ENFIRE

EN FÎRE

Introduction: Purpose of the Demo

Why automate: Outlining the current manual testing workflow for ENFIRE and why it's slow, error-prone, and not scalable.

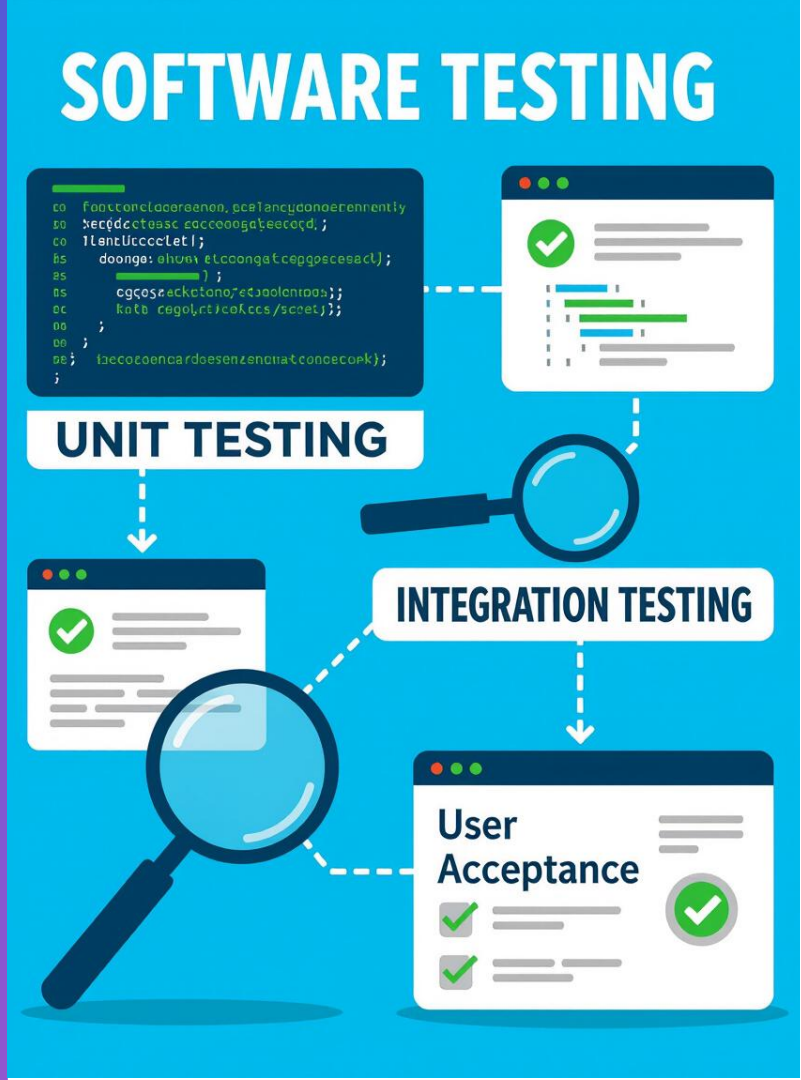
First steps in automation:"Today" mode. coordinate- and screenshot-based automation approach and its fragilities.

Demo: demo the UI Testing Toolkit, showing how it records and replays workflows and uses screenshot comparisons to verify outcomes.

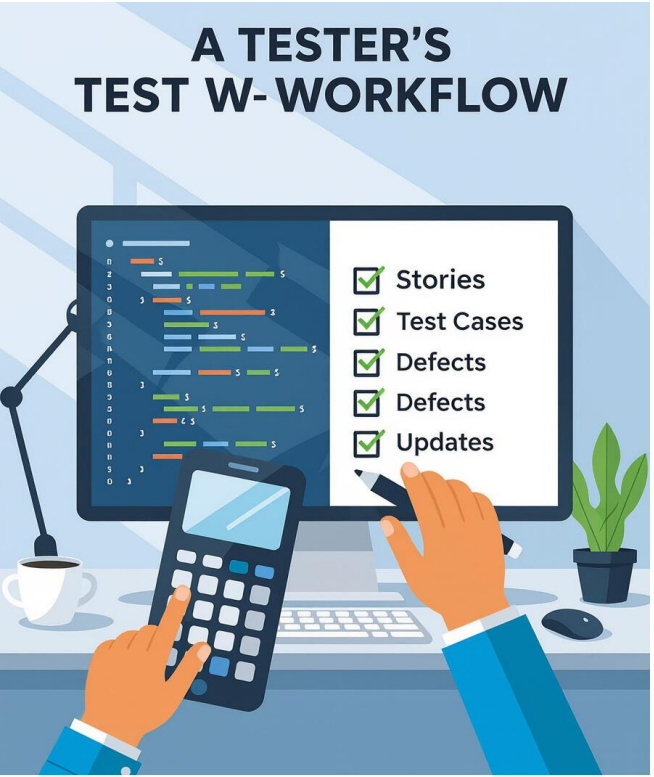
Going semantic: "Future" mode. Introduce AutomationIDs and semantic assertions—unique control identifiers and `assert.property` checks that make tests more resilient to UI changes.

Applications and replacements: What portions of our current acceptance testing can be automated and other potential uses for the toolkit.

Roadmap: Planned improvements and next steps (notes and feedback)

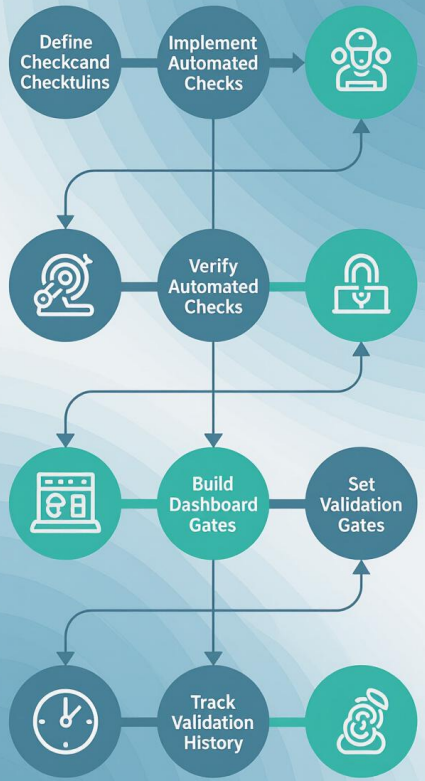


The Current State: Manual Testing in ENFIRE



Currently, ENFIRE testing relies on a manual workflow.

- Testers follow predefined procedures documented in **Excel spreadsheets**.
- These procedures detail specific actions to perform within the ENFIRE application.
- Testers manually record pass/fail outcomes based on visual inspection and expected results.
- This process is **labor-intensive**, **time-consuming**, and **prone to human error**.



Motivation for Automation

Manual testing presents several challenges:

- **Slow**: Each test cycle requires significant manual effort.
- **Error-Prone**: Manual execution increases the risk of inconsistencies and overlooked defects.
- **Not Scalable**: Expanding test coverage becomes increasingly difficult and costly.

These limitations motivate the need for an automated testing framework to enhance efficiency and reliability.

First Iteration (“Today” mode): Coordinate & Screenshot Approach

My initial automation effort involved:

- Automating user interactions based on screen coordinates.
- Using screenshot comparison to validate expected results.
- A test automation tool was developed to record and replay user actions.

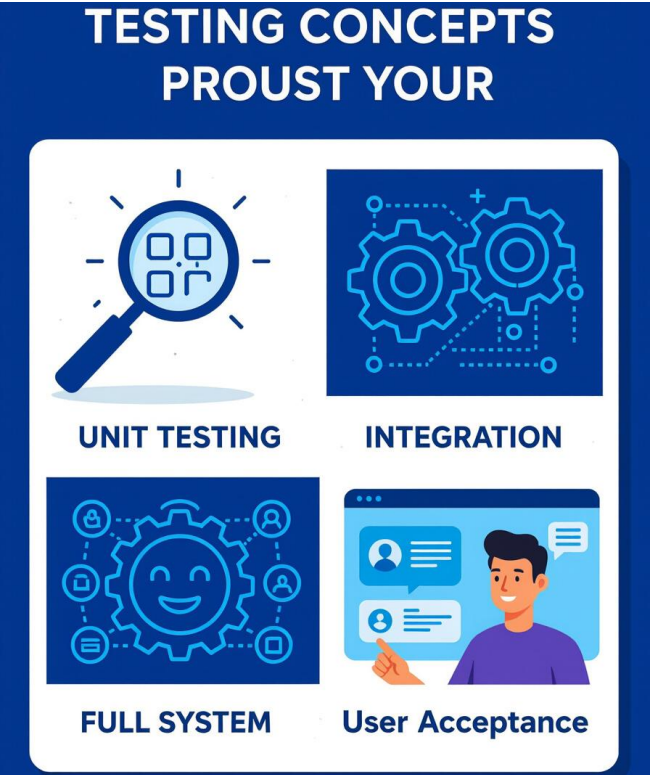
While this approach provided some initial benefits, it also revealed limitations:

Fragility: UI changes break coordinate-based scripts.

Limited Scope: Validating complex scenarios is difficult.

Maintenance Overhead: Script maintenance becomes cumbersome.

Demonstration: Coordinate-Based Automation in Action



demonstrate the coordinate-based automation:

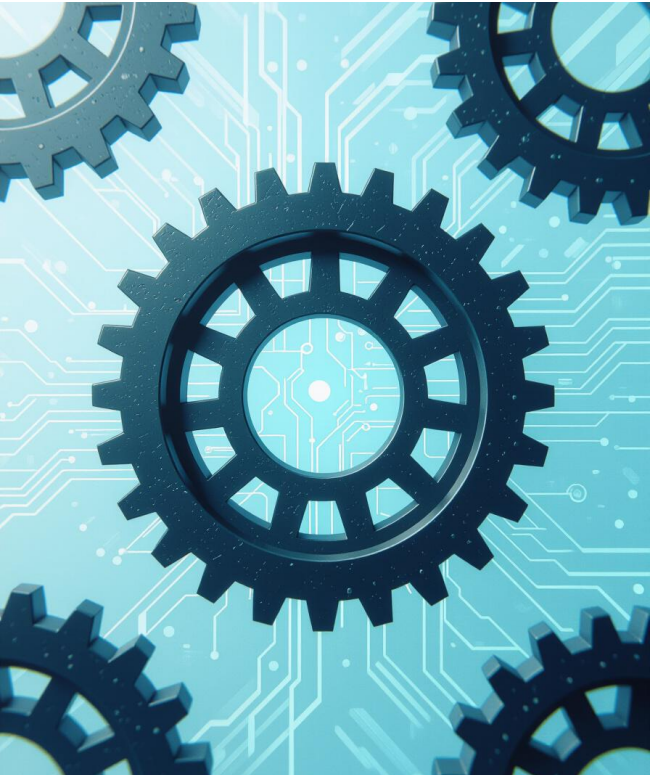
- record a simple test case in **ENFIRE**.
- Replay the recorded test.
- Observe the screenshot comparison results.
- Discuss the benefits of automating the manual process.
- We'll also address the challenges of maintaining coordinate-based scripts.

“Future” mode: AutomationIDs and Semantic Testing

To overcome the limitations of the coordinate-based approach, I would like to transition to semantic testing using AutomationIDs and metadata.

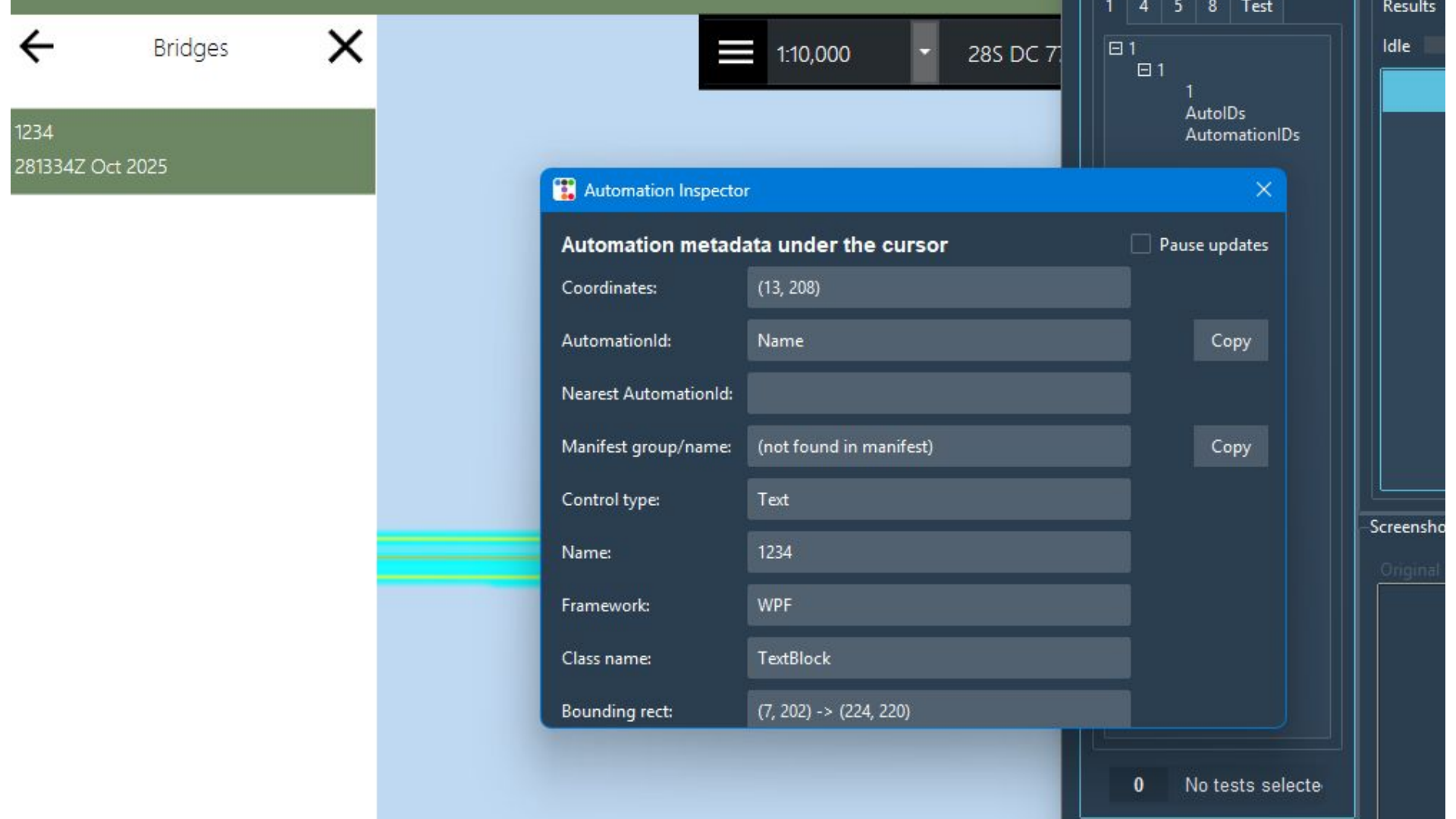
- **AutomationIDs:** Unique identifiers assigned to **UI elements**.
- **Semantic Validation:** Validating **UI element properties** and states based on **AutomationIDs**.
    - assert.property('SaveButton','IsEnabled',true)
- This approach provides a more robust and maintainable automation solution.

Benefits of the AutomatedID/Semantic Approach



Semantic automation offers several advantages:

- **Increased Reliability: UI** changes have less impact on scripts due to **AutomationID** stability.
- **Improved Maintainability:** Easier to update and maintain tests with semantic validation.
- **Enhanced Accuracy:** More precise validation of **UI element properties** and states.
- **Industry Standards:** Most industries use a hybrid of manual and automated testing for better coverage, efficiency, and reliability


Bridges
1234
281334Z Oct 2025
1:10,000
28S DC 7
1 4 5 8 Test
Results
Idle
1
1
1
AutoIDs
AutomationIDs
Automation Inspector
Automation metadata under the cursor
Pause updates
Coordinates:
(13, 208)
AutomationId:
Name
Copy
Nearest AutomationId:
Manifest group/name:
(not found in manifest)
Copy
Control type:
Text
Name:
1234
Framework:
WPF
Class name:
TextBlock
Bounding rect:
(7, 202) -> (224, 220)
0
No tests selecte
Screensho
Original

Roadmap & Future Vision

roadmap includes:

- **Expanding AutomationID coverage** across ENFIRE.
- **Gradually building out/replacing coordinate-based scripts** with semantic tests.
- **Integrating richer metadata and contextual validation**.
- **Incorporating CI/CD reporting** for continuous feedback.

The vision is a fully automated testing pipeline that ensures high-quality ENFIRE releases.

Potential Broader Applications

The ENFIRE UI Testing Toolkit has potential applications beyond ENFIRE:

- The **framework** can be adapted for other testing purposes.
- **Reusable components and libraries** can be leveraged across projects.
- This **approach promotes consistency and efficiency** in UI testing across ENFIRE.

Conclusion/notes

Demo the program

potential uses (part of pipeline? image testing?)

what it can replace (can we use this instead of the weeklong acceptance test? or maybe a portion of it for things it can reliably test?

improvements? (dockerize to reset enfire state between tests. Instead of time delays, wait for expected UI states)

automationIDs and metadata in ENFIRE to not have to use coordinates on screen or screenshots etc.