



"IS IT A BIRD?
IS IT A PLANE?"

THE WEIRD SHAPES WE USE
TO TALK ABOUT TESTING

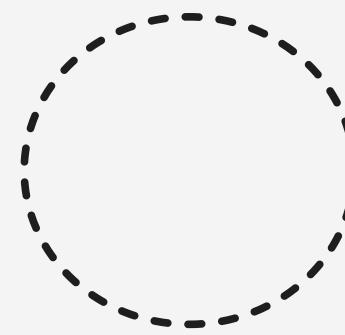
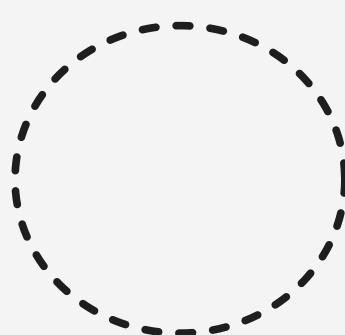
PART II

Common testing
methodologies

THE DIFFERENT TYPES OF TESTS

THE DIFFERENT TYPES OF TESTS

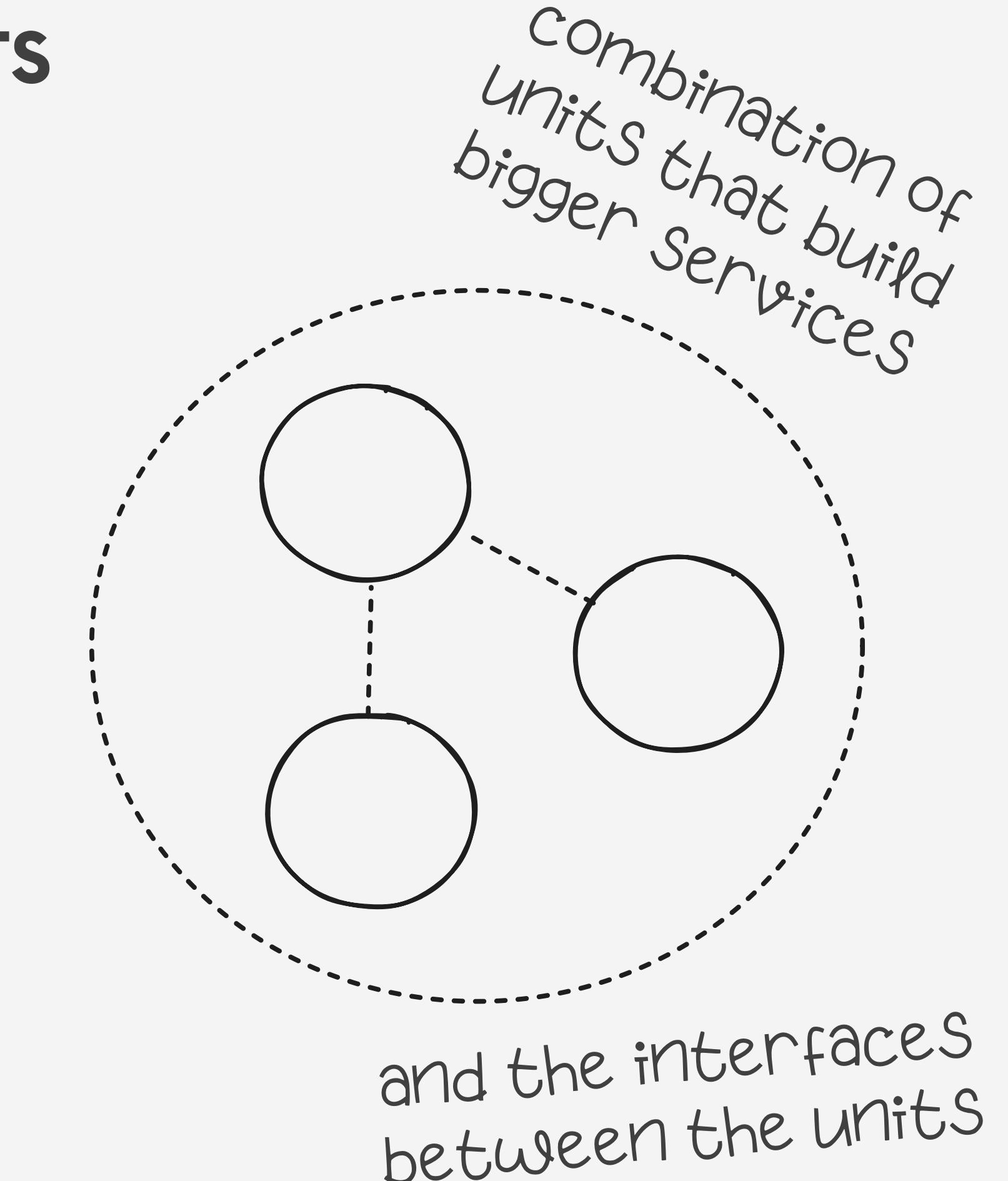
1. Unit tests



your smallest
units of code

THE DIFFERENT TYPES OF TESTS

1. Unit tests
2. Integration tests



Unit test vs. Integration test

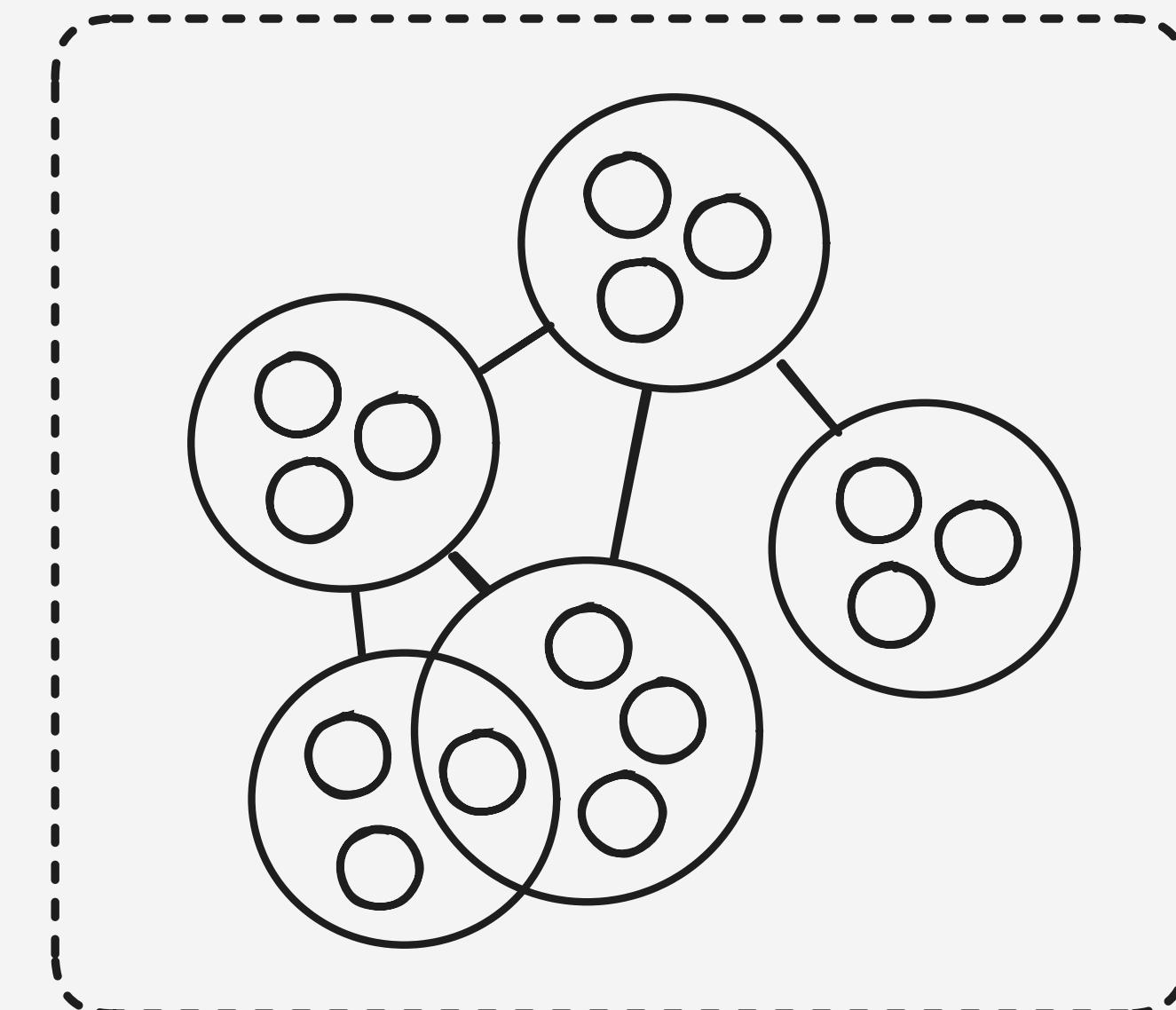


https://www.reddit.com/r/ProgrammerHumor/comments/isidkn/unit_testing_vs_integration_testing/

THE DIFFERENT TYPES OF TESTS

1. Unit tests
2. Integration tests
3. End-to-end tests

the end-to-end
behaviours of
the system



THE DIFFERENT TYPES OF TESTS

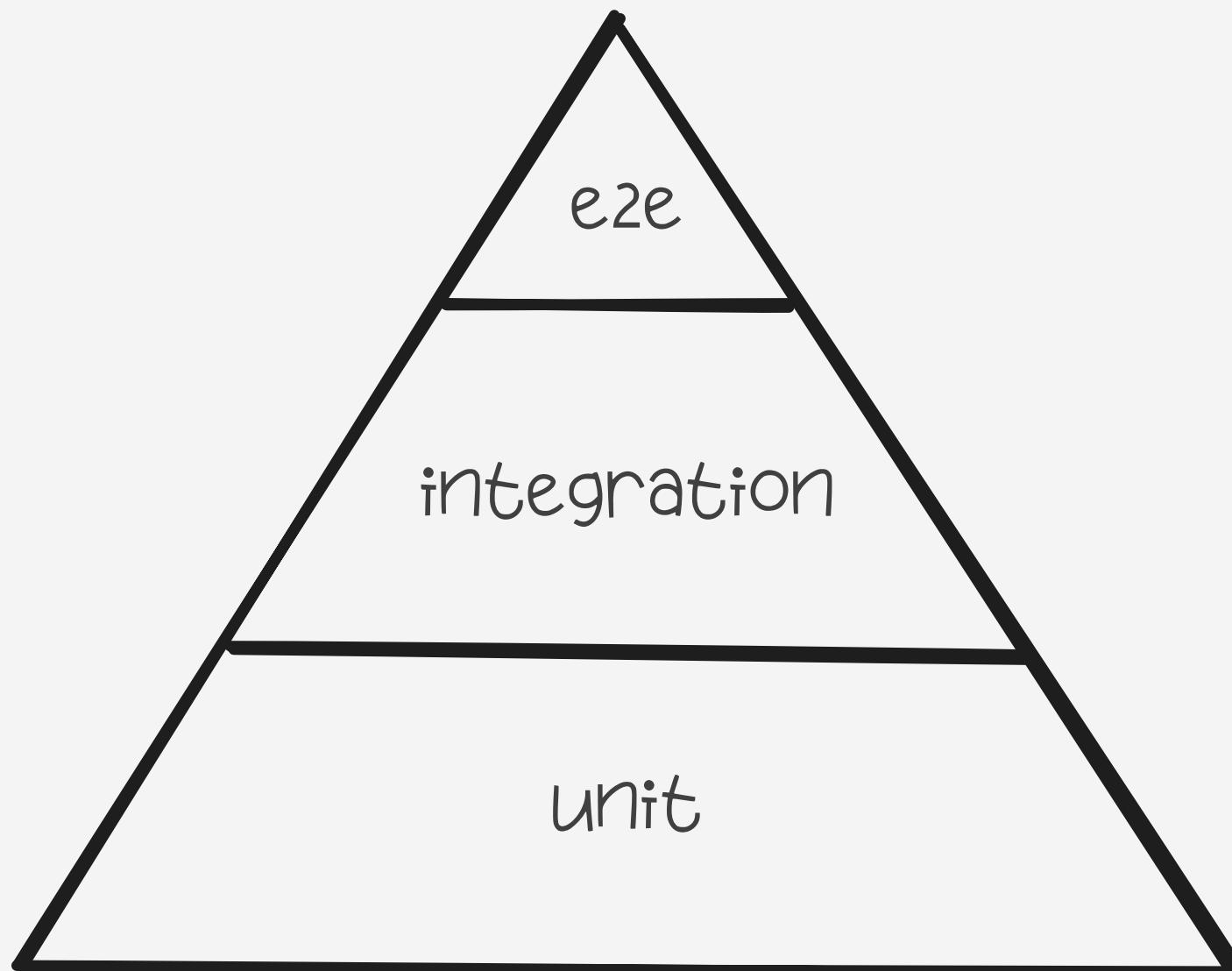
1. Static tests
2. Unit tests
3. Integration tests
4. End-to-end tests

THE DIFFERENT TYPES OF TESTS

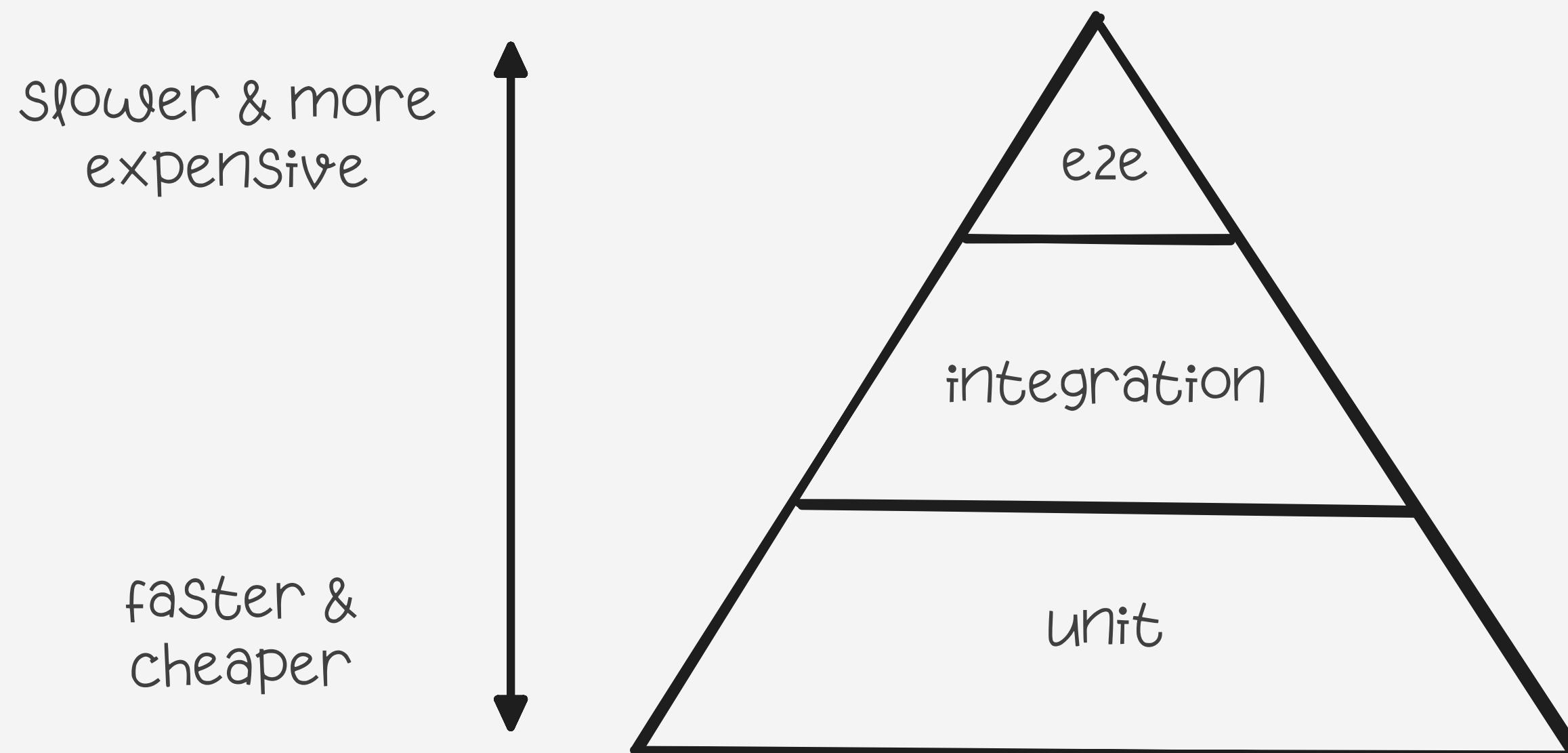
1. Static tests
2. Unit tests
3. Integration tests
4. End-to-end tests
5. Manual tests

“THE SHAPES”

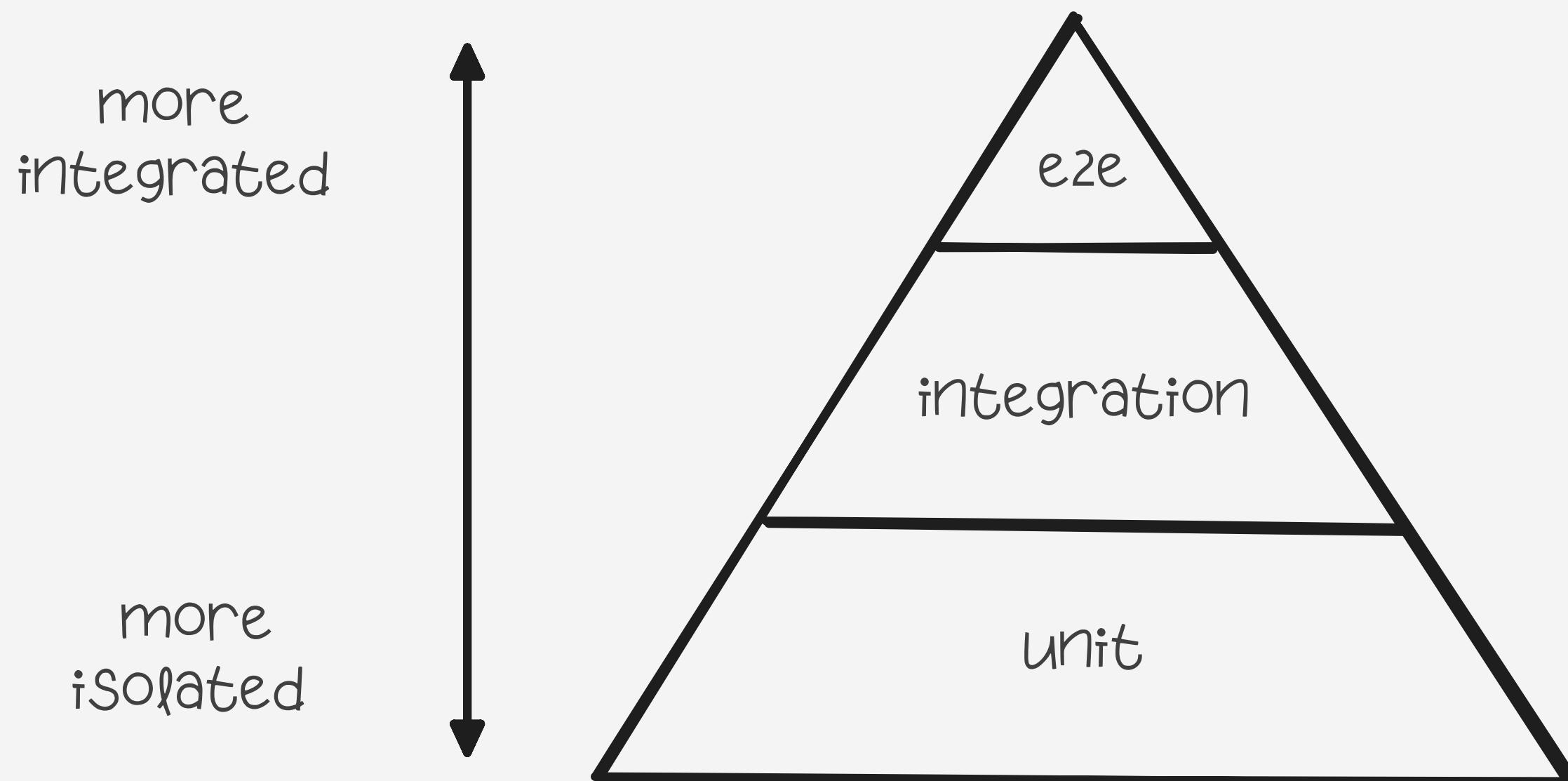
“THE SHAPES” — THE PYRAMID



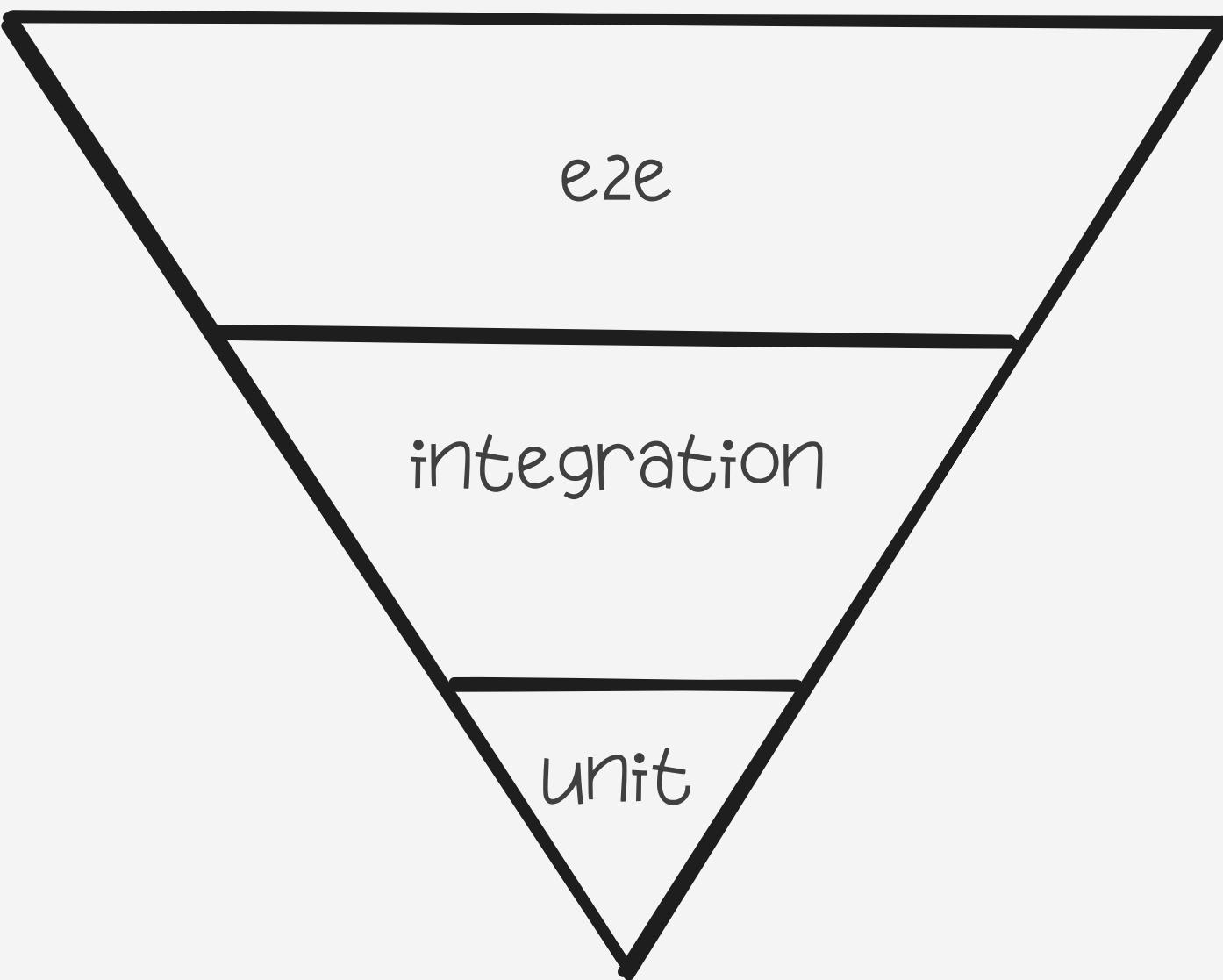
“THE SHAPES” — THE PYRAMID



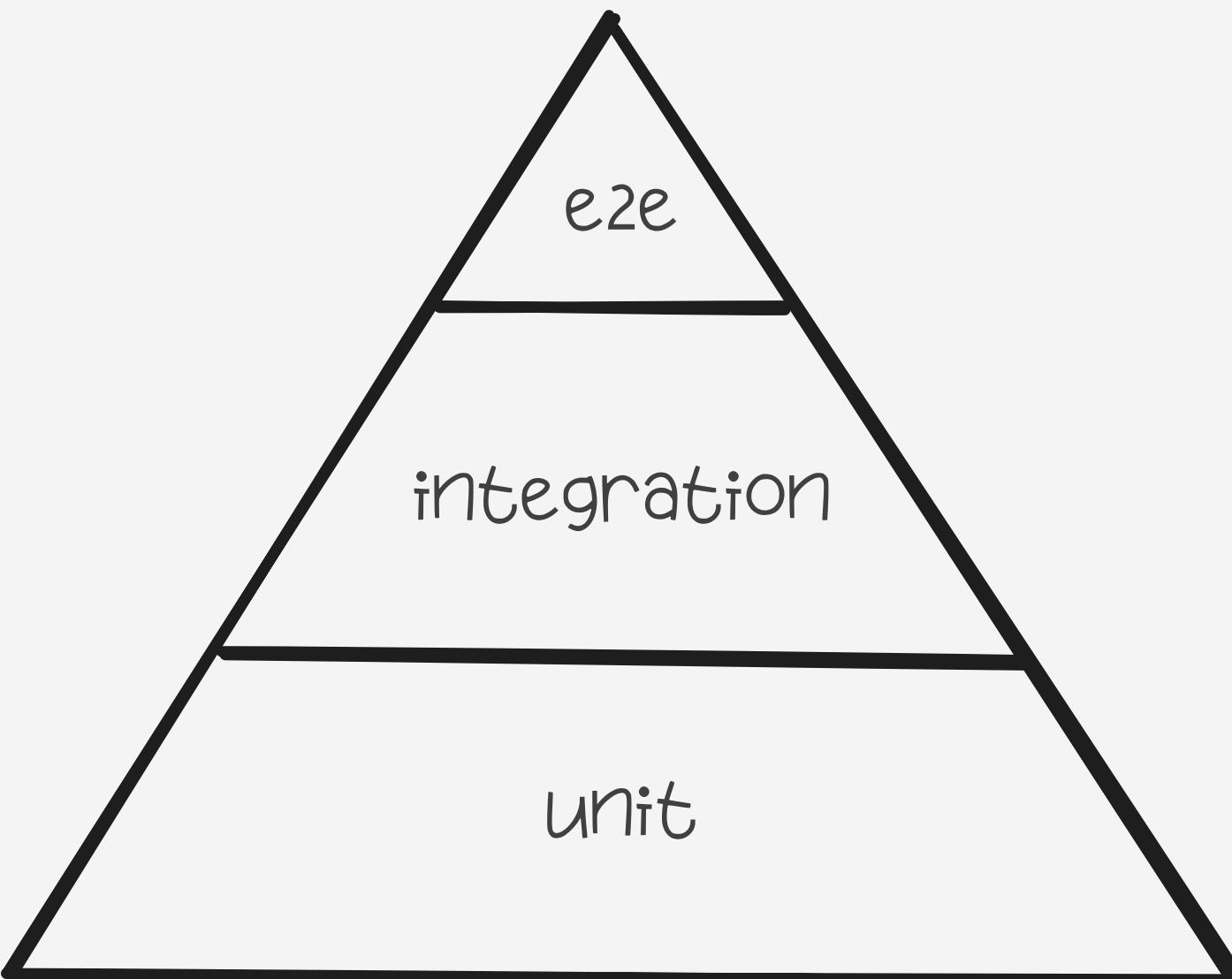
“THE SHAPES” — THE PYRAMID



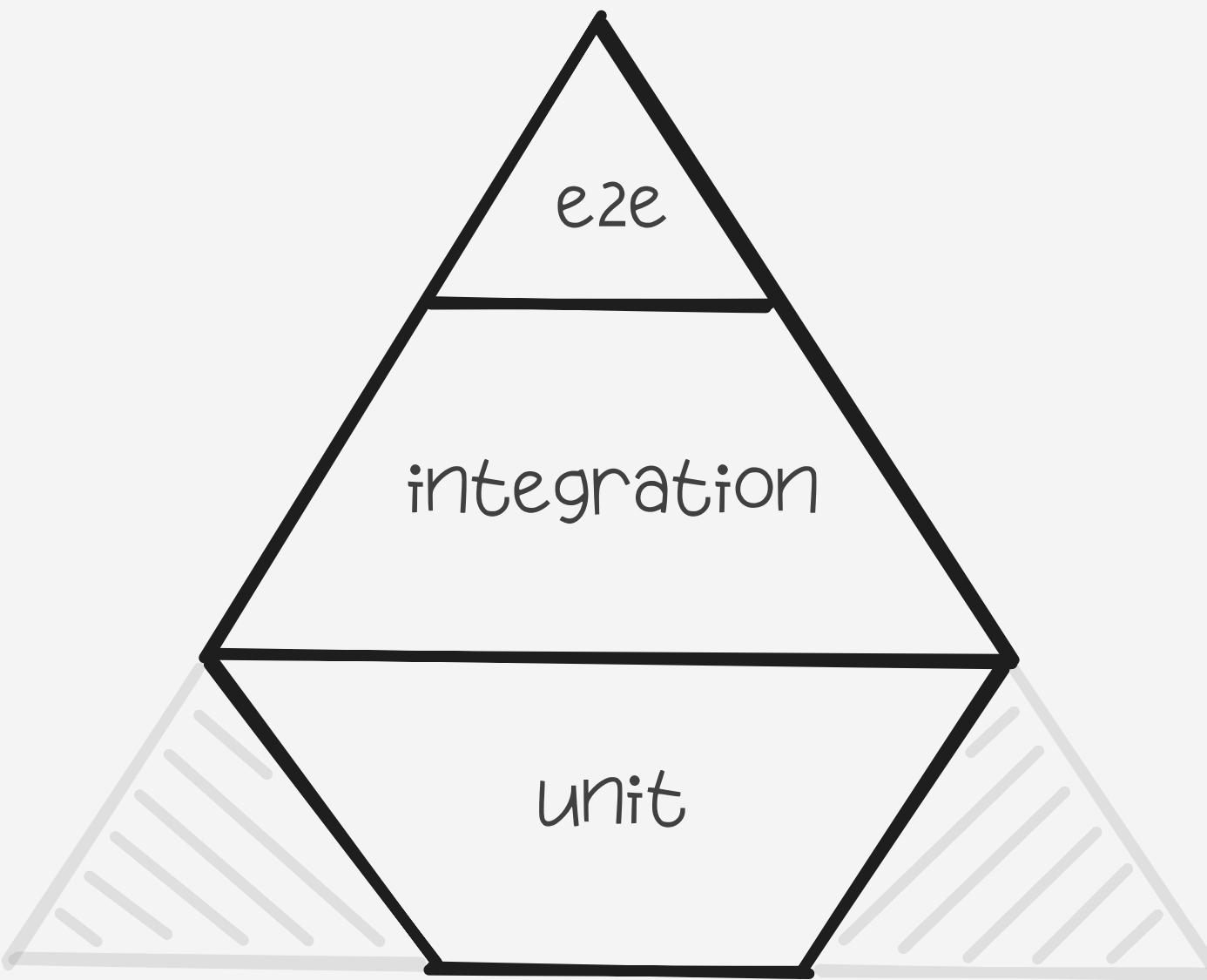
“THE SHAPES” — THE ICE CONE



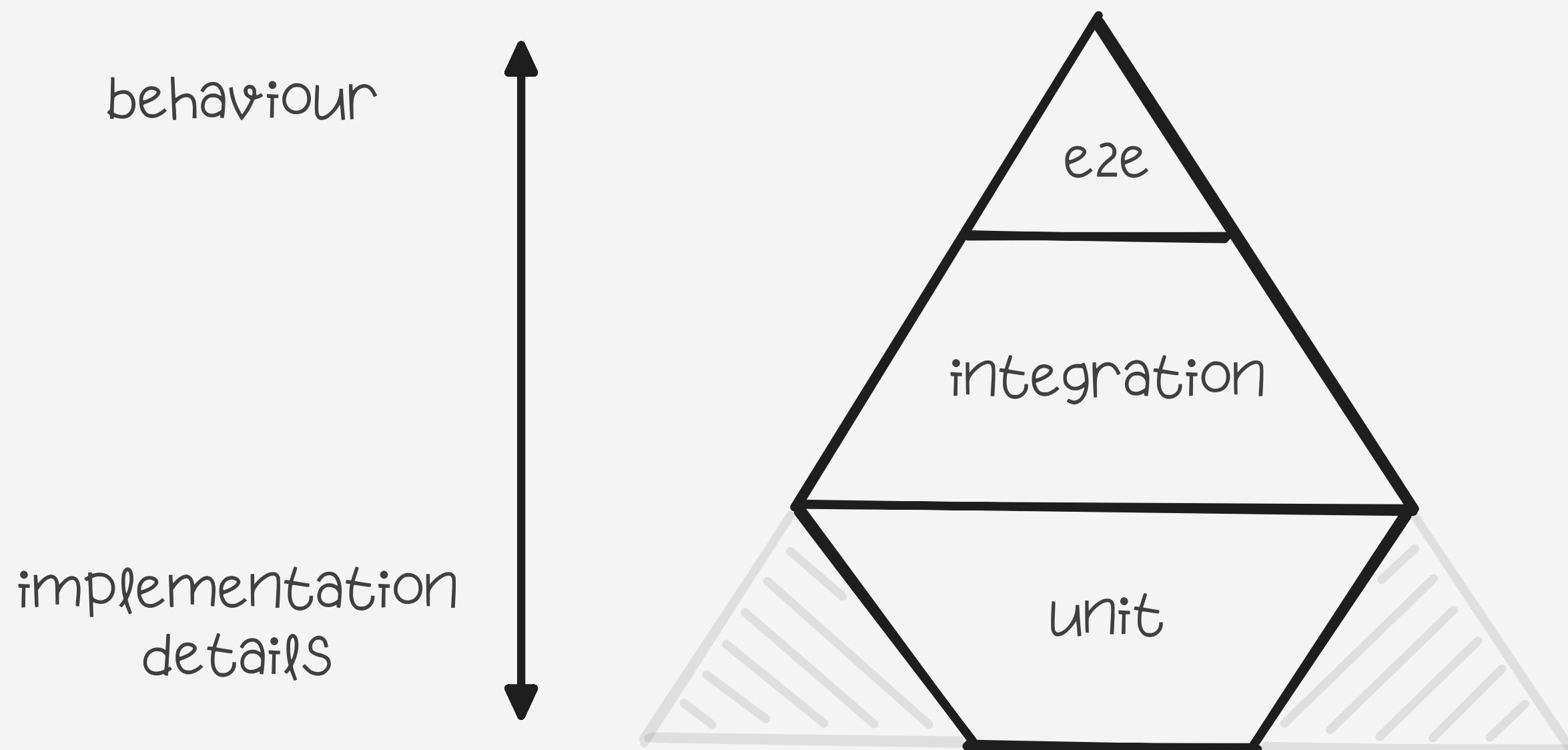
“THE SHAPES” — THE DIAMOND



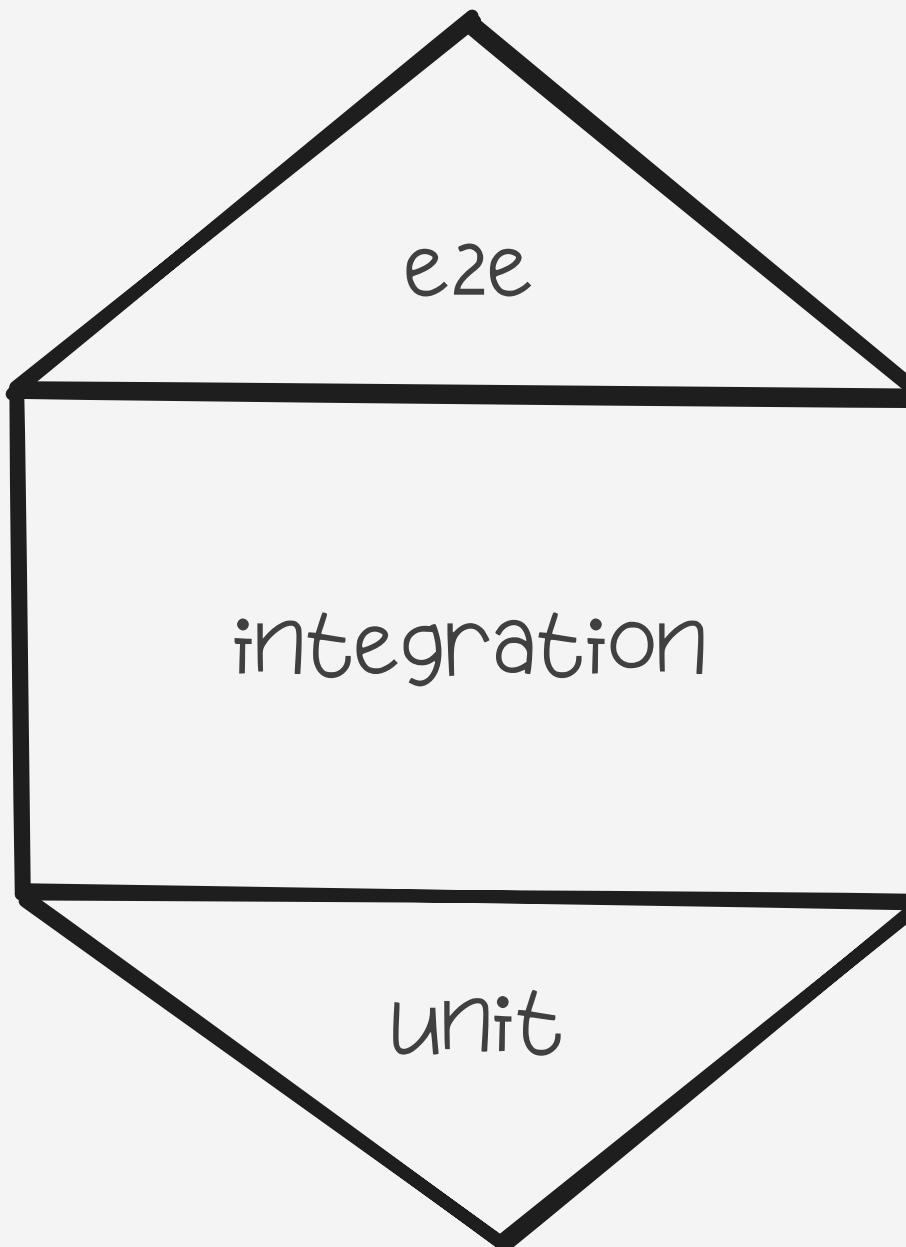
“THE SHAPES” — THE DIAMOND



“THE SHAPES” — THE DIAMOND



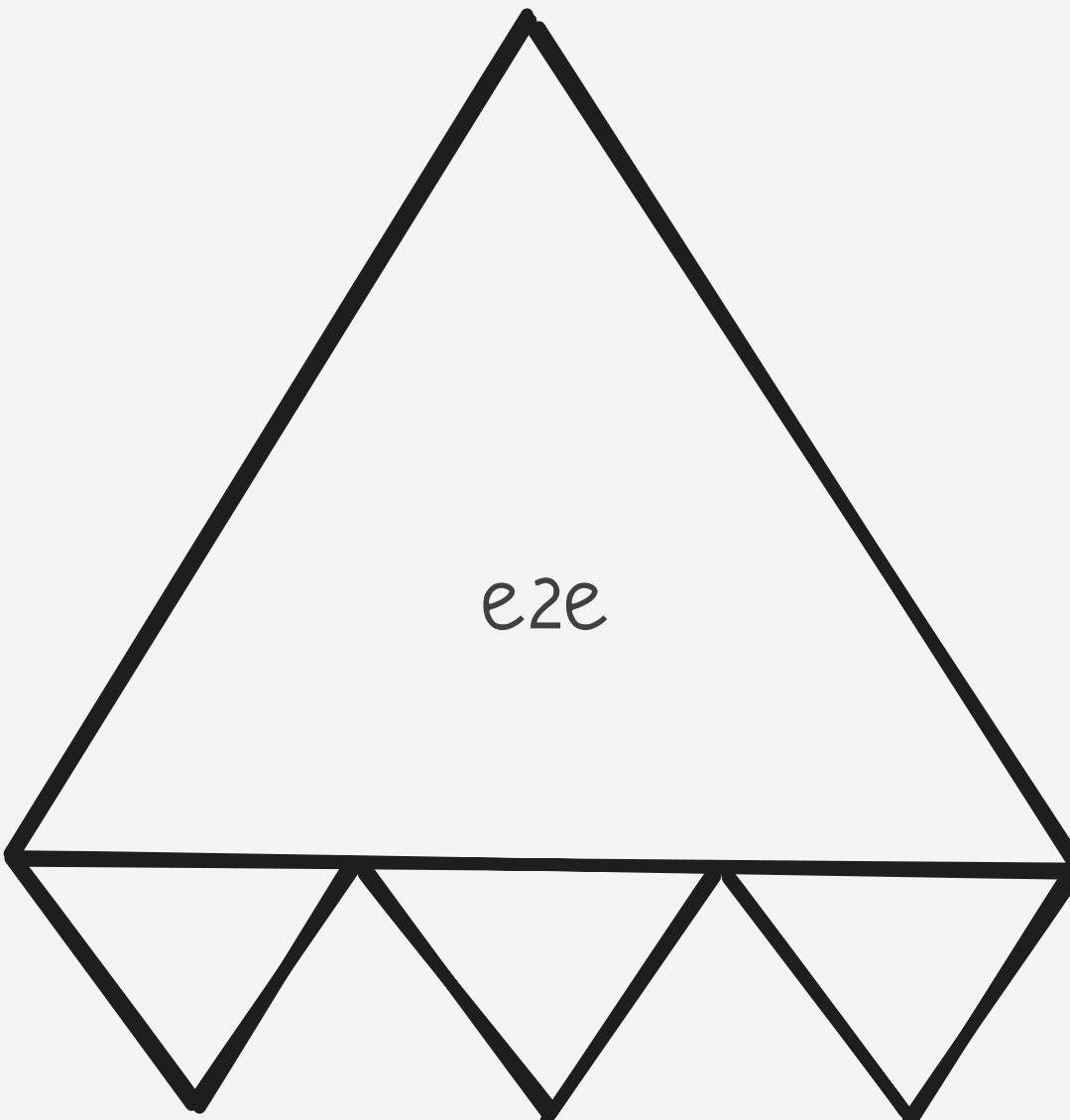
“THE SHAPES” — THE HONEYCOMB



**WE ARE ABLE TO REFACTOR THE INTERNALS
WITHOUT TOUCHING ANY TESTS. WE COULD
EVEN REPLACE THE DATABASE FROM
POSTGRESQL TO NOSQL WITHOUT HAVING TO
MODIFY THE ACTUAL TEST METHODS.**

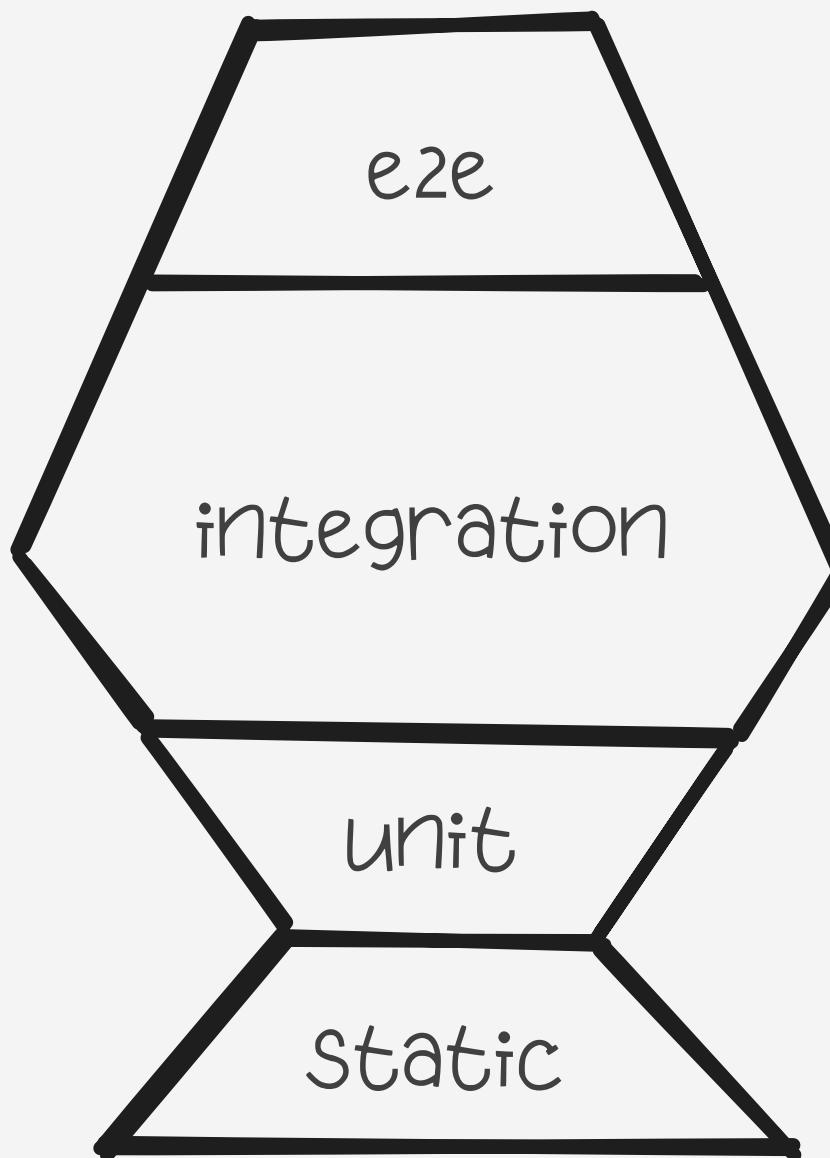
**THE TRADE-OFF HERE IS SOME LOSS OF SPEED
IN TEST EXECUTION [... AND ...] THAT WE
MIGHT LOSE SOME FEEDBACK ACCURACY
WHEN A TEST FAILS**

“THE SHAPES” — THE CRAB 🦀



integration &
unit tests

“THE SHAPES” — THE TROPHY





Justin Searls
@searls

...

I'm realizing I've never shared this publicly before, but I probably should: almost all the advice you hear about software testing is bad. It's either bad on its face or it leads to bad outcomes or it distracts by focusing on the wrong thing (usually tools). A brief thread on why



Justin Searls @searls · May 15, 2021

People love debating what percentage of which type of tests to write, but it's a distraction. Nearly zero teams write expressive tests that establish clear boundaries, run quickly & reliably, and only fail for useful reasons. Focus on that instead.

...

[Show more](#)

12:17 AM · May 16, 2021

<https://twitter.com/searls/status/1393571227650908162>



Justin Searls
@searls

Follow

...

People love debating what percentage of which type of tests to write, but it's a distraction. Nearly zero teams write expressive tests that establish clear boundaries, run quickly & reliably, and only fail for useful reasons. Focus on that instead.



swyx @swyx · May 15, 2020

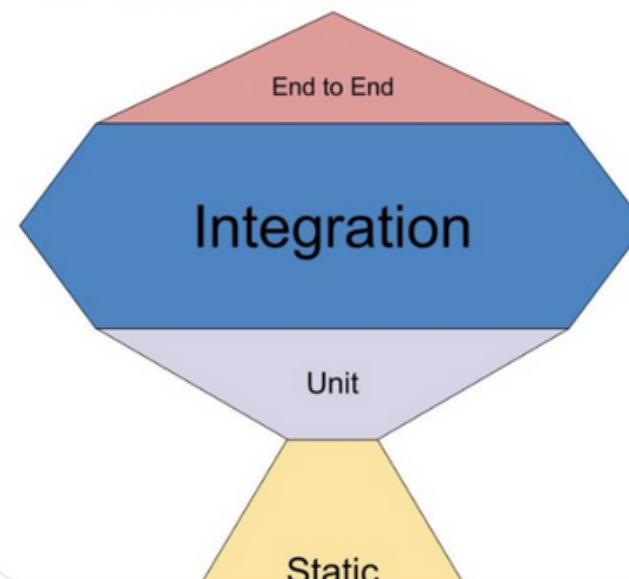
The @MartinFowler Test Pyramid has fallen out of style.
Integration > Unit tests is the new conventional wisdom.

In frontend, we now have the "Testing Trophy" from @rauchg and @kentcdodds....

Show more

A general guide for the **return on investment** 😊 of the different forms of testing with regards to testing JavaScript applications.

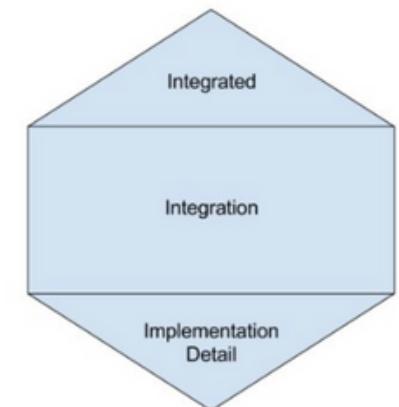
- End to end w/ [@Cypress.io](#) ●
- Integration & Unit w/ [@fbjest](#) 🎉
- Static w/ [@flowtype](#) F and [@geteslint](#) ●



y Labs

Microservices test strategy

way of structuring our tests for Microservices would be the Testing Honeycomb



Microservices Testing Honeycomb

We should focus on Integration Tests, have a few Implementation Detail Tests (ideally none).

11:58 AM · May 15, 2021

<https://twitter.com/searls/status/1393385209089990659>

PART II:

Why do we
write tests?

PART II:

Why do we not
write tests?

WHY DO WE NOT WRITE TESTS?

WHY DO WE NOT WRITE TESTS?

“We don’t have time for this”

WHY DO WE NOT WRITE TESTS?

“We don’t have time for this”

“Tests cause extra work; they start failing when changing unrelated code, are becoming brittle quickly”

WHY DO WE DO WRITE TESTS?

“To make our managers or team lead happy”

**ANY OBSERVED STATISTICAL REGULARITY WILL
TEND TO COLLAPSE ONCE PRESSURE IS
PLACED UPON IT FOR CONTROL PURPOSES**

— Goodhart's Law

WHY DO WE DO WRITE TESTS?

~~"To make our managers or team
lead happy"~~

WHY DO WE DO WRITE TESTS?

~~"To make our managers or team-
lead happy"~~

Stability for the end-user

SIMPLE TESTING CAN PREVENT MOST CRITICAL FAILURES

— Research paper from the University of Toronto (2014)

SIMPLE TESTING CAN PREVENT MOST CRITICAL FAILURES

— Research paper from the University of Toronto (2014)

“A majority of the production failures (77%) can be reproduced by a unit test.”

WHY DO WE DO WRITE TESTS?

~~"To make our managers or team-lead happy"~~

Stability for the end-user

Enable refactoring

**CODE IS READ AND UPDATED
WAY MORE OFTEN THAN IT
IS WRITTEN**

— Tim Bray

**“WALKING ON WATER AND DEVELOPING
SOFTWARE FROM A SPECIFICATION ARE
EASY IF BOTH ARE FROZEN”**

— Edward V. Berard

PART III:

TDD — The misunderstood monster under your bed

*a little detour,
Sorry!*

RED-GREEN-REFACTOR

RED-GREEN-REFACTOR

Write a test that describes the desired behaviour

RED-GREEN-REFACTOR

Write a test that describes the desired behaviour

Write the minimal amount of code necessary to make the test pass

RED-GREEN-REFACTOR

Write a test that describes the desired behaviour

Write the minimal amount of code necessary to make the test pass

Refactor the code

SELF-TESTING CODE



SELF-TESTING CODE

Tests should be part of your code delivery

SELF-TESTING CODE

Tests should be part of your code delivery

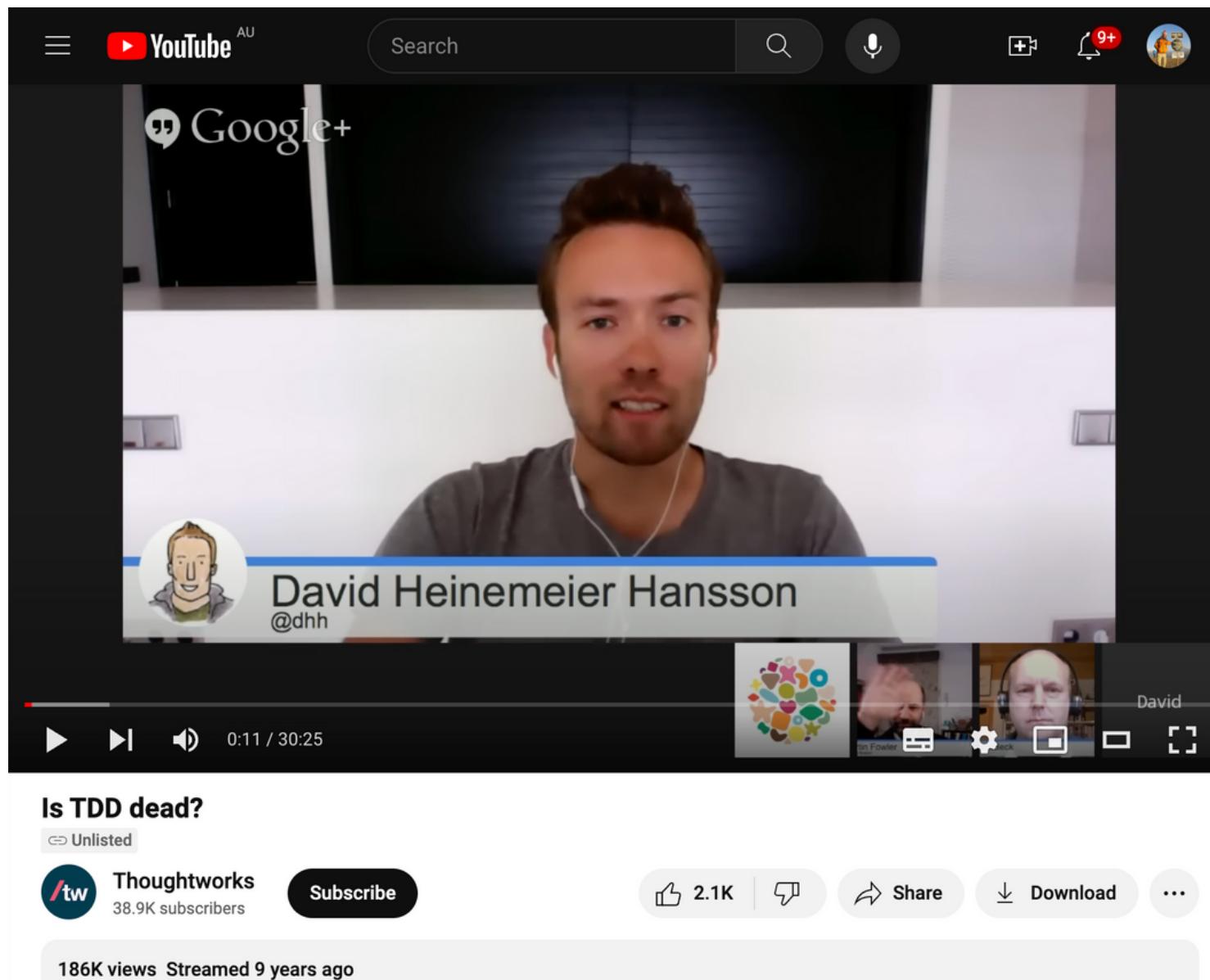
You should be running tests as often as possible — ideally automated

TDD — SOME MISCONCEPTIONS

TDD — SOME MISCONCEPTIONS

“TDD means I have to
write my tests first!”

IS TDD DEAD? — VIDEO SERIES WITH DHH, MARTIN FOWLER AND KENT BECK



<https://martinfowler.com/articles/is-tdd-dead/>

TDD — SOME MISCONCEPTIONS

“TDD says I have to
mock everything
outside of the unit
test scope”

"SIMILARLY THEIR NOTION OF INTEGRATION TEST SOUNDS VERY MUCH LIKE WHAT I WOULD CALL A SOCIABLE UNIT TEST. THIS MAKES THE PYRAMID VERSUS HONEYCOMB DISCUSSION MOOT"

— Martin Fowler

THE MORAL OF THE STORY

Practical tips to get
started with testing



1

Be the advocate!



2

**Embrace tests as a way
to improve developer
confidence**



3

**Test behaviour, not
implementation details**



4

Automate your tests!

5

**Identify the most
common or likely points
of failure in your system**

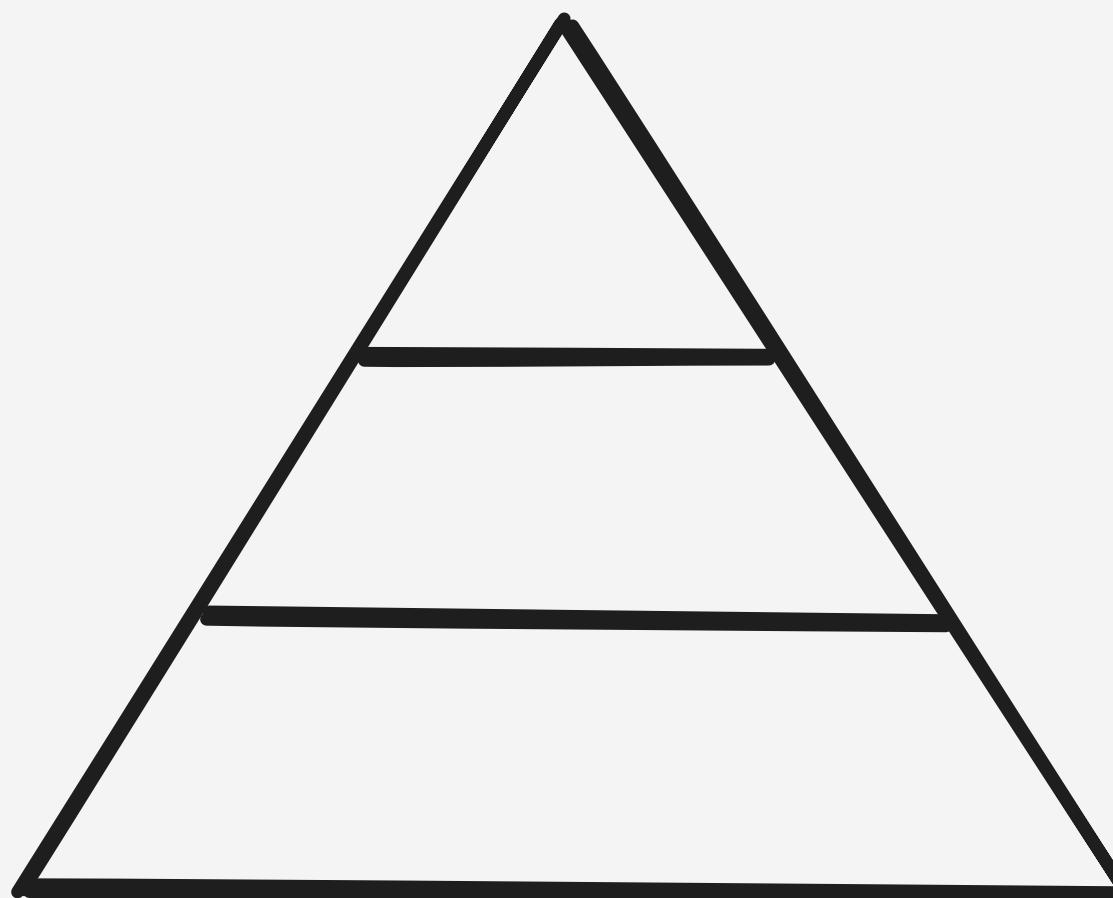


6

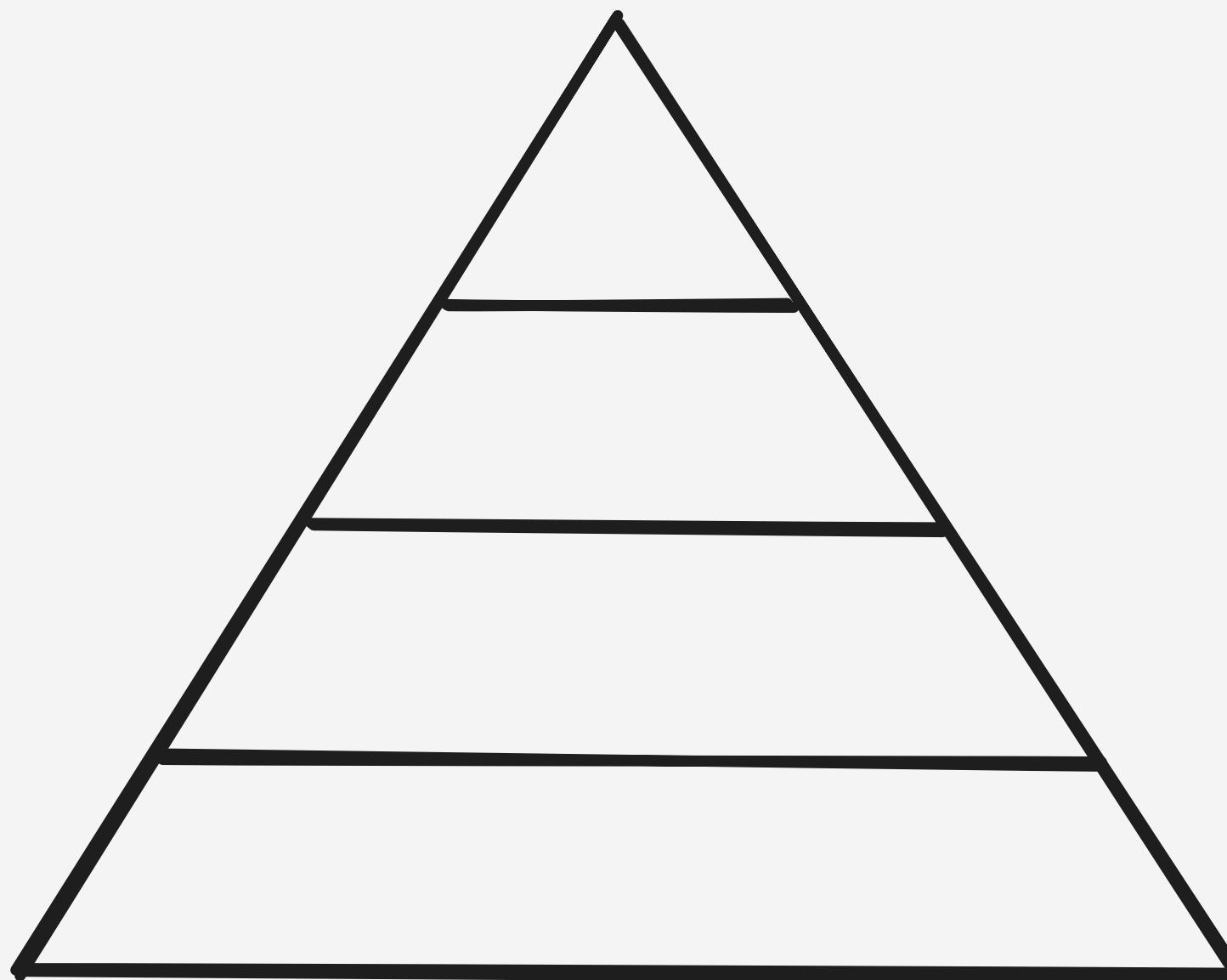
**Don't just blindly follow
a "shape" or strategy**



ENSURE YOUR “SHAPE” HAS INTENTION

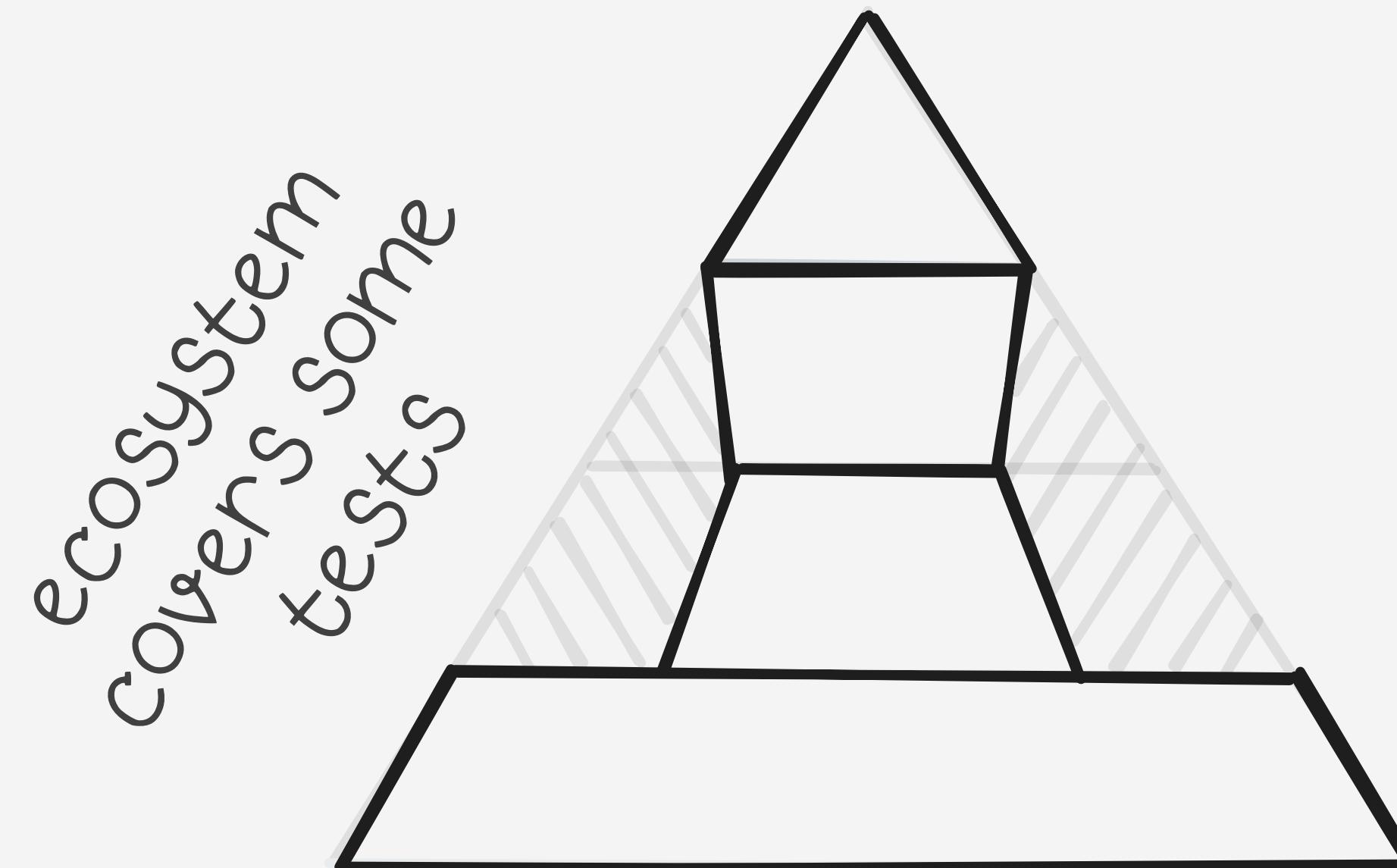


ENSURE YOUR “SHAPE” HAS INTENTION

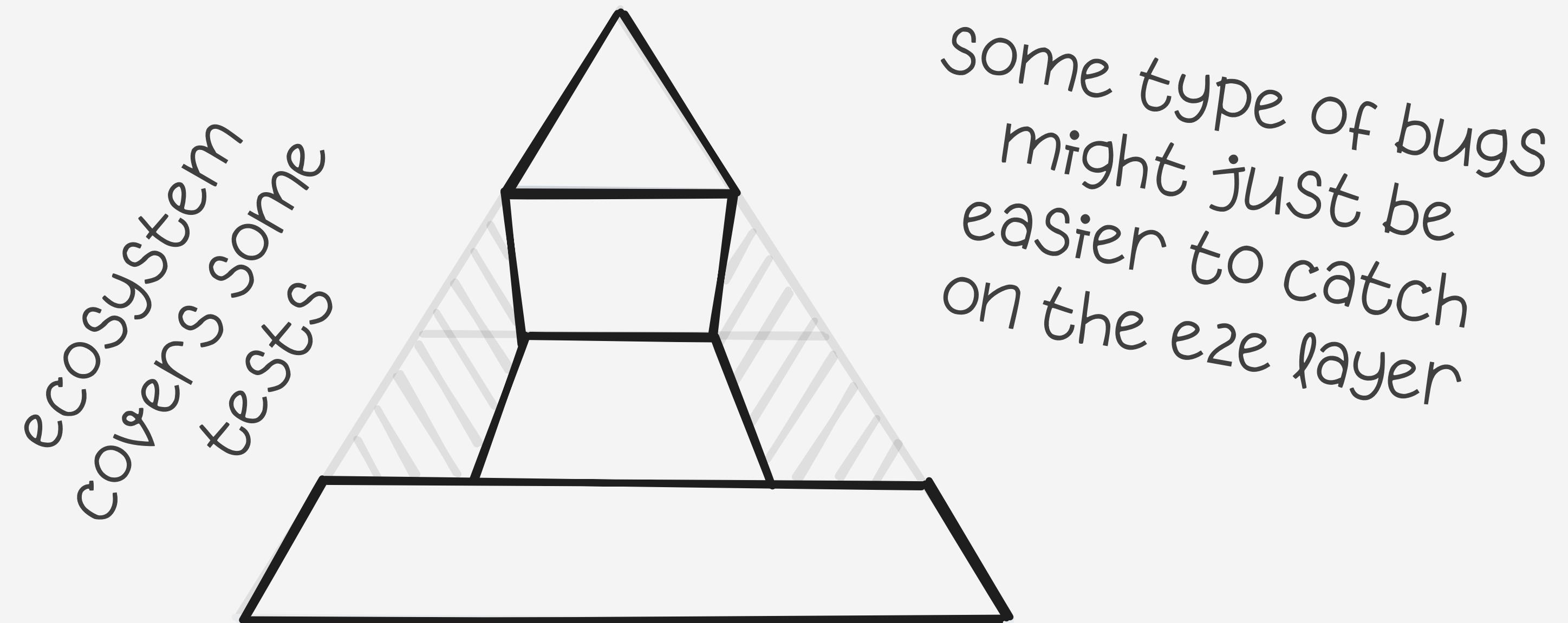


let's add the
static layer

ENSURE YOUR “SHAPE” HAS INTENTION



ENSURE YOUR “SHAPE” HAS INTENTION



7

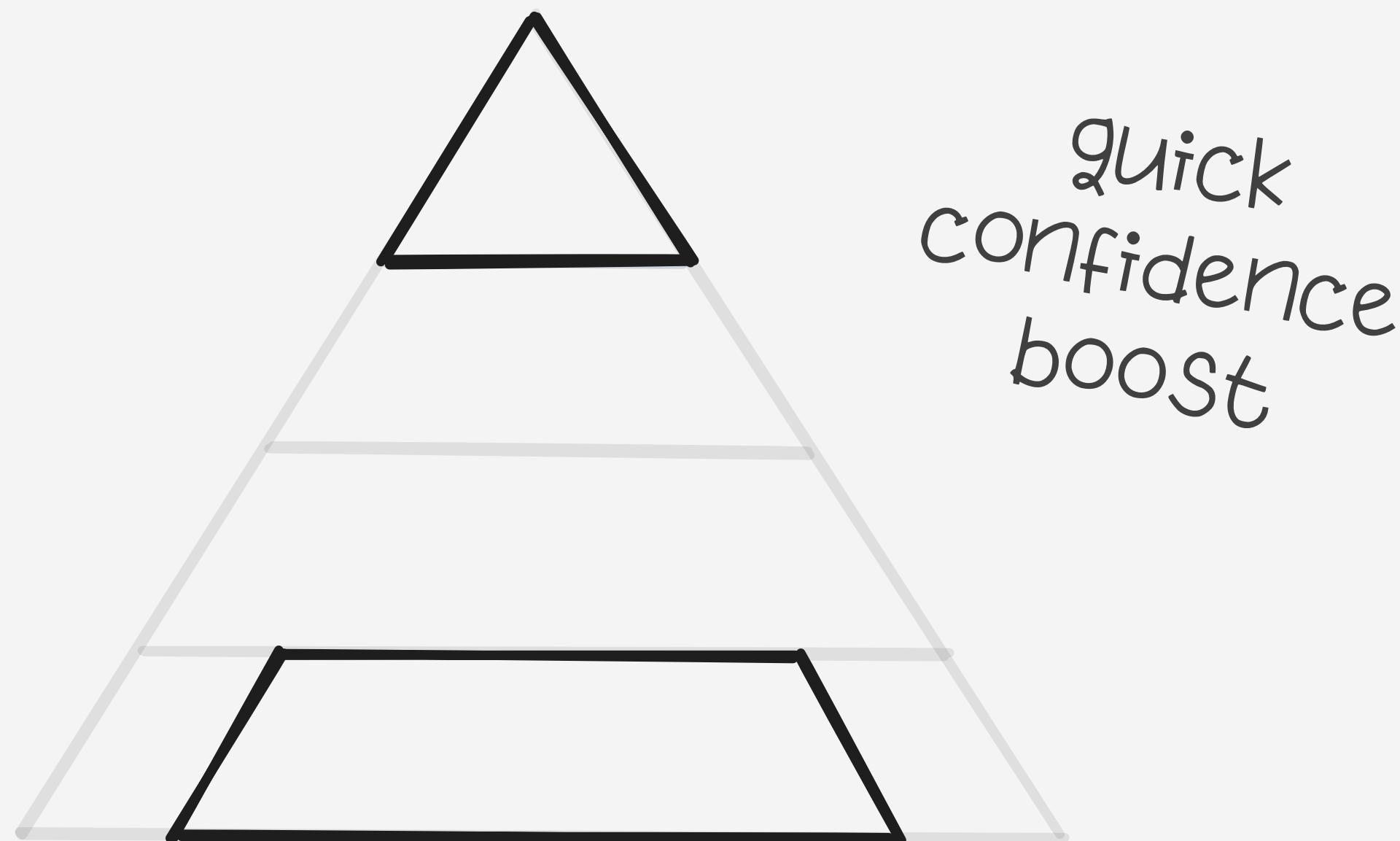
**Have smaller milestones
to work towards the
end goal**



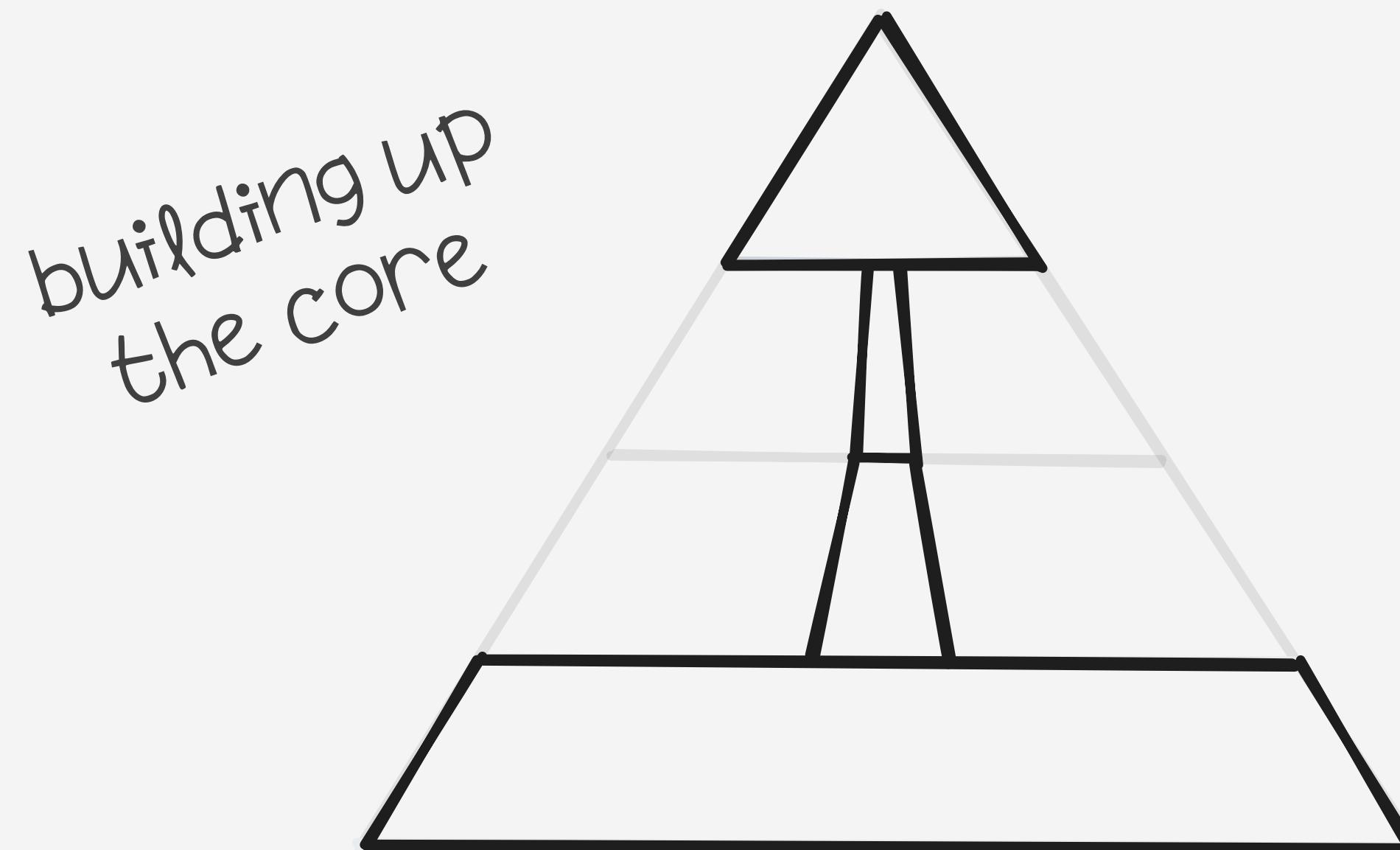
MILESTONES TO YOUR IDEAL SETUP



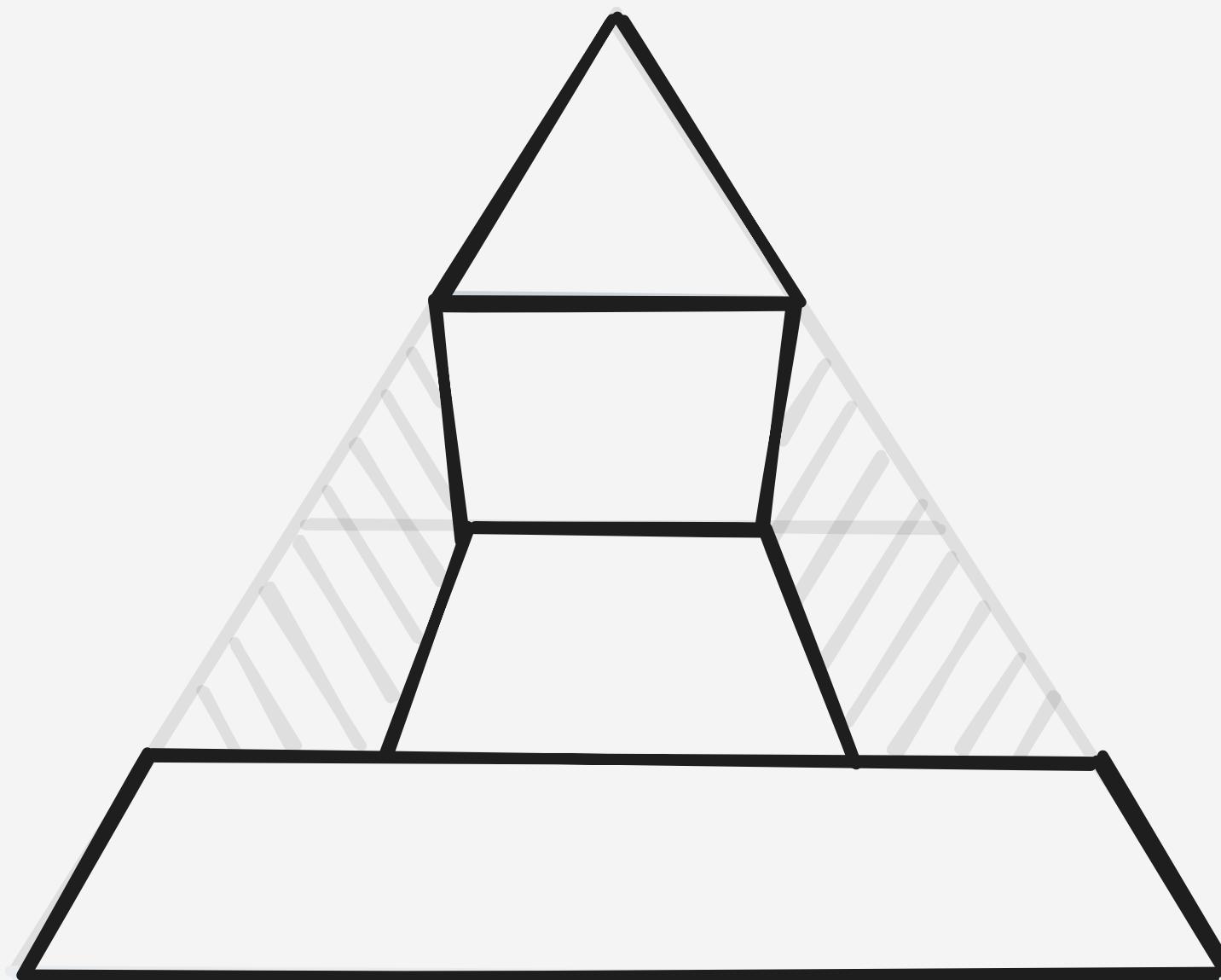
MILESTONES TO YOUR IDEAL SETUP

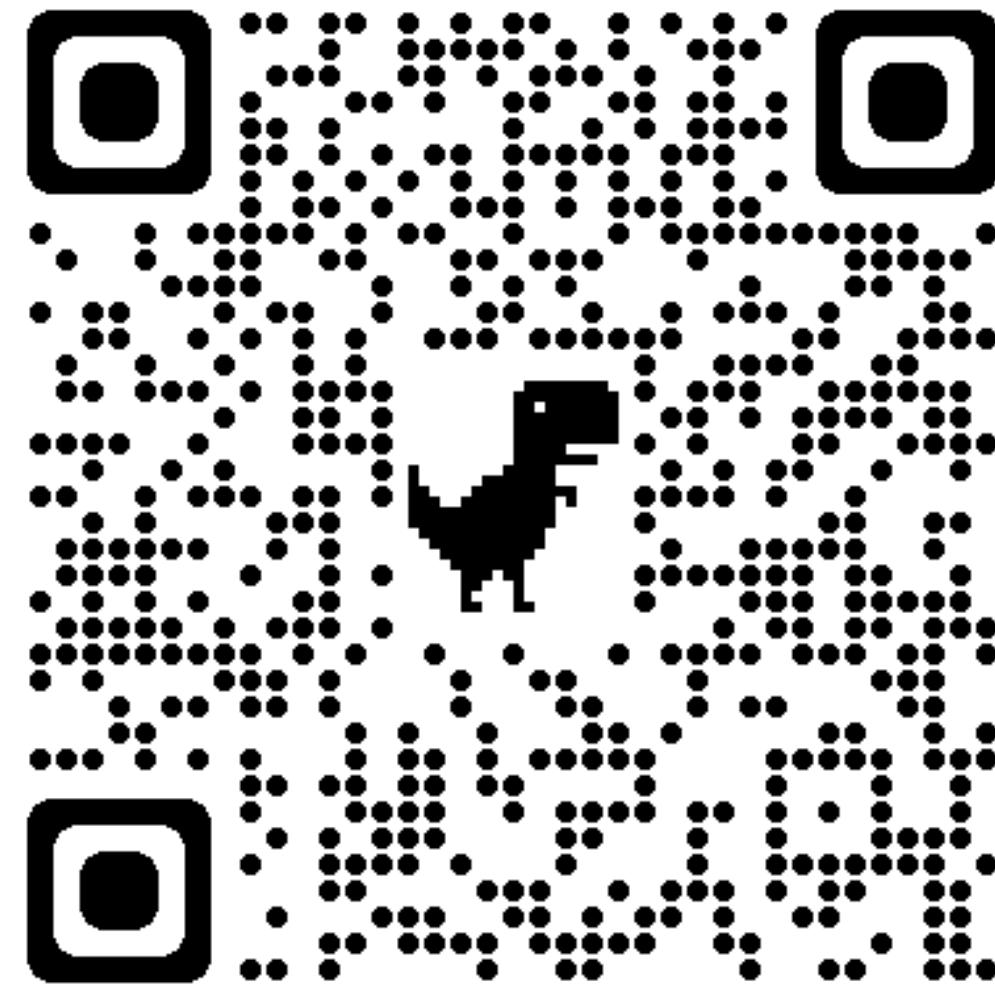


MILESTONES TO YOUR IDEAL SETUP



MILESTONES TO YOUR IDEAL SETUP





THE END.

@jburr90 / Julian Burr

[https://www.julianburr.de/
ddd-melbourne-2024-slides.pdf](https://www.julianburr.de/ddd-melbourne-2024-slides.pdf)