
Cypress.env('host') // 'veronica.dev.local'
Cypress.env('api_server') // 'http://localhost:8888/api/v1/'
```

# Documentación

## Cypress:

<https://docs.cypress.io/guides/overview/why-cypress>

<https://github.com/cypress-io/cypress>

## Testing library:

<https://testing-library.com/docs/>

<https://github.com/testing-library/cypress-testing-library>