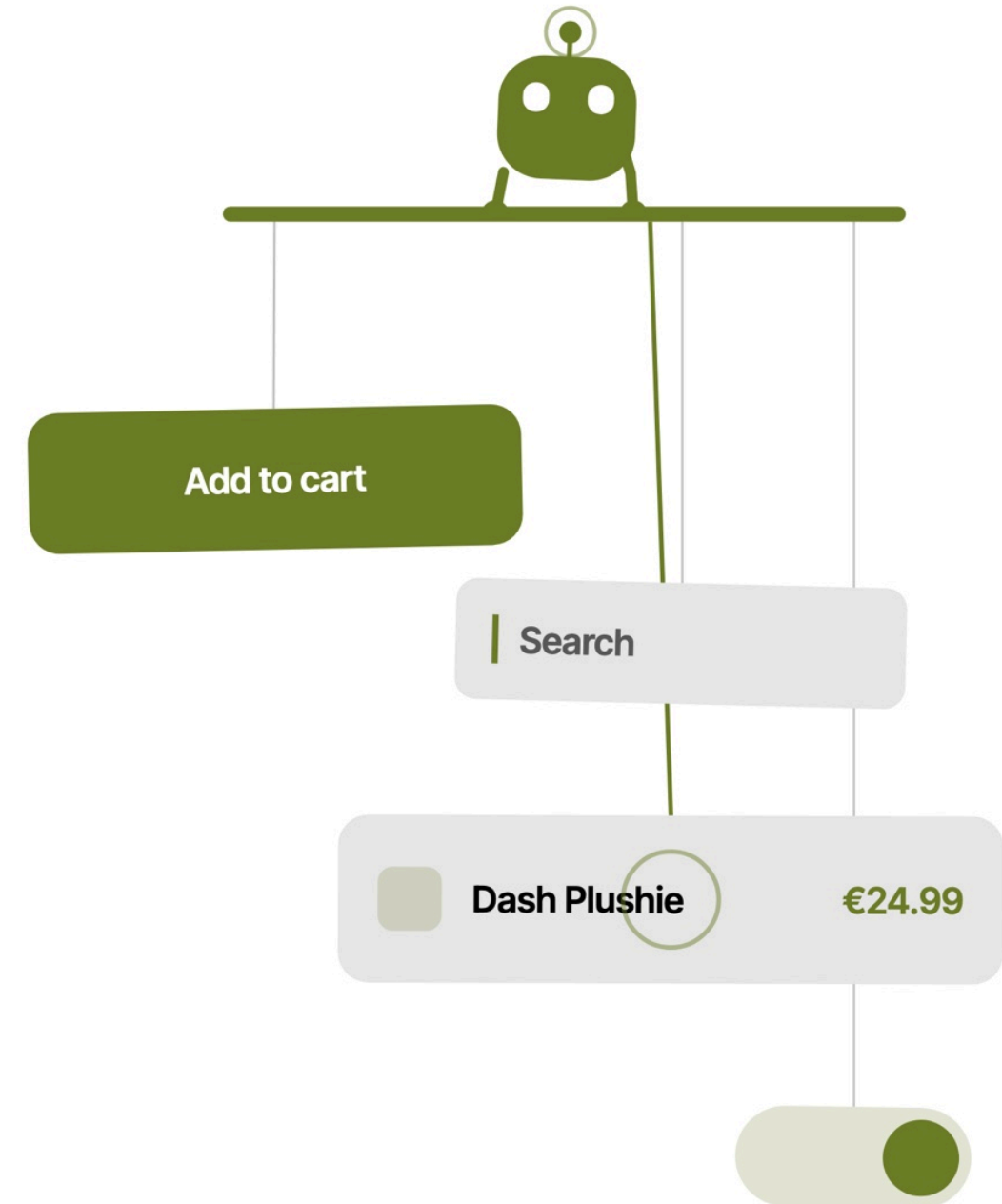


The AI that uses your app **for you**

Agentic Testing on Mobile

Gabriel Palmisano



Gabriel Palmisano



01 Mobile Developer at EBWorld

I've worked on EBWorld's mobile apps for over four years, and today I support the team building products with AI.

02 Maintainer of MapLibre Flutter

I'm fascinated by the world of maps and I like pushing the limits of geospatial programming.

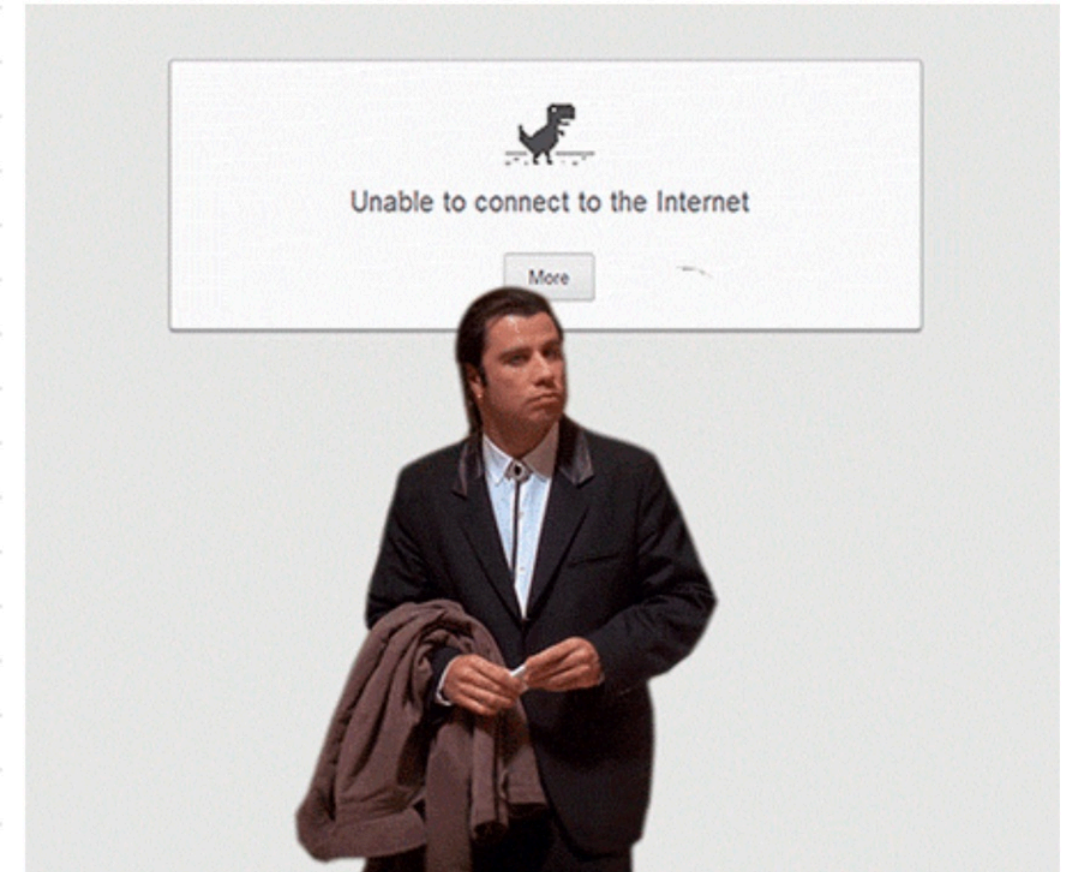
03 Active in the community

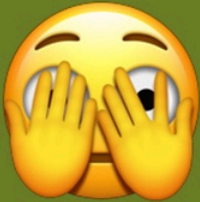
I take part in tech events in Italy and abroad, sharing experiences and learning from other developers.

Have you **ever** written automated tests?

If yes, raise your hand 🙋

If not, don't worry.
We hadn't either!



We can explain 

It all started with **a need**

- **Improve** the tests inside the app
- And we wanted to **automate** them
- We started an **R&D** activity



And we hit **three walls**

01

Complex setups

And hundreds of lines of mocks.



02

Fragile finders

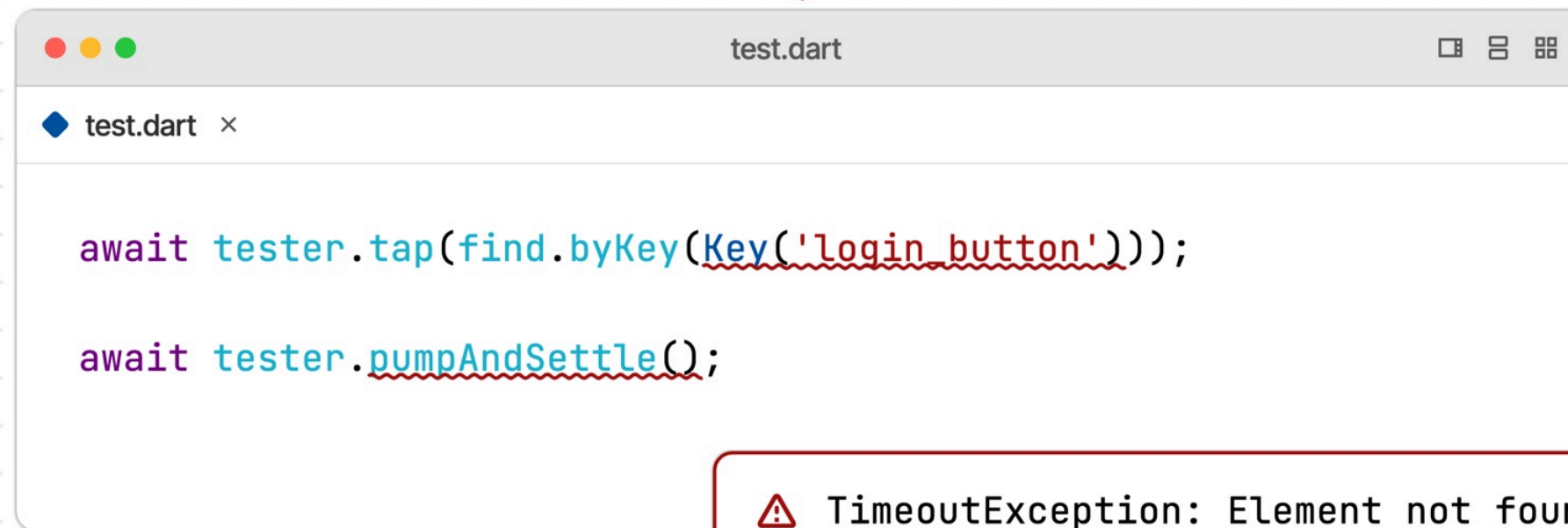
Anchored to the widget tree: they break at the first UI refactoring.



03

The CI paradox

Tests pass locally but fail in the pipeline.



```
await tester.tap(find.byKey(Key('login_button')));  
await tester.pumpAndSettle();
```

 TimeoutException: Element not found

How much **vocabulary** a simple test takes

```
import 'package:flutter_test/flutter_test.dart';
import 'package:integration_test/integration_test.dart';
import 'package:my_app/main.dart' as app;

void main() {
  IntegrationTestWidgetsFlutterBinding.ensureInitialized();

  testWidgets('add product to cart and verify total',
    (WidgetTester tester) async {
      // Launch the app
      app.main();
      await tester.pumpAndSettle();

      // Navigate to products and add first item
      await tester.tap(find.text('Products'));
      await tester.pumpAndSettle();
      await tester.tap(find.byIcon(Icons.add_shopping_cart).first);
      await tester.pumpAndSettle();

      // Go to cart and verify
      await tester.tap(find.byIcon(Icons.shopping_cart));
      await tester.pumpAndSettle();
      expect(find.text('Total: €29.99'), findsOneWidget);
    });
}
```

adapt

IntegrationTestWidgetsFlutterBinding

mock

setup

TestWidgetsFlutterBinding

pumpAndSettle

pump

finder

tap

assert

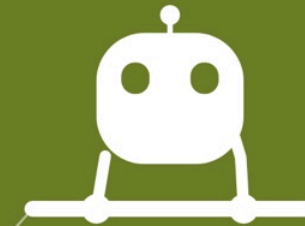
expect

equals



Hold on Hold on Hold on

**What if instead of
writing tests, you could
simply talk to your app?**



Go to the Products screen, add the first product and tell me if the total is there

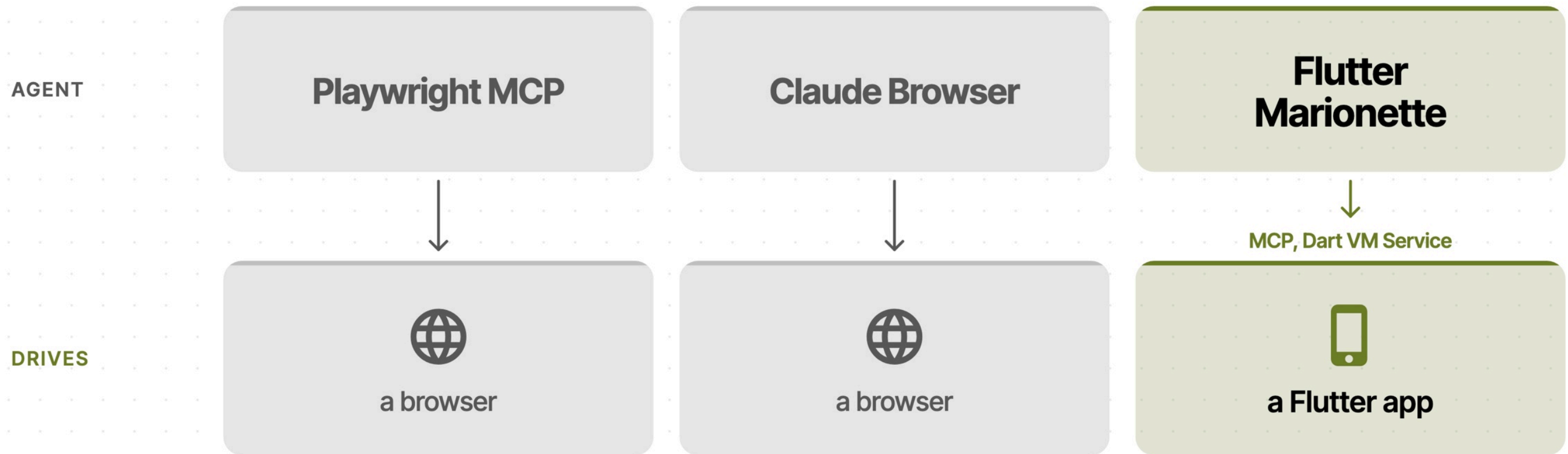


Flutter Marionette takes the stage



The Playwright MCP, but **for Flutter**

Built by LeanCode, the creators of Patrol



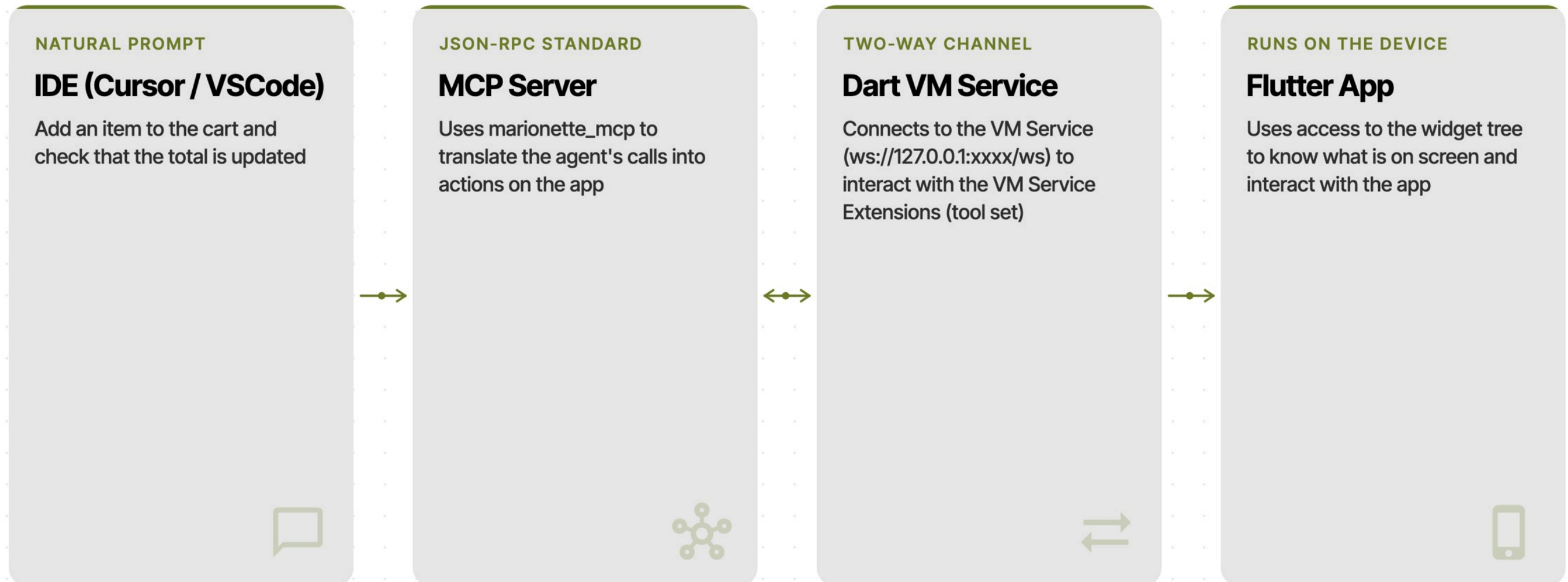
IN COMMON

All three read the element tree while the app runs, not its code.

The AI inspects, taps, scrolls and captures screenshots of your app, **in debug mode**

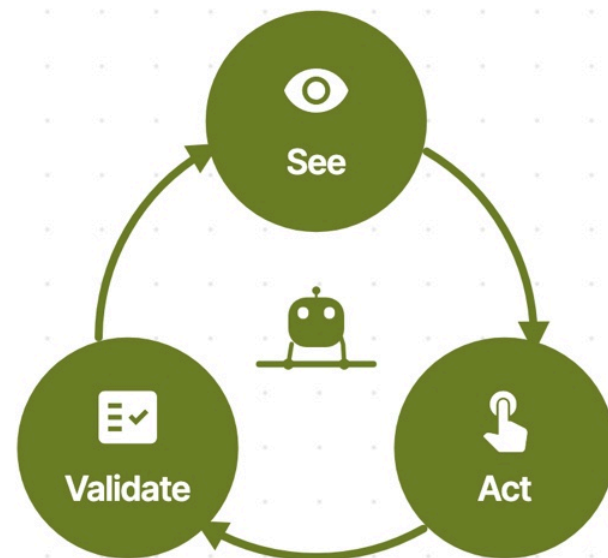
A **bridge** between the AI and the app in debug

It creates extensions in Flutter's debugger for each tool at its disposal, and uses MCP to interact with the app following the developer's instructions.



See, act, validate

It's the loop a human tester runs. Marionette repeats it at every iteration, until the goal is reached.



1 See

Inspection: it looks at the widget tree and works out which components are interactive

`get_interactive_elements``take_screenshots`

2 Act

Interaction: once it has found the elements to work with, it can tap or fill in text on its own

`tap``enter_text``scroll_to``hot_reload`

3 Validate

Feedback: it gives a detailed summary of the operations performed. Flags any regressions

`get_logs``Fix`

From **a sentence** to an action on the device

A dictionary of the main tools exposed and how they work.

REQUEST

→ MCP COMMAND

→ ACTION ON THE DART VM

What's on the screen?

get_interactive_elements

take_screenshot

Extracts the filtered widget tree or captures the screen

Open the products screen

tap(text: 'Products')

Performs a pointer action

Search for a product

enter_text(key: 'pen')

Injects strings into the TextField

Is the total correct?

get_logs

take_screenshot

Reads the logs and the image for validation

Setup in 2 minutes

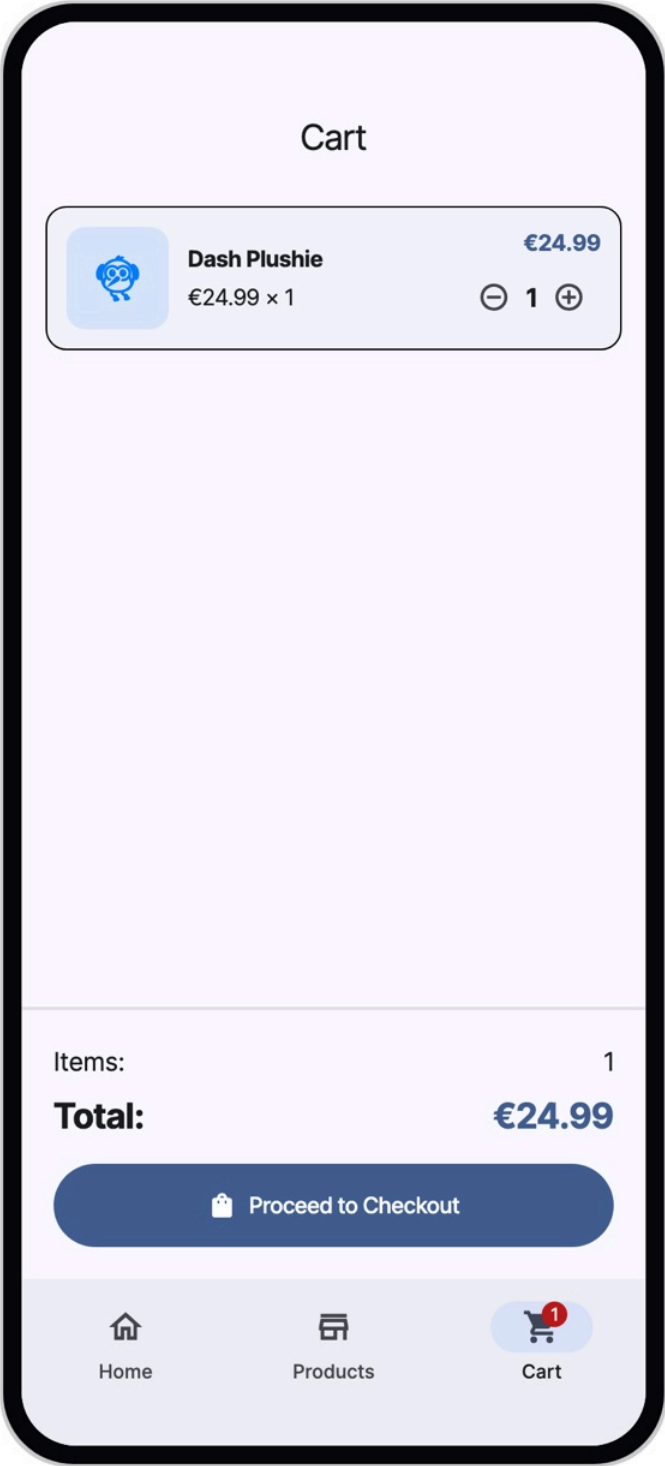
A few lines are enough, and it hooks into the active debugger.

TERMINAL

```
dart pub global activate marionette_cli  
  
dart run marionette_mcp
```

MAIN.DART

```
void main() {  
  MarionetteBinding.ensureInitialized();  
  // The only binding required  
  runApp(const MyApp());  
}
```



 ● claude code idle

stopclear

actOK

tap

seeok

get_interactive_elements

validateok

get_logs

Total is correct.

I opened the Products screen and added the first item in the grid, Dash Plushie, to the cart. In the cart the line shows €24.99 × 1 and the total reads €24.99.

STEPS 4>

Item	Value
Product	Dash Plushie
Cart line	€24.99 × 1
Total	€24.99

Go to the Products screen, add the first product to the cart and tell me if the total is correct.

send

Two **different** tools, not one better than the other

Probabilistic or deterministic? Let's find out together which tool fits best.

	Marionette MCP	Patrol or flutter_test
GOAL	Explore and test the app at runtime	Formal testing in CI/CD
FEATURES	Flexible, tolerates UI changes	Rigid, exact assertions, finder anchored to the widget tree
USAGE	Quick iteration, visual sanity checks	Core regressions, release runs, critical test suites
LIMITS	No CI/CD, no release mode. A widget with no key and no text stays invisible	Complex setup, constant maintenance, fragile to UI refactoring

IDEAL FLOW

- 1 Use Marionette to test the feature on the device while you build it.
- 2 Iterate quickly with smoke and visual sanity checks
- 3 Write tests for CI/CD to improve pre-production testing

From copilot to **autonomous agent**

An experiment or the right direction?

Phase 01

Co-Pilot (Passive)

Autocomplete, suggestions. The human drives, the AI suggests.



Phase 02

Generator

Code and feature generation. The AI writes, the human validates.



Phase 03

Autonomous Agent

Generation, interaction and validation. The AI writes and verifies its own work.



Automated validation could become **the safety net** that lets Agentic Coding scale.

Stop wrestling with manual tests. Start **talking** to your app.

Gabriel Palmisano

g.palmisano@ebw.it



[leancodepl/marionette_mcp](https://github.com/leancodepl/marionette_mcp)



My contacts

Questions?

Gabriel Palmisano

g.palmisano@ebw.it



My contacts