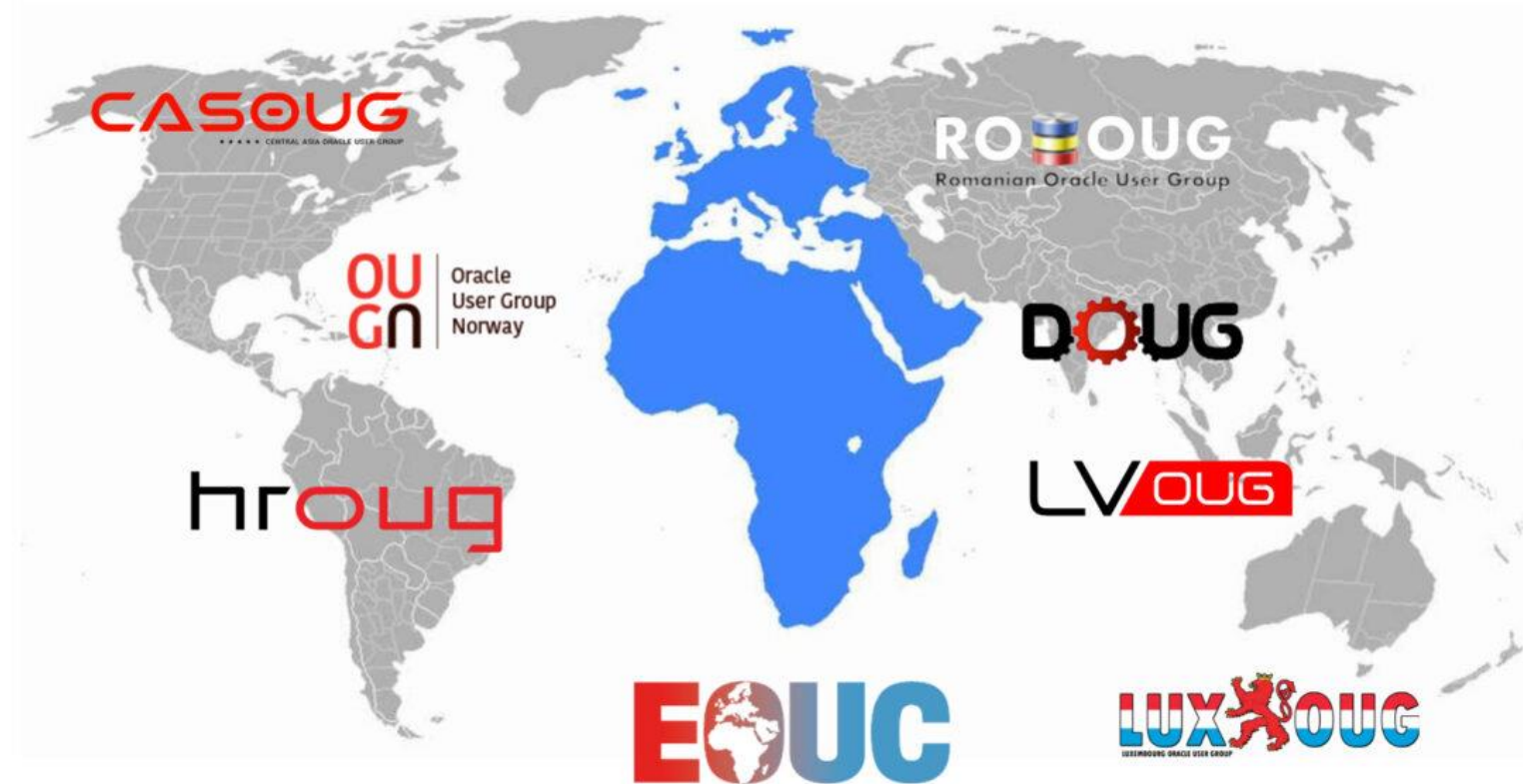


Power of automated testing

EMEA Community Tour 2022



Lino Schildenfeld

@LinoSchild

lschilde.blogspot.com

APEX Office hours

September 29, 2022 14:00 - 15:00 UTC

Start Times Around the World

Subscribe

⚙️ Actions

What's new in Flows for APEX 22.2!

In this session, the Flows for APEX team introduces APEX a powerful solution for modeling. Key features include integration with AF easier gateway routing definition.

Join Niels de Bruijn (MT AG) and Richard C Evans

Catch up on the recording for our last session!

Don't miss our next live session!

This session: <https://asktom.oracle.com/pl>

Total Videos 664

Most Recent ▾

↺ Reset

▶ View Upcoming



Introducing Oracle Machine Learning for R for Oracle Autonomous Database

Mark Hornick, Sherry LaMonica
285 views · August 30, 2022



Preguntas Frecuentes sobre Oracle APEX

Mónica Godoy
499 views · August 25, 2022



Multi-Cloud Security - Oracle Database and Azure Active Directory integration

Alan Williams, Richard C Evans
253 views · August 24, 2022

apex.oracle.com/officehours

AUSOUG APEX News



[Become a Member](#) [What's On](#) [Branches](#) [Resources](#) [Past Events](#) [Blog](#) [My Account](#)

Connect 2022
One Oracle, Endless Opportunities
November 7 - 10 | Virtual

www.ausoug.org.au/connect-2022/





Mentor and Speaker Hub

- Our goal is to *connect* speakers with mentors to assist in *preparing* technical sessions and *improving* presentation skills

Interested? Read more and get in touch

<https://mashprogram.wordpress.com>

My story

@LinoSchilder



AUSOUG



APEX RD
research & development

- AUSOUG APEX committee chair
- NZ APEX meetup organizer
- APEX blogger
- Conference speaker
- APEX World Member of the Month
- Mentor and trainer

Today's goal

Sample Database Application Help demo

[Cancel](#) [Add Customer](#)

First Name *	<input type="text"/>	Last Name *	<input type="text"/>
Street Address	<input type="text"/>	Line 2	<input type="text"/>
City	<input type="text"/>	State *	<input type="text" value="- Choose State -"/>
Zip Code *	<input type="text"/>		
Credit Limit *	<input type="text"/>		
Phone Number	<input type="text" value="999-999-9999"/>	Alternate Number	<input type="text" value="999-999-9999"/>
Email	<input type="text"/>	URL	<input type="text"/>
Tags	<input type="text"/>		

WHY Automated testing?!

- **Improves software quality**
- **Much faster than manual**
- Saves Time and Money
- Reduces manual errors
- Improves Standards
- Faster time to market
- Faster Feedback
- Improved Productivity
- Faster Debugging
- Return on investment is high
- **Faster APEX Upgrading**
- Improves experience - **Happier clients**

Interesting – decision time

- **Licence cost or no cost option**
- **Support**
- **Usability and maintenance**
- **What do I want it to do**
 - Record a video
 - Upload to the cloud
 - Store and Export results
 - Handle a mobile testing
 - Documentation
 - Easy to use/install

Automated testing tools

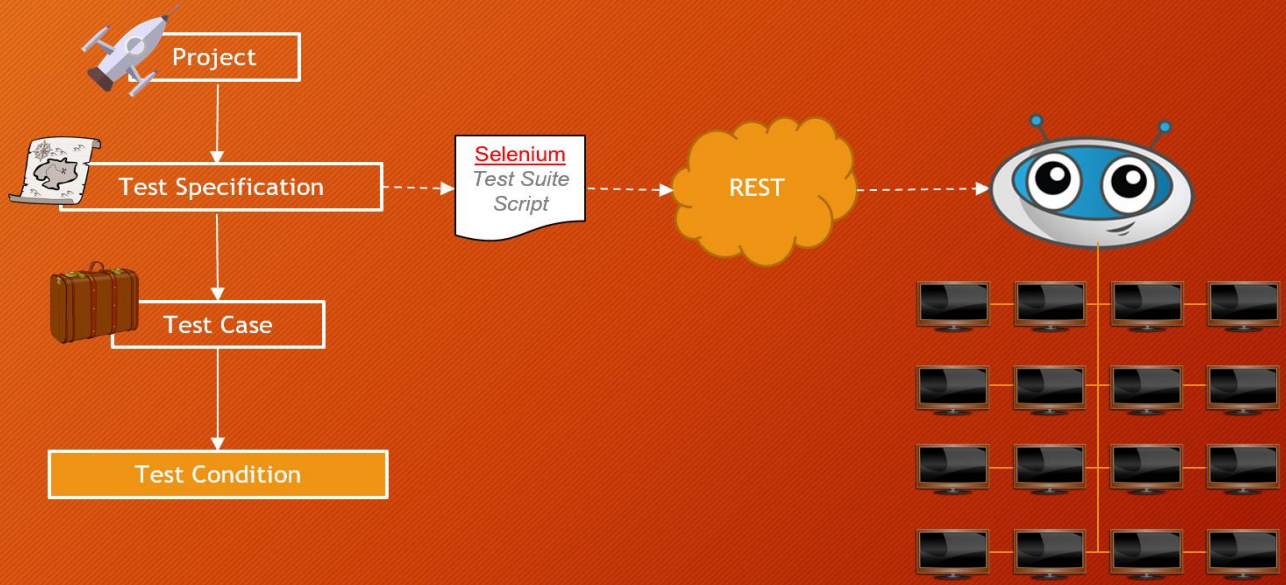
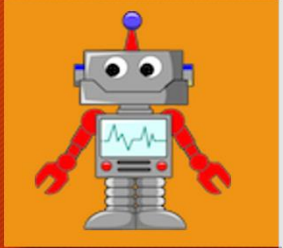
- [Selenium](#)
- **CYPRESS**
- ATAF
- [utPLSQL](#)
- Codeconcept.io
- JMeter
- Katalon
- Gatling
- HammerDb
- Appium
- Autoit
- Eggplant
- **Ghost Inspector**
- [Playwright](#)

- Rarely run and costly
- Can improve your coding standards
- Built by developers as part of DEV cycle
- Biggest challenge is synchronization (unreliable)

“Web testing is one of the critical parts of modern software lifecycle”

- 6 Running Scripts Manually
- 7 Running Scripts In The Cloud
- 8 In Development - Web Driver
- 9 In Development - Kantu

Running Scripts In The Cloud



CYPRESS

WHY CYPRESS

- **Easy to get going** and **learn** how to use
- **Good documentation**
- It can video **record** the entire **tests** & takes snapshots
- **Automatically waits** (for commands and assertions before moving on*)
- Supports Firefox, MS Edge, Chrome
- Uses Mocha syntax and Chai assertions
- Smaller community with momentum in last few years



How it works

Installation

- Install node.js
- Create a directory
- Position your self into it
- Run `npm init -y`
- Run `npm install cypress --save-dev`

```
LS@LS-HP-x360 MINGW64 ~/Desktop/CYPRESS
$ npm install cypress -D
npm WARN deprecated har-validator@5.1.5: this library is no longer supported

> cypress@5.1.0 postinstall C:\Users\LS\Desktop\CYPRESS\node_modules\cypress
> node index.js --exec install

Installing Cypress (version: 5.1.0)

  ✓ Downloaded Cypress
  ✓ Unzipped Cypress
  ✓ Finished Installation C:\Users\LS\AppData\Local\Cypress\Cache\5.1.0

You can now open Cypress by running: node_modules\.bin\cypress open

https://on.cypress.io/installing-cypress

npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN CYPRESS@1.0.0 No description
npm WARN CYPRESS@1.0.0 No repository field.

+ cypress@5.1.0
added 216 packages from 147 contributors and audited 216 packages in 256.478s

12 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

First run

- We need text editor and a command line
- Run `npx cypress open`
- Check your `package.json` for correct dependencies
- Notice cypress subdirectory (rm examples)
- Create new directory under `integration`
- `npx cypress verify`
- `npx cypress info`
- `npx cypress version`

```
LS@LS-HP-x360 MINGW64 ~/Desktop/CYPRESS  
$ npx cypress run
```

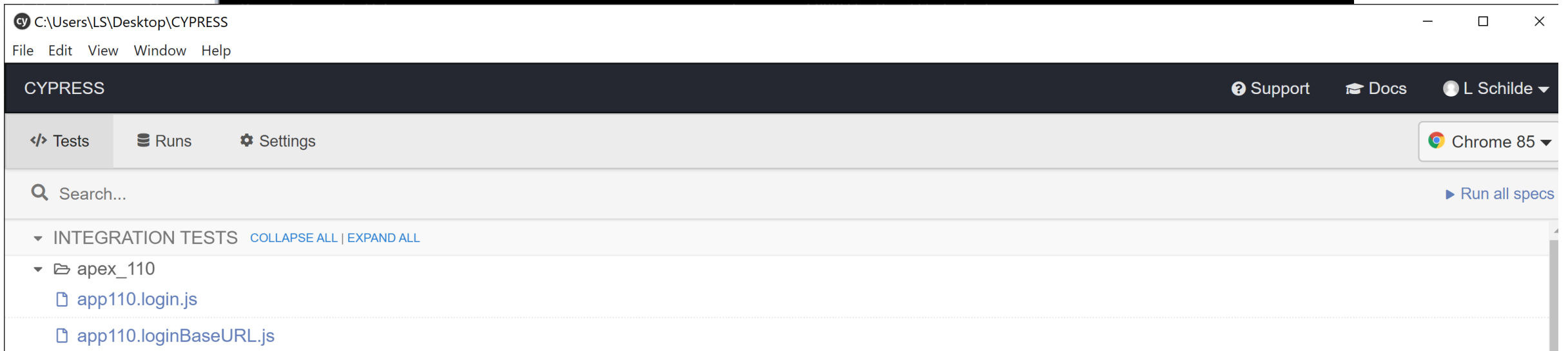
HelloWorld.js

```
describe('Welcome to Cypress', () => {  
  | context('Trying out things', () => {  
    | Open Cypress | Set ".only"  
    | it('Test 1', () => {  
      | expect(1).to.equal(1);  
    | })  
    | it.skip('Test2', () => {  
      | expect(true).to.equal(true);  
    | });  
  | });  
});|
```


TestRunner – older version

```
LS@LS-HP-x360 MINGW64 ~/Desktop/CYPRESS
$ npx cypress open
It looks like this is your first time using Cypress: 5.1.0

  ✓ Verified Cypress! C:\Users\LS\AppData\Local\Cypress\Cache\5.1.0\Cypress
Opening Cypress...
```




The screenshot shows the Cypress Test Runner application window. The title bar indicates the path `C:\Users\LS\Desktop\CYPRESS`. The application has a dark theme and a menu bar with `File`, `Edit`, `View`, `Window`, and `Help`. The main interface includes a top navigation bar with `CYPRESS`, `Support`, `Docs`, and a user profile `L Schilde`. Below this is a secondary navigation bar with `Tests`, `Runs`, and `Settings`, along with a browser selection dropdown showing `Chrome 85`. A search bar is located below the navigation. The main content area displays a tree view of test files under the heading `INTEGRATION TESTS`, with options to `COLLAPSE ALL` or `EXPAND ALL`. The tree shows a folder `apex_110` containing two files: `app110.login.js` and `app110.loginBaseURL.js`. A `Run all specs` button is visible on the right side of the interface.

CYPRESS 10.2

```
lschi@LAPTOP-3MI58QCO MINGW64 ~/Desktop/OUG_CYPRESS10_DEMO
$ npx cypress open
It looks like this is your first time using Cypress: 10.2.0

[STARTED] Task without title.
[TITLE]   Verified Cypress!       C:\Users\lschi\AppData\Local\Cypress\Cache\10.2.0\Cypress
[SUCCESS] Verified Cypress!       C:\Users\lschi\AppData\Local\Cypress\Cache\10.2.0\Cypress


Opening Cypress...
```



E2E Testing

Build and test the entire experience of your application from end-to-end to ensure each flow matches your expectations.

Configured




Component Testing

Build and test your components from your design system in isolation in order to ensure each state matches your expectations.


Not Configured

Choose a Browser


Choose your preferred browser for E2E testing.




Chrome
v102



Edge
v103



Electron
v100



Firefox
v101

[Start E2E Testing in Chrome](#)

[Switch testing type](#)

E2E specs	Last updated
cyress\e2e	
OUG.cyjs	33 minutes ago
demo1.cyjs	33 minutes ago

Key tips

Selections

- Most popular **Chai assertions** (including Sinon & jQuery)
- **Hardest** in web automation testing to get right
- Learning how to do this is a **key challenge**
- Modal dialogs, reports and interactive grid validation
- Watch out for modals

```
cy.url().should("contain", "110:LOGIN_DESKTOP");  
cy.get('.t-Login-title').should("contain", "Sample Database Application");
```

<https://docs.cypress.io/guides/references/assertions.html#Chai>

Final demo

Playwright

```
Ischi@LAPTOP-3MI58QCO MINGW64 ~/Desktop/Playwright
$ npm init playwright@latest
npx: installed 1 in 1.692s
Getting started with writing end-to-end tests with Playwright:
Initializing project in '.'
? Do you want to use TypeScript or JavaScript? ...
> TypeScript
  JavaScript
✓ Do you want to use TypeScript or JavaScript? · JavaScript
✓ Do you want to use TypeScript or JavaScript? · JavaScript
✓ Do you want to use TypeScript or JavaScript? · JavaScript
✓ Do you want to use TypeScript or JavaScript? · JavaScript
? Where to put your end-to-end tests? » tests
✓ Where to put your end-to-end tests? · tests
? Add a GitHub Actions workflow? (y/N) » false
✓ Add a GitHub Actions workflow? (y/N) · true
```

```
tschi@LAPTOP-3MI58QCO MINGW64 ~/Desktop/Playwright
```

```
$ npx playwright test
```

```
Running 3 tests using 3 workers
```

```
[1/3] [chromium] > example.spec.js:4:1 > homepage has Playwright in title and get started 1
```

```
[2/3] [firefox] > example.spec.js:4:1 > homepage has Playwright in title and get started li
```

```
[3/3] [webkit] > example.spec.js:4:1 > homepage has Playwright in title and get started lin
```

```
3 passed (6s)
```

```
To open last HTML report run:
```

```
npx playwright show-report
```

```
tschi@LAPTOP-3MI58QCO MINGW64 ~/Desktop/Playwright
```

```
$
```

Q		All 3	Passed 3	Failed 0	Flaky 0	Skipped 0
✓	example.spec.js					6.8s
✓	homepage has Playwright in title and get started link linking to the intro page		chromium			3.1s
	example.spec.js:4					
✓	homepage has Playwright in title and get started link linking to the intro page		firefox			2.0s
	example.spec.js:4					
✓	homepage has Playwright in title and get started link linking to the intro page		webkit			1.7s
	example.spec.js:4					

MINGW64:/c/Users/Ischi/Desktop/Playwright

```
Running 3 tests using 3 workers
[1/3] [chromium] > example.spec.js:4:1 > homepage has Playwright in title and get start
[2/3] [firefox] > example.spec.js:4:1 > homepage has Playwright in title and get starte
[3/3] [webkit] > example.spec.js:4:1 > homepage has Playwright in title and get started
```

3 passed (6s)

To open last HTML report run:

```
npx playwright show-report
```

```
Ischi@LAPTOP-3MI58QCO MINGW64 ~/Desktop/Playwright
```

```
$ npx playwright show-report
```

```
Serving HTML report at http://127.0.0.1:9323. Press Ctrl+C to quit.
```


C: > Users > Ischi > Desktop > Playwright > tests > JS example.spec.js > ...

```
1  // @ts-check
2  const { test, expect } = require('@playwright/test');
3
4  test('homepage has Playwright in title and get started link linking to the intro page', async ({ page }) => {
5    await page.goto('https://playwright.dev/');
6
7    // Expect a title "to contain" a substring.
8    await expect(page).toHaveTitle(/Playwright/);
9
10   // create a locator
11   const getStarted = page.locator('text=Get Started');
12
13   // Expect an attribute "to be strictly equal" to the value.
14   await expect(getStarted).toHaveAttribute('href', '/docs/intro');
15
16   // Click the get started link.
17   await getStarted.click();
18
19   // Expects the URL to contain intro.
20   await expect(page).toHaveURL(/.*intro/);
21 });
22
```

Manual run

- Runs all tests 'headlessly'
- Ability to pass in parameters
- By default `cypress run` **records a video locally**
- Disable video with

```
--config video[Recording]=false
```

- Record into Dashboard with

```
--record --key
```

Options	Option
<code>--headed</code>	
<code>--headless</code>	
<code>--help, -h</code>	
<code>--key, -k</code>	
<code>--no-exit</code>	
<code>--parallel</code>	
<code>--port, -p</code>	
<code>--project, -P</code>	
<code>--browser, -b</code>	
<code>--quiet, -q</code>	
<code>--record</code>	

```
$ npx cypress run --record
```

(Run Starting)

```
Cypress: 5.1.0
Browser: Electron 83 (headless)
Specs: 34 found (apex_110\app110.login.js, apex_110\app110.loginBaseUrl.js, apex_110\app110.loginCommon.js, apex_110\app110.loginPwd_Assertions.js, apex_110\app110.loginPwd_AssertionsPost.js, apex_110\app110.loginPwd.js, apex_110\app110.loginPwdBad.js, apex_110\ap...)
Params: Tag: false, Group: false, Parallel: false
Run URL: https://dashboard.cypress.io/projects/9vk5ep/runs/1
```

Running: apex_110\app110.login.js

(1 of 34)

```
APEX login page
  ✓ simple login (3564ms)
```

1 passing (4s)

(Results)

```
Tests: 1
Passing: 1
Failing: 0
Pending: 0
Skipped: 0
Screenshots: 0
Video: true
Duration: 3 seconds
Spec Ran: apex_110\app110.login.js
```

Test results

(Run Finished)

Spec	Tests	Passing	Failing	Pending	Skipped
✓ apex_110\app110.login.js	00:03	1	1	-	-
✓ apex_110\app110.loginBaseURL.js	00:02	1	1	-	-
✓ apex_110\app110.loginCommon.js	00:11	2	2	-	-
✓ apex_110\app110.loginPwd_Assertions.js	00:04	1	1	-	-
✗ apex_110\app110.loginPwd_AssertionsPost.js	00:09	1	-	1	-
✓ apex_110\app110.loginPwd.js	00:04	1	1	-	-
✓ apex_110\app110.loginPwdBad.js	00:04	1	1	-	-
✓ apex_110\app110.loginPwdObf.js	00:03	1	1	-	-
✓ apex_110\app110.p2aMultiple.js	00:10	2	2	-	-
✓ apex_110\app110.p2BypassLoginX.js	00:10	1	1	-	-

Dashboard

- Optional web-based option
- Visualizes and records your runs
- Stores recordings and snapshots
- Runs tests faster
- Efficiency with parallelization
- Automatic load balancing

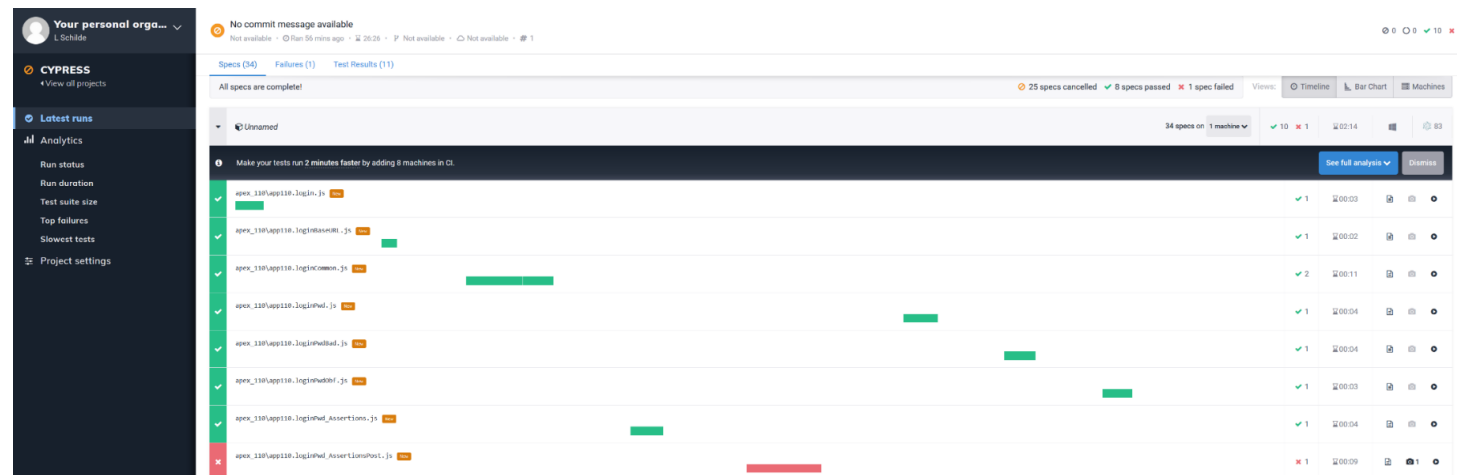
Getting Started with Cypress Dashboard ➤ Inbox

Success @ Cypress.io <success@cypress.io> [Unsubscribe](#)
to me ▼

Hi there,

Thanks for signing up for the Cypress Dashboard!

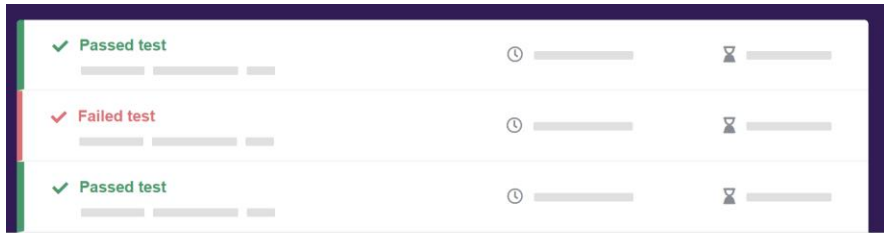
Using the Dashboard will accelerate your testing and unlock insights for your team.



Recordings

- records a video for each spec file when running tests during
- `cypress run`

```
cypress run --record --key 5b5a9e17-19e0-4dca-8d37-bc4bec37b1e4
```



You could see test recordings here!

Connect to Cypress Dashboard for free:

- Record test runs in CI and debug failed tests with ease
- Understand the health of your tests with test analytics
- Improve testing efficiency with parallelization, load balancing, and more

[Connect to Dashboard](#)

After logging in, you'll see recorded test runs here and in your Cypress Dashboard.

Congrats! You recorded your first run 🎉 Inbox x

Success @ Cypress.io <success@cypress.io>
to me ▾

Hi L,

Congratulations on recording your first test run to the Cypress Dashboard!

Each recording unlocks powerful insights into each of your test runs.

Recording runs in CI can help you dig even deeper, allowing you to:

- 👁️ See all the tests running in your CI pipeline
- 🔍 Find the precise error that resulted in a failed test
- 📺 Watch a video recording of failed tests

Want to supercharge your testing and increase deployment velocity?

👉 Check out our guide to running Cypress tests in [Continuous Integration](#).

Happy Testing!

- Cypress Success Team

Set up project X

What's the name of the project? (You can change this later)

CYPRESS_DEMO1

Who should own this project? ? [Manage organizations](#)

👤 Your personal organization | ▾

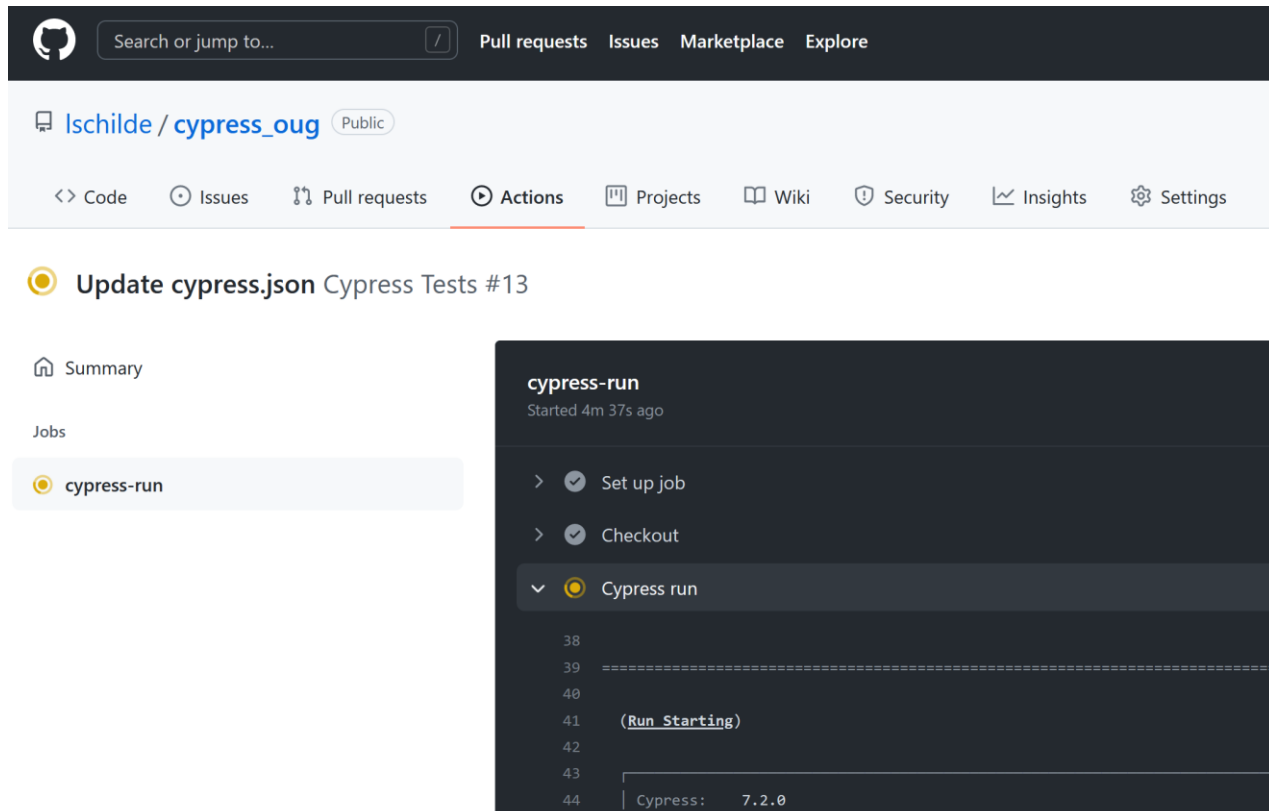
Who should see the runs and recordings? ?

- 👁️ **Public:** Anyone has access.
- 🔒 **Private:** Only invited users have access.

[Set up project](#)

Continuous Integration

- **Cypress is compatible with all Continuous Integration (CI) providers**
- Integrates with SLACK, GitHub, GitLab, Circle CI



The screenshot shows a GitHub Actions workflow run for the repository 'lschilde / cypress_oug'. The workflow is titled 'Update cypress.json Cypress Tests #13'. The 'Jobs' section shows a single job named 'cypress-run'. The 'Summary' section is visible, and the 'Jobs' section is expanded to show the 'cypress-run' job. The 'cypress-run' job is expanded to show its steps: 'Set up job', 'Checkout', and 'Cypress run'. The 'Cypress run' step is expanded to show its output, which includes the line 'Cypress: 7.2.0'.

Search or jump to... Pull requests Issues Marketplace Explore

lschilde / cypress_oug Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Update cypress.json Cypress Tests #13

Summary

Jobs

cypress-run

cypress-run
Started 4m 37s ago

> ✓ Set up job

> ✓ Checkout

▼ ● Cypress run

38

39 =====

40

41 (Run Starting)

42

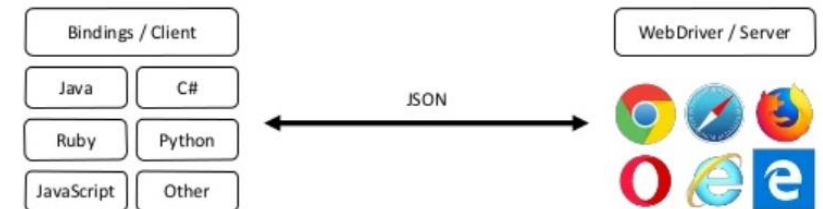
43

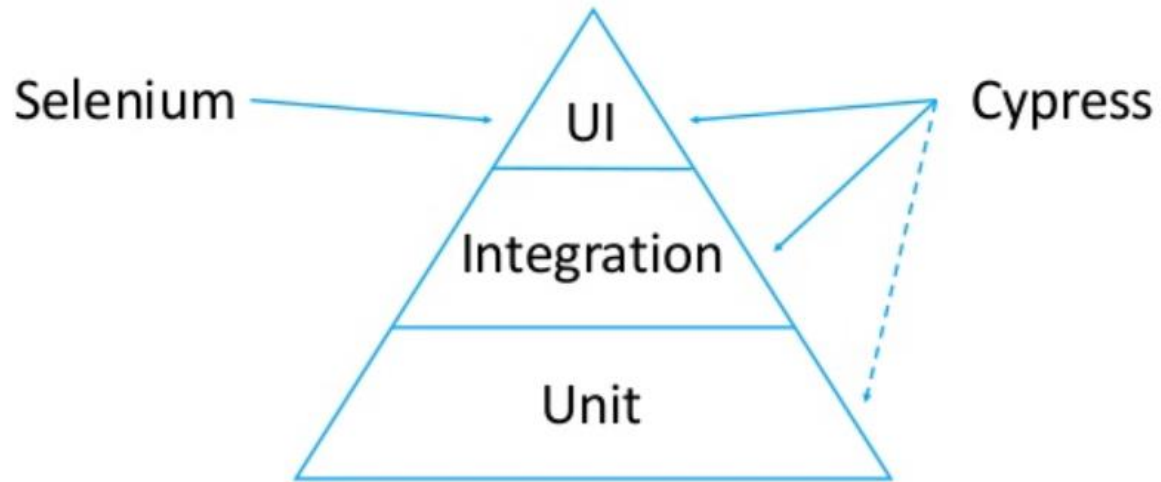
44 Cypress: 7.2.0

vs. Selenium?

- **Different architecture** (Library vs complete framework)
- **Easier** to set/get test **video recording**/snapshots
- Does not use WebDriver and **runs test directly in the browser**
- 'record your steps' is not supported
- JS vs 9 prog. languages (JS, Java, Perl, PHP, Python, Ruby, and C#..)
- **Fewer browser support** (cross browser testing limitation)
- Faster than Selenium
- Not as flexible
- Still market place for both

Selenium Overview





Selenium

- Java
- C#
- Ruby
- Python
- JavaScript
- PHP, Perl, Golang, Haskel, Objective-C, R, Elixir, etc.

Cypress

- JavaScript

Selenium

- ID
- Name
- Class Name
- Tag Name
- (Partial) Link Text
- CSS Selector
- XPath Selector

Cypress

- jQuery

Selenium



And many more ...



Cypress

Ghost Inspector



Ghost Inspector - Installation

- Installable as Browser add on for Firefox and Chrome

<https://ghostinspector.com/docs/getting-started/>

- Guides
- Environments & Networking
- Account Management
- Integration Tutorials
- API
- Resources

How it works

1. Sign up
To get started, first [sign up for a free trial account](#).
2. Test recorder
To create tests, you can install our [test recorder](#) in your browser:
 

After installing the extension, a Ghost Inspector icon will appear in your toolbar. Clicking this icon will present you with the Ghost Inspector recording tool. You will need to log into your Ghost

Ghost Inspector – First run

- “Codeless testing”
- Allows you to record and play browser tests quickly and easy
- Very similar how Selenium IDE worked
- No Open Source version only 14 days trial
- We must buy a license

Ghost Inspector

Websites must be accessible to our servers. If a login is required, you must record the login step as part of your test.

✓ I understand, don't show again.

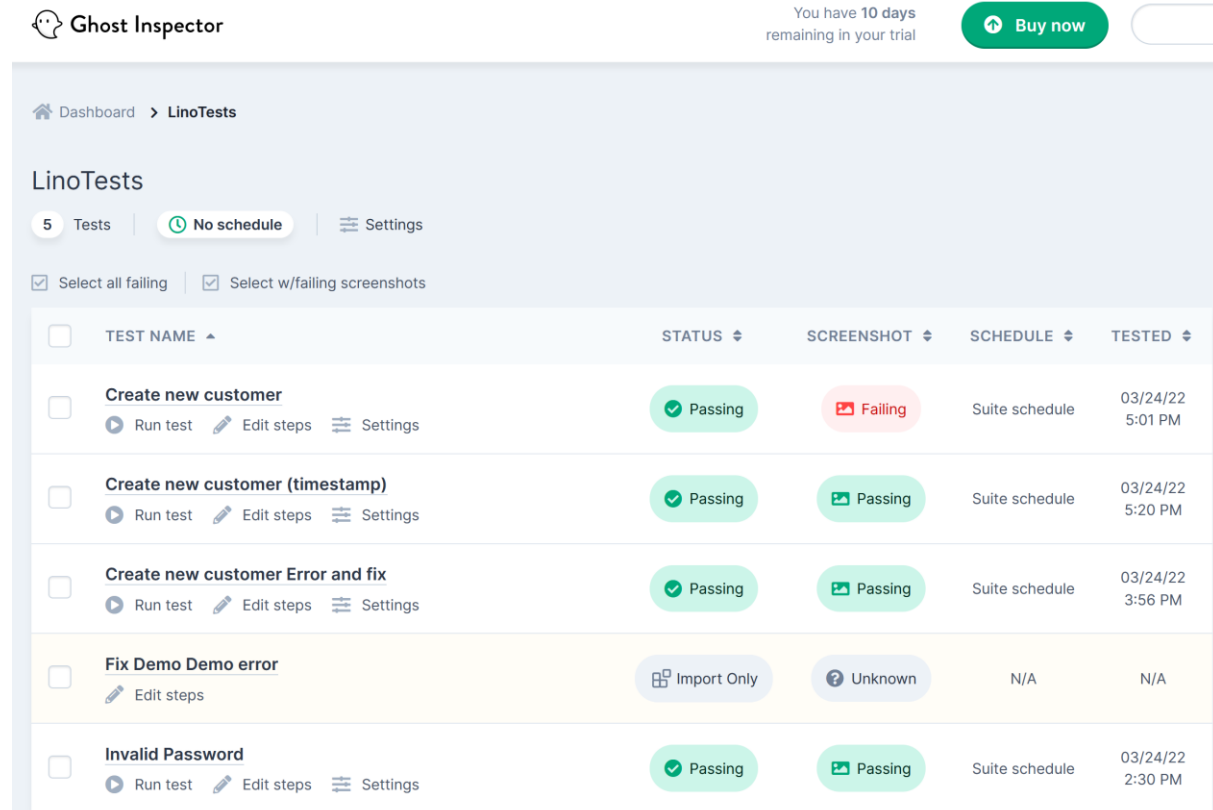
Start recording your operations below. Click the Ghost Inspector toolbar icon at anytime to switch modes or complete your recording.

Create a new test Add to existing test

Recording...

Ghost Inspector - Dashboard

- Test recorder selectors are not 100% there but still great
- Offers familiar selectors CSS and XPath
- Easy to learn
- “Lives in a cloud”

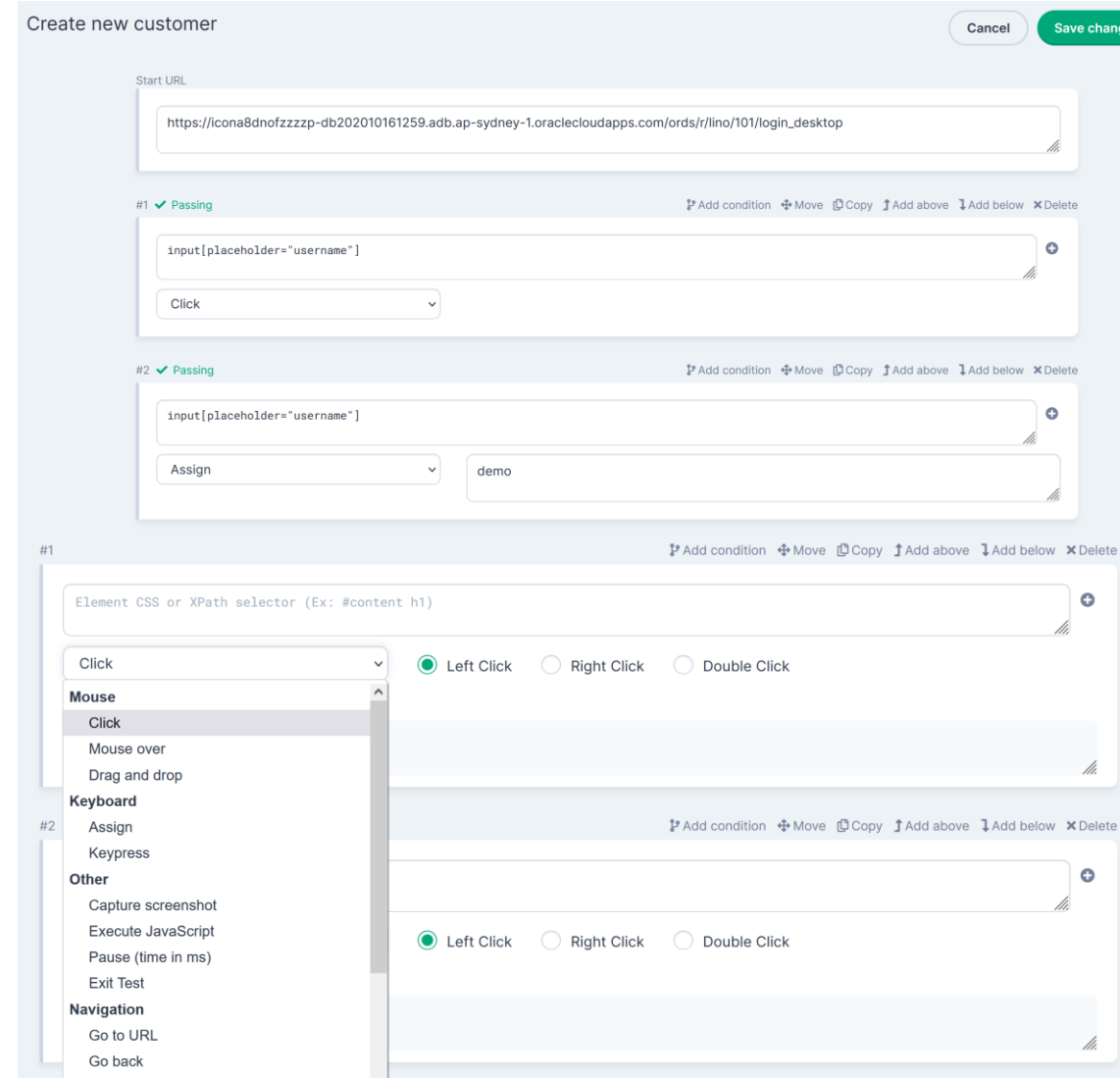


The screenshot shows the Ghost Inspector dashboard. At the top right, it says "You have 10 days remaining in your trial" and a "Buy now" button. The main content area is titled "LinoTests" and shows a list of tests. The tests are:

TEST NAME	STATUS	SCREENSHOT	SCHEDULE	TESTED
Create new customer Run test Edit steps Settings	Passing	Failing	Suite schedule	03/24/22 5:01 PM
Create new customer (timestamp) Run test Edit steps Settings	Passing	Passing	Suite schedule	03/24/22 5:20 PM
Create new customer Error and fix Run test Edit steps Settings	Passing	Passing	Suite schedule	03/24/22 3:56 PM
Fix Demo Demo error Edit steps	Import Only	Unknown	N/A	N/A
Invalid Password Run test Edit steps Settings	Passing	Passing	Suite schedule	03/24/22 2:30 PM

Ghost Inspector – Test steps

- Saves videos and snapshots
- Supports geolocations
- Multiple screen sizes
- Multiple browser versions
- It can send notifications to:
 - Emails
 - REST webhook
 - MS Teams, Slack or PagerDuty
- It can get imported and exported to/from Selenium



The screenshot displays the 'Create new customer' interface in Ghost Inspector. At the top, there is a 'Start URL' field containing the URL: `https://icon8dnofzzzp-db202010161259.adb.ap-sydney-1.oraclecloudapps.com/ords/r/lino/101/login_desktop`. Below this, there are two test steps, both marked as '#1 Passing'.

The first step is a 'Click' action on the element selector `input[placeholder="username"]`. The second step is an 'Assign' action on the same element selector, with the value 'demo' assigned to it.

At the bottom, a third step is partially visible, showing an 'Element CSS or XPath selector (Ex: #content h1)' field and a dropdown menu for the action type. The dropdown menu is open, showing options for 'Click', 'Mouse', 'Keyboard', and 'Other'. Under 'Click', there are radio buttons for 'Left Click' (selected), 'Right Click', and 'Double Click'. Under 'Mouse', there are options for 'Click', 'Mouse over', and 'Drag and drop'. Under 'Keyboard', there are options for 'Assign' and 'Keypress'. Under 'Other', there are options for 'Capture screenshot', 'Execute JavaScript', 'Pause (time in ms)', and 'Exit Test'. Under 'Navigation', there are options for 'Go to URL', 'Go back', and 'Go forward'.

Ghost Inspector – Variables

- Built-in
- Mock Variables
- Runtime Variables
- Extract from Element
- Extract from JavaScript
- Suite-level Variables
- Organization-level Variables
- Execute with CSV File

Set Variable

One common scenario where this comes up is when we need to generate a unique email address for a signup form, then wish to assert that email address later on.

The screenshot displays two steps in a test suite:

- Step #1:** A 'Set Variable' action is configured. The value field contains the expression `test+{{TIMESTAMP}}@domain.com`. The variable name field is labeled `email`.
- Step #2:** An 'Assign' action is configured. The value field contains the expression `{{email}}`. The variable name field is labeled `#email-input`.

Each step includes control icons for 'Move', 'Add Above', 'Add Below', and 'Delete'.

Ghost Inspector – CLI

- It offers a Command Line Interface (CLI)
- Run as Docker image
- API Key
- Multiple config parameters

```
npm install -g ghost-inspector
```

```
ghost-inspector test execute {{my-test-id}}
```

```
export GHOST_INSPECTOR_API_KEY={{my-api-key}}
```

```
ghost-inspector test execute {{test-id}} \  
  --browser chrome --browser firefox \  
  --region us-east-1 --region ap-south-1 \  
  --viewport 1024x768 --viewport 800x600 \  
  --immediate
```

Summary

- It is about learning selectors and standardizing them
- Assertions - Verifying that expected actually happened
- Flakiness

Summary

- Make AAT part of your dev lifecycles
 - Reduces time and costs testing your apps
- Ghost Inspector or Cypress (Playwright or Selenium)
 - You can't go wrong
- Integrate well with any CI tools
- Standardize methods
 - Reuse and share code
 - Passwords
 - Input Variables
 - Consider changing your APEX apps

Q&A

Thank you for attending



Resources

- <https://docs.cypress.io/>
- https://www.youtube.com/watch?v=QFzN_0soxiQ
- <https://dzone.com/articles/selenium-vs-cypress-is-webdriver-on-its-way-out-1>
- <https://automationrhapsody.com/cypress-vs-selenium-end-era/>
- <https://github.com/cypress-io/cypress-realworld-app>
- <https://docs.cypress.io/guides/dashboard/github-integration>