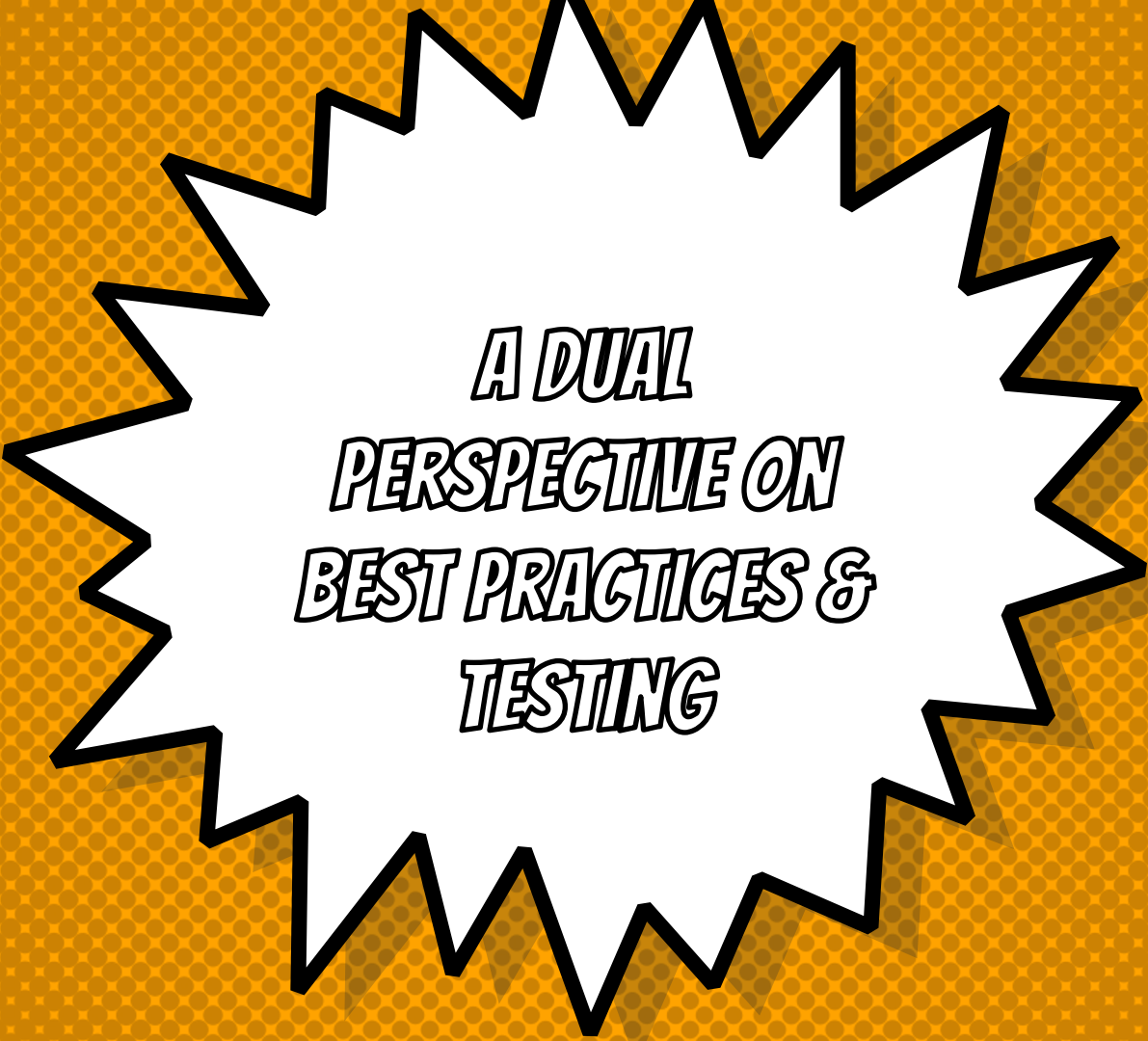




***UNLOCKING  
DEVELOPER  
PRODUCTIVITY***



***A DUAL  
PERSPECTIVE ON  
BEST PRACTICES &  
TESTING***

*DUAL PERSPECTIVE  
ON DEVELOPER  
PRODUCTIVITY?*

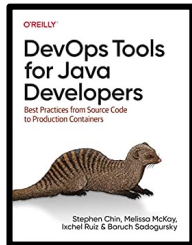


*DUAL PERSPECTIVE  
ON DEVELOPER  
PRODUCTIVITY?*



# ***BARUCH SADOGURSKY - @JBARUCH***

- × Developer Productivity Advocate
- × Gradle Inc
- × Development → DevOps → #DPE



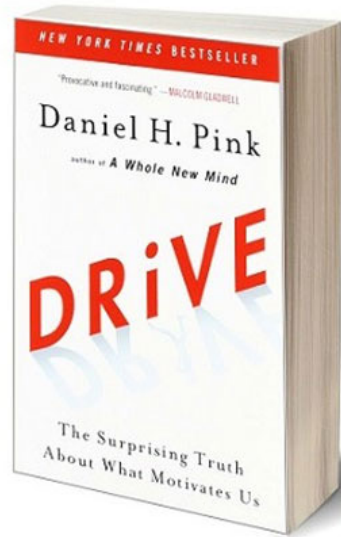
***ELI ALEYNER - @EALEYNER***

- × Co-founder
- × AtomicJar Inc
- × Founding team @AWS
- × Founding team @Bing.com

## ***SHOWNOTES***

- × [speaking.jbaru.ch](https://speaking.jbaru.ch)
- × Slides
- × Video
- × All the links!





@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH





3 FACTORS LEAD TO BETTER  
PERFORMANCE & PERSONAL  
SATISFACTION...

AUTONOMY  
MASTERY  
PURPOSE



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



# ***DON'T RUIN THE FLOW***



***@JBARUCH***

***@EALEYNER***

***#DPE***

***#MONTREALJUG***

***SPEAKING.JBARU.CH***



***"THE BUILD TAKES FOREVER, I  
AM DISTRACTED TO DO OTHER  
THINGS AND THE CONTEXT  
SWITCH IS TERRIBLE"***



***@JBARUCH***

***@EALEYNER***

***#DPE***

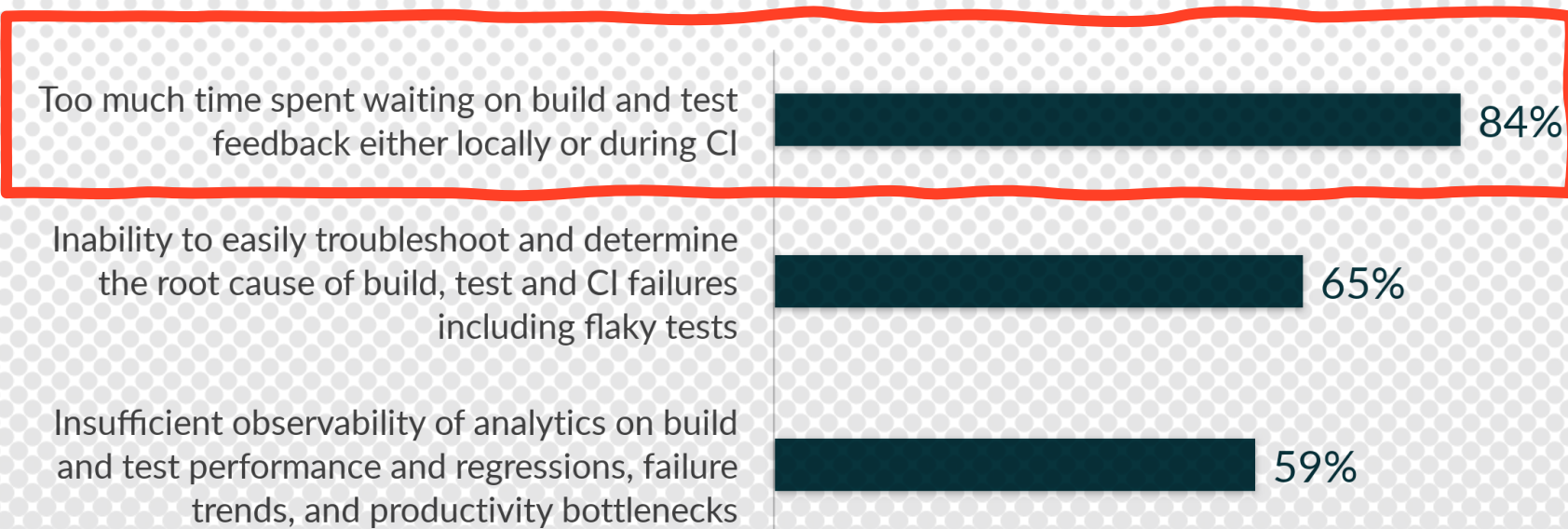
***#MONTREALJUG***

***SPEAKING.JBARU.CH***



# Development Pains are Widespread

Which of the following challenges or pain points did your organization experience prior to implementing Developer Productivity Engineering?



@JBARUGH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUGH



# ***DON'T FRUSTRATE THE DEVELOPERS***



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



***"WE HAVE A FLANKY TEST, IT IS  
IRRELEVANT 99.5% OF THE  
TIME , BUT IT ALWAYS RUNS  
AND IT IS LAST IN THE SUITE"***



[@JBARUCH](#)

[@EALEYNER](#)

[#DPE](#)

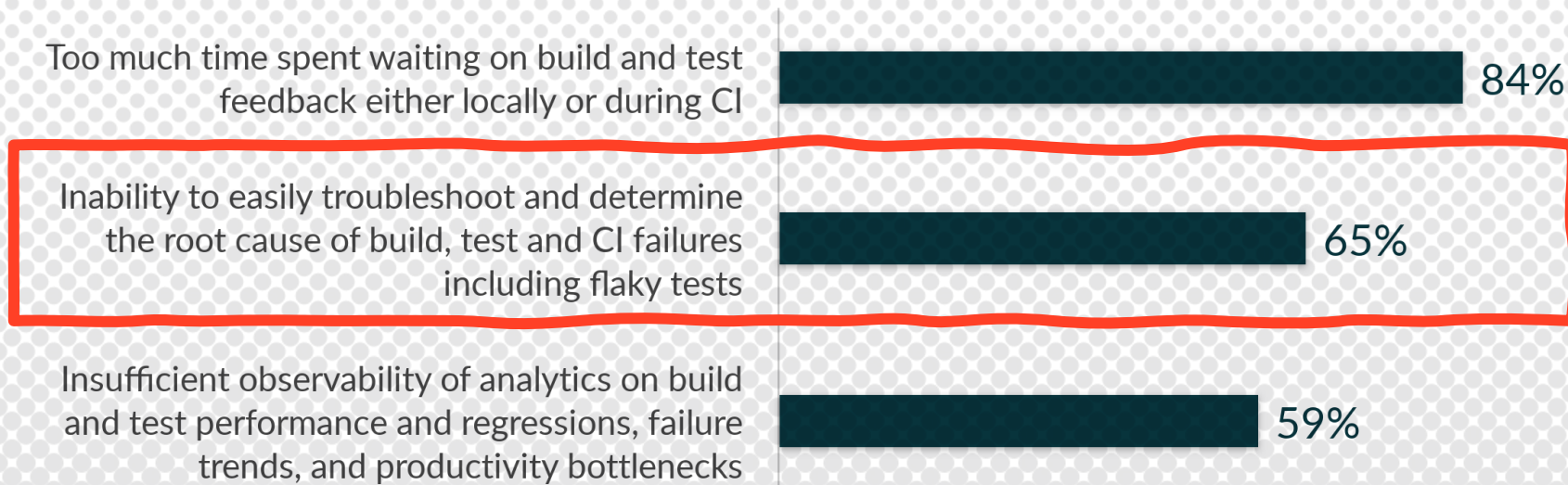
[#MONTREALJUG](#)

[SPEAKING.JBARU.CH](#)



# Development Pains are Widespread

Which of the following challenges or pain points did your organization experience prior to implementing Developer Productivity Engineering?



@JBARUGH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUGH



***DON'T BOIL THE FROG***



***@JBARUCH***

***@EALEYNER***

***#DPE***

***#MONTREALUG***

***SPEAKING.JBARU.CH***





***"I HAVE A FEELING THAT  
EVERYTHING IS SLOWER  
SOMEHOW..."***



***@JBARUGH***

***@EALEYNER***

***#DPE***

***#MONTREALUG***

***SPEAKING.JBARU.CH***



# Development Pains are Widespread

Which of the following challenges or pain points did your organization experience prior to implementing Developer Productivity Engineering?

Too much time spent waiting on build and test feedback either locally or during CI



Inability to easily troubleshoot and determine the root cause of build, test and CI failures including flaky tests



Insufficient observability of analytics on build and test performance and regressions, failure trends, and productivity bottlenecks



@JBARUGH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUGH



# ***DEVELOPER PRODUCTIVITY :: A/M/P :: MOTIVATION***

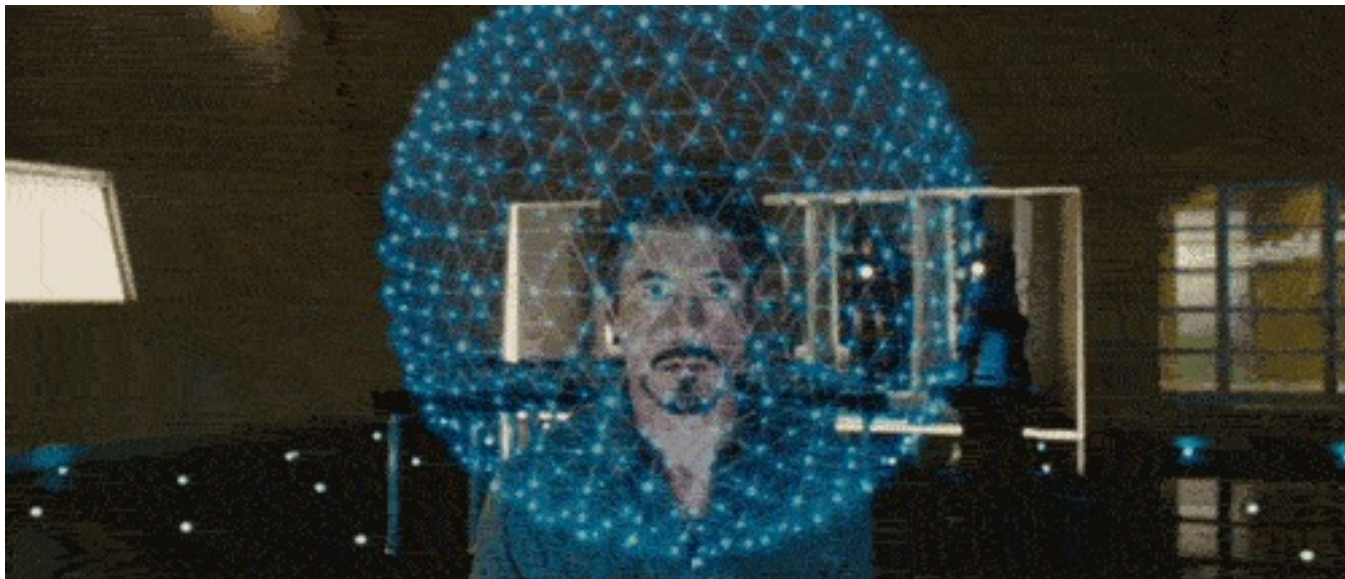
Autonomy → Tools and people aren't in my way

Mastery → Tools and processes help me to excel

Purpose → I want to be *productive*, i.e. create the product



# ***DEVELOPER PRODUCTIVITY ENGINEERING!***



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



# ***DEVELOPER PRODUCTIVITY ENGINEERING***

Foster Faster Feedback

Collaborate through  
Effective Tooling

Embrace Rigorous  
Observability for  
Proactive Improvement

Eliminate Toil for  
Developers

Prioritize Automation  
and Eliminate  
Bottlenecks

Dedicated  
Organizational Mindset

Outcomes Over Output



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



***TALK IS CHEAP,  
SHOW ME THE  
GOODS!***



[@JBARUCH](#)

[@EALEYNER](#)

[#DPE](#)

[#MONTREALJUG](#)

[SPEAKING.JBARU.CH](#)



## ***SMALL DPE IMPROVEMENTS MAKE A HUGE DIFFERENCE***

- × Generate code faster: Better IDE
- × Test better: Testcontainers
- × Enforce better code: Sonar
- × Test more reliably: Flaky test detection
- × Foster Faster Feedback:



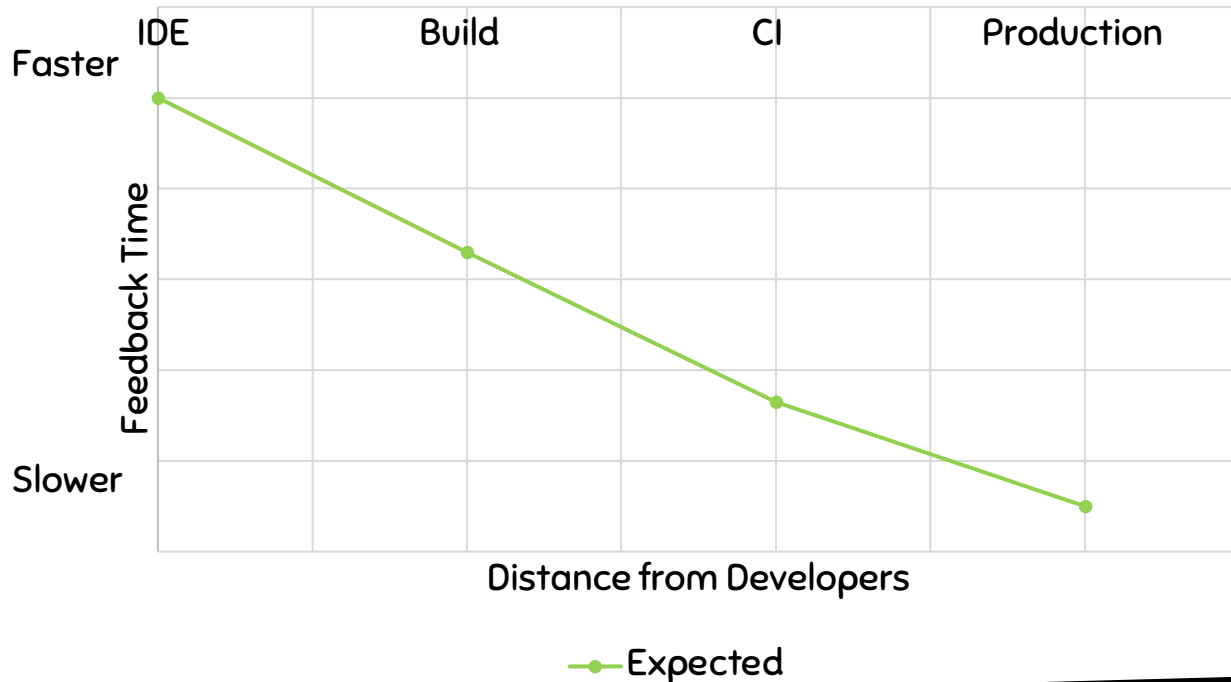
## ***FEEDBACK EFFICIENCY***

- × IDE: Sub-seconds (I type, it marks it red)
- × Build: Seconds
- × CI: Minutes
- × Production: Hours/Days





# REVERSE DEPENDENCY ON DISTANCE FROM DEVELOPERS



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

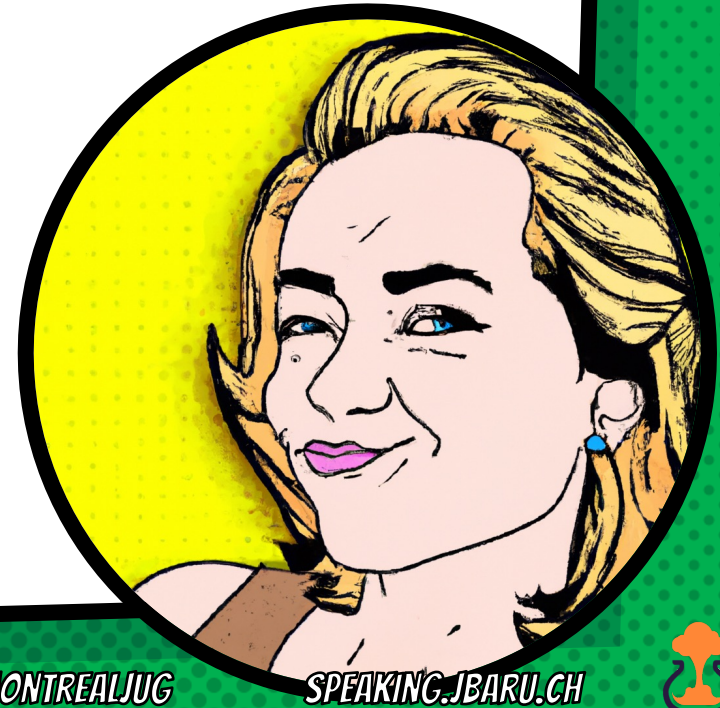
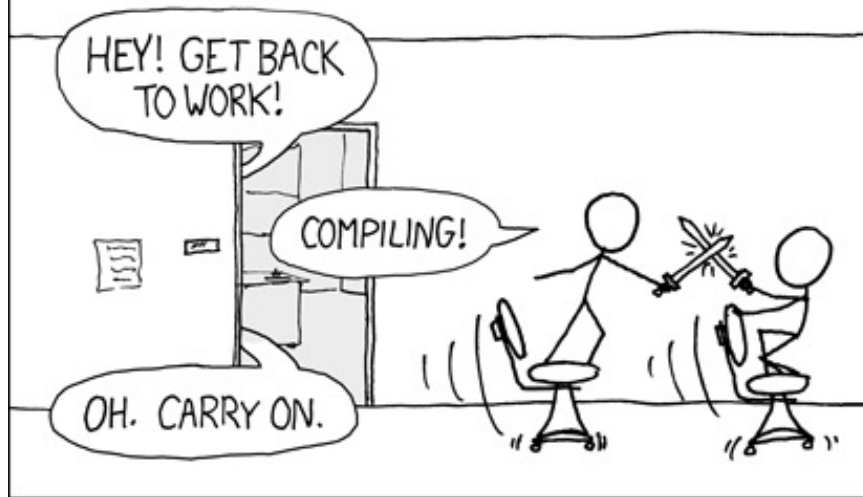
SPEAKING.JBARU.CH



# REVERSE DEPENDENCY ON DISTANCE FROM DEVELOPERS



THE #1 PROGRAMMER EXCUSE  
FOR LEGITIMATELY SLACKING OFF:  
"MY CODE'S COMPILING."



@JBARUCH

@EALYNER

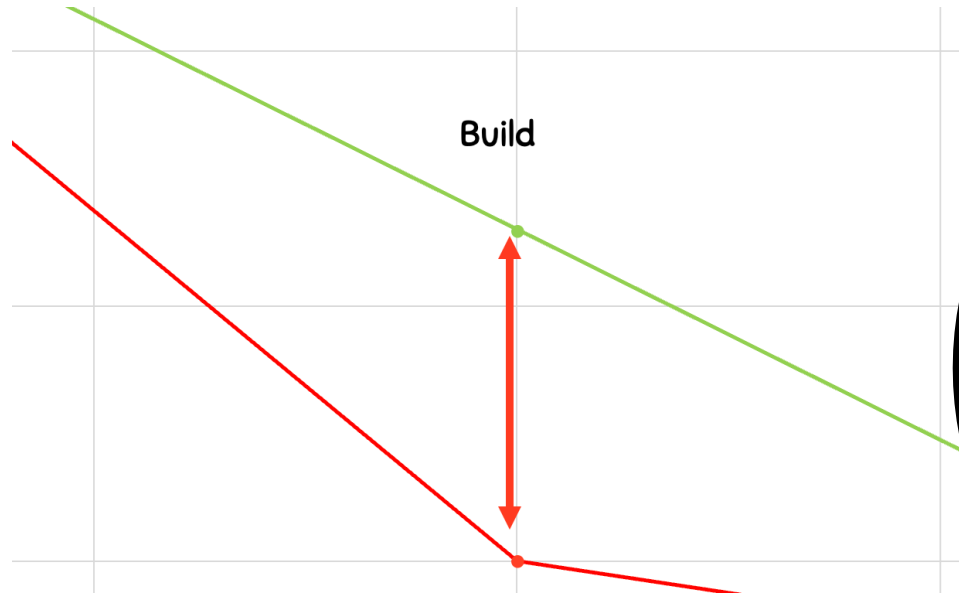
#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



**IT IS SLOW!**



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



← → ↻ 🛡️ 🔒 <https://www.bruceeckel.com/2021/01/02/the-problem-with-gradle/>

stymied me. This is the problem I had with Gradle:

### ***To do anything you have to know everything***

Yes, it's hypothetically possible to create a simple `build.gradle` file for a basic build. But usually by the time you get to the point of *needing* a Gradle build, your problem is complicated enough that you must do more. And it turns out that “doing more” translates to “knowing everything.” Once you get past the simple things you fall off a cliff.

Think of the grappling shoes in the very first episode of *Rick and Morty*. Rick explains that the shoes allow you to walk on vertical surfaces, so Morty puts them on and promptly falls down a cliff, after which Rick explains that “you have to turn them on.” Gradle is my grappling shoes.

My goal here is to give you perspective, so as you fall down the cliff face you will understand what is happening, and what is necessary to climb back up.



@JBARUGH

@EALYNER

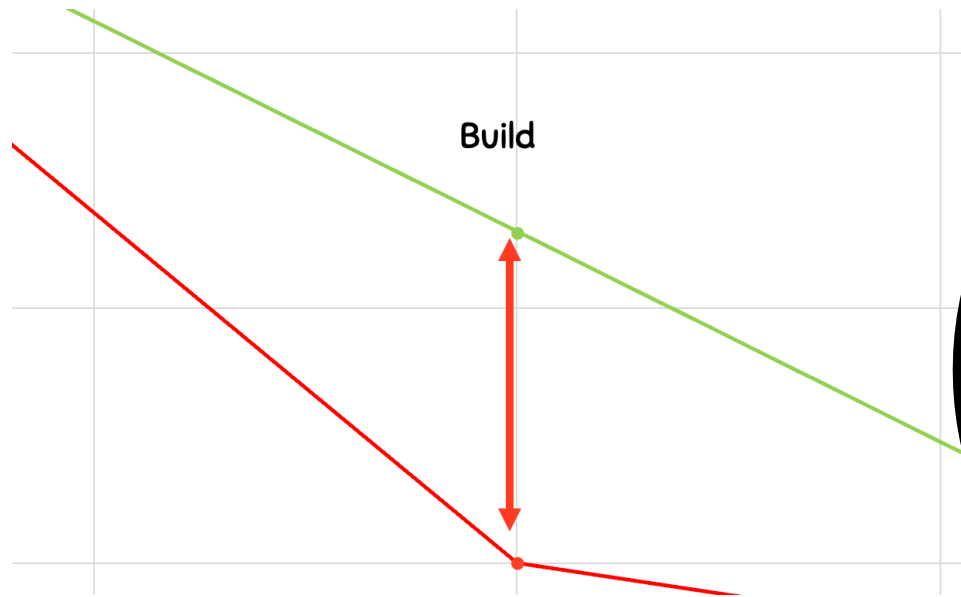
#DPE

#MONTREALUG

SPEAKING.JBARU.CH



**IT IS SLOW AND THE DEVELOPERS HAVE NO IDEA WHY!**



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



## ***WHAT IS BUILD?***

- × Project setup
- × Downloading the Internet
- × Artifact generation: Compilation, packaging, etc
- × Tests
- × Artifact deployment



## ***WHAT CAN GO WRONG?***

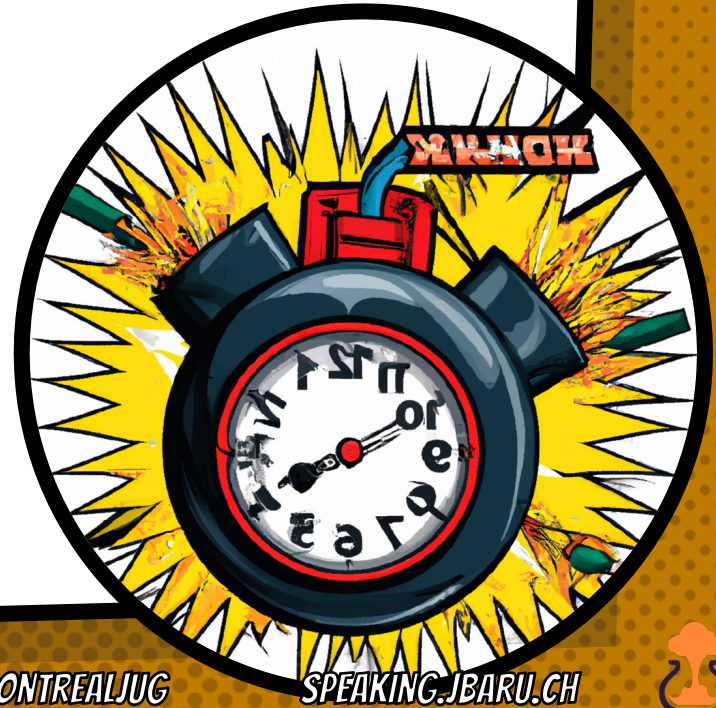
- × Project setup
- × Downloading the Internet
- × Artifact generation: Compilation, packaging, etc
- × Tests
- × Artifact deployment





***WHEN CAN IT GO WRONG?***

***AT ANY TIME.***



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



# ***THE BUILD FRUSTRATES THE DEVELOPERS***



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



# LET'S ASK CHATGPT



What techniques can I use to speed up a Maven build?



Sure, here are some quick tips to speed up a Maven build:

1. **Skip Tests:** Use `-DskipTests` to skip running tests.
2. **Parallel Execution:** Use `-T` flag to run in parallel. E.g., `-T 4` for 4 threads.
3. **Offline Mode:** Use `--offline` to avoid checking for updates.
4. **Incremental Build:** Use `mvn -am -pl` to build only changed modules and their dependencies.
5. **Profile:** Use Maven profiles to conditionally enable plugins.
6. **Resource Filtering:** Limit resource filtering to only necessary files.
7. **Use a Mirror:** Set up a local Nexus or Artifactory mirror.

Hope that helps! Got a preference for Gradle, but these should give your Maven build a boost.



## ***WHAT THE ACTUAL F\*CK?!***

- × Skipping tests defeats the purpose of the build!
- × How about skipping compilation?
- × We want faster feedback, not less feedback



***WHAT FEEDBACK DO WE WANT?***



***@JBARUCH***

***@EALEYNER***

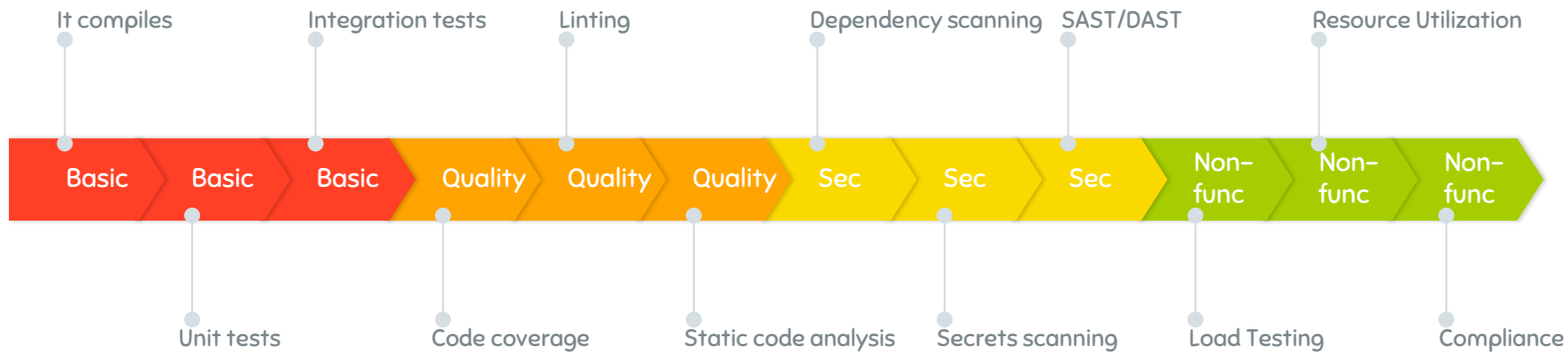
***#DPE***

***#MONTREALJUG***

***SPEAKING.JBARU.CH***



# CI/CD PIPELINE QUALITY GATES





**CI/CD? NOT GREAT, NOT TERRIBLE.**



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



# TWO TYPES OF FEEDBACK

## ASYNCHRONOUS

- x e.g., CI/CD
- x we never wait for it
- x results are distracting

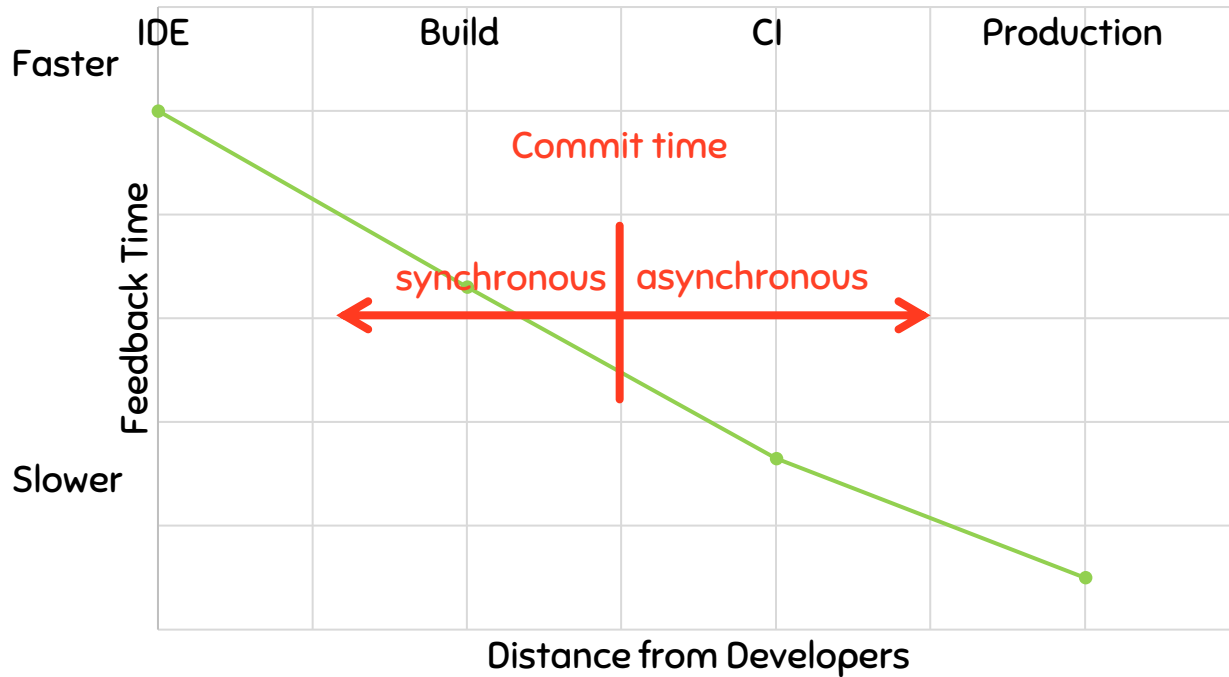
## SYNCHRONOUS

- x e.g., build
- x we'll wait for it in the flow
- x we'll be pissed off when it's slow

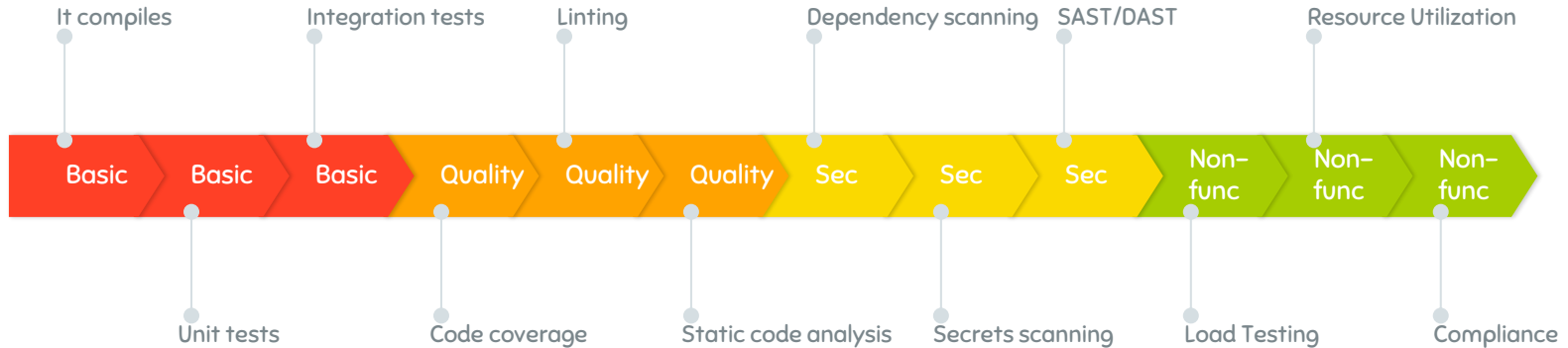




# REVERSE DEPENDENCY ON DISTANCE FROM DEVELOPERS



# IDEAL BUILD TIME FEEDBACK



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



***BUT WON'T IT SLOW  
DOWN THE BUILD?!***



***@JBARUCH***

***@EALEYNER***

***#DPE***

***#MONTREALJUG***

***SPEAKING.JBARU.CH***



## ***DELIGHTFUL BUILD (PICK TWO):***



***PROVIDES MAX FEEDBACK***



***FAST***



[@JBARUCH](https://twitter.com/JBARUCH)

[@EALEYNER](https://twitter.com/EALEYNER)

[#DPE](https://twitter.com/DPE)

[#MONTREALJUG](https://twitter.com/MONTREALJUG)

[SPEAKING.JBARU.CH](https://speaking.jbaru.ch)



***SKIP WHAT CAN BE SKIPPED  
(BUT NO MORE!)***



***@JBARUCH***

***@EALEYNER***

***#DPE***

***#MONTREALJUG***

***SPEAKING.JBARU.CH***



## ***AVOIDANCE: INCREMENTAL BUILD***

- × Don't build what didn't changed
- × Don't build what isn't affected



## ***AVOIDANCE: INCREMENTAL BUILD SHORTCOMINGS***

- × Relies on produced artifacts
- × Relies on architectural decisions



## ***AVOIDANCE: CACHING***

- × Makes the build faster
- × Makes the build faster for everybody
- × Makes the build faster always
- × Makes all parts of the build faster





Build time ⓘ  
12 min 28 sec

Serial execution ⓘ  
15 min 13 sec (2.46x)

Avoidance savings ⓘ  
39 min 10 sec (73.62%)

Build cache overhead ⓘ  
18 sec

Dependency downloading ⓘ  
2.2 sec

Up to date ⓘ  
19 min 33 sec

Local build cache ⓘ  
4.3 sec

Remote build cache ⓘ  
19 min 33 sec



1h 40m

1h 15m

50m

25m



@JBARUCH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUCH



## ***AVOIDANCE: PREDICTIVE TEST SELECTION***

- × Learns code changes effects de-facto
- × Skips tests with high degree of confidence



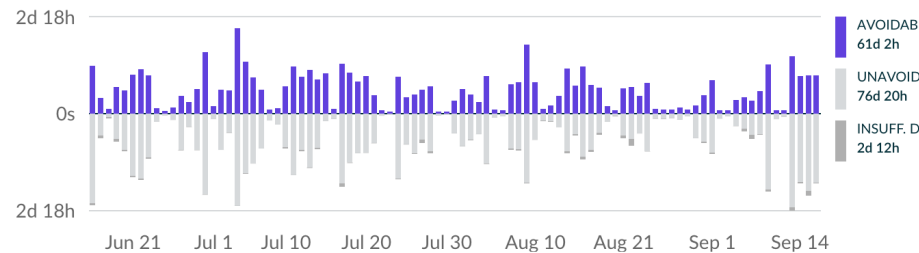
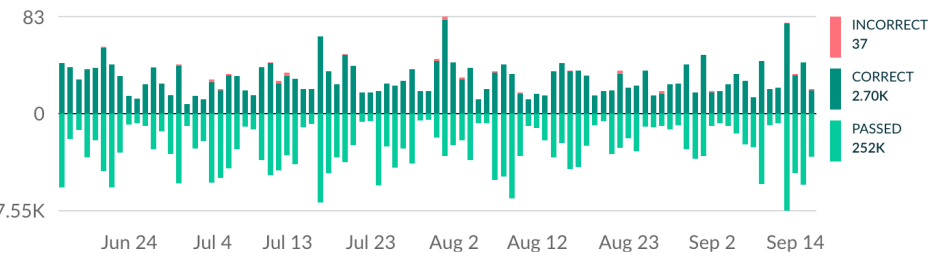
Predictive Test Selection > [Find test task/goal](#)

age (78 builds) Simulations (51.3K builds)  
 in build time: 11 min 18 sec **Mean build time: 5 min 27 sec**

Selection profile ?  
Conservative Standard F

**Task/Goal failures predicted ?** 98.7% (2.70K of 2.74K total)  
**Test failures predicted ?** 96.4% (3.55K of 3.68K total)

**Savings potential ?** 61 d 2 hr (44%)  
**Avoidable tests ?** 1.84M (30%)  
**Unavoidable tests ?** 4.13M (68%)



Tasks/Goals by mean duration (top 50) ?

All predictions Only incorr

	Task/Goal failures predicted ?	Test failures predicted ?	Simulations ?	Mean duration ?	Total test time ?	Savings potential ?
ng-boot-build > :spring-boot-project:spring-boot-tools:spring-b...	15 / 16 (93.8%)	15 / 17 (88.2%)	637 / 677	31 min 12 sec	13 d 14 hr	7 d 9 hr (54%)
ng-boot-build > :spring-boot-project:spring-boot-tools:spring-b...	84 / 87 (96.6%)	86 / 89 (96.6%)	612 / 661	18 min 18 sec	7 d 10 hr	5 d 2 hr (69%)

***SPEED UP WHAT  
CAN'T BE SKIPPED***



***@JBARUCH***

***@EALEYNER***

***#DPE***

***#MONTREALJUG***

***SPEAKING.JBARU.CH***



## ***TEST PARALLELIZATION***

- × Use max power of local machine
- × (Yes, your boss should buy you the bleeding edge)



Task path	Started after ?	Duration ?
:clean	0.499s	0.053s
:compileJava	0.553s	0.146s
:processResources NO-SOURCE	0.699s	0.001s
:classes	0.700s	0.000s
:jar	0.701s	0.040s
:assemble	0.741s	0.000s
:compileTestJava	0.741s	0.242s
:processTestResources NO-SOURCE	0.984s	0.000s
:testClasses	0.984s	0.001s
<b>:test</b>	<b>0.985s</b>	<b>1m 59.135s</b>
:check	2m 0.120s	0.001s
:build	2m 0.121s	0.001s

Task path	Started after ?	Duration ?
:clean	0.416s	0.048s
:compileJava	0.465s	0.085s
:processResources NO-SOURCE	0.550s	0.000s
:classes	0.550s	0.000s
:jar	0.551s	0.040s
:assemble	0.591s	0.000s
:compileTestJava	0.592s	0.212s
:processTestResources NO-SOURCE	0.804s	0.001s
:testClasses	0.805s	0.000s
<b>:test</b>	<b>0.805s</b>	<b>10.553s</b>
:check	11.359s	0.000s
:build	11.359s	0.000s

```

tasks.test { this: Test!
    onlyIf { true }
    useJUnitPlatform()
    maxParallelForks = Runtime.getRuntime().availableProcessors()
    testLogging { this: TestLoggingContainer

```



## ***TEST DISTRIBUTION***

- × CI uses fan-out to speed-up tests
- × Shouldn't you enjoy it for local tests?
- × Use the cloud to distribute test load
- × RUN ALL THE TESTS!



## ***WHY NOT JUST USING CI FAN-OUT?***

- × Relying on shared CI infrastructure
- × CI infrastructure is not optimized for real-time feedback!
- × Are the agents as fast as they can be?





***DON'T LET IT SLIDE***



**@JBARUCH**

**@EALEYNER**

**#DPE**

**#MONTREALJUG**

**SPEAKING.JBARU.CH**



## ***OBSERVE AND IMPROVE***

- × Measure local build times across time and environments
- × Detect downfacing trends
- × Find root causes and improve



Basic search Advanced search

User Hostname Project Requested tasks/goals/targets Build tool Build tool version Start time Custom values Tags Outcome Refresh

Overview Build time Serial execution Avoidance savings Build cache overhead Dependency downloading Day Week Month

Build time

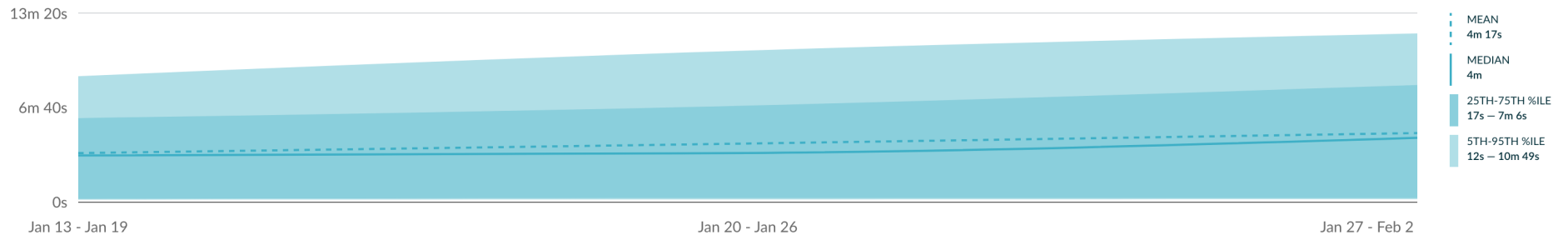
4 min 17 sec

Non execution

28 sec

Execution

3 min 49 sec



Serial execution

35 min 41 sec (7.7x)



Non-cacheable

44 sec



Build cache miss

34 min 40 sec



Build cache hit

4.2 sec



Up to date & non-actionable

13 sec



***SAME TRADEOFF FOR TESTS***

***EXPENSIVE, SLOW, BUT PROD-LIKE  
OR  
NAÏVE, USELESS, BUT FAST***



***@JBARUCH***

***@EALEYNER***

***#DPE***

***#MONTREALJUG***

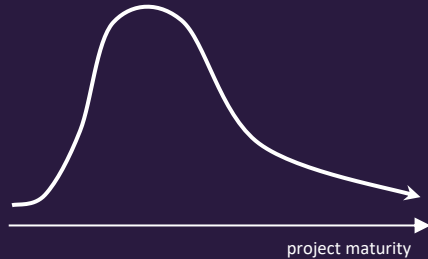
***SPEAKING.JBARU.CH***



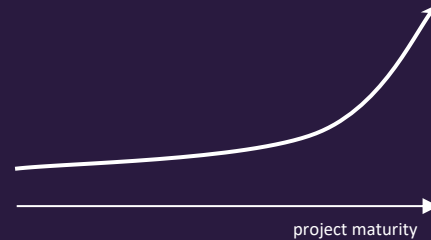
# CI often depends on Staging environments

*But fixing staging environments is just “faster horses”*

Value of Staging Environment tests degrades over time as they become longer to run, provide flaky outcomes, while still sitting too far from Software Development Lifecycle



Cost of maintaining Staging Environment based tests increases as projects mature



# Modern Infrastructure Has Made Testing Harder For Developers

## Mocks

+ FAST

- FALSE

- HARD TO MAINTAIN

## Staging / QA

+ REALISTIC

- FRAGILE

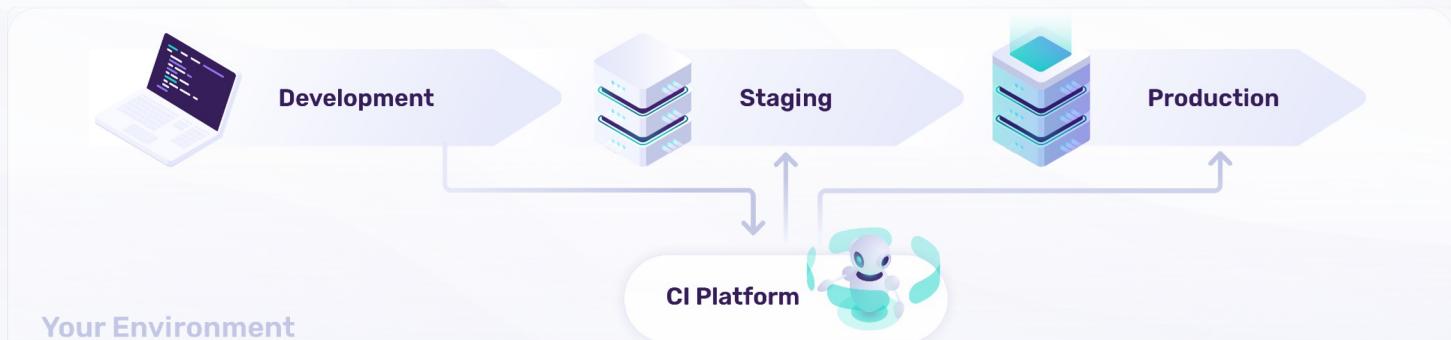
- SLOW

## Test in Production

+ REAL

- EXPENSIVE

- RISKY



@JBARUCH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARU.GH



# The Testcontainers Experience



## Testcontainers

Unit tests with real dependencies

+ FAST

+ REALISTIC

+ COST-EFFECTIVE



@JBARUGH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUGH





**Testcontainers**  
BY ATOMICJAR

# Unit Tests with Real Dependencies

Testcontainers is an open source framework for providing throwaway, lightweight instances of databases, message brokers, web browsers, or just about anything that can run in a Docker container.



*@JBARUCH*

*@EALEYNER*

*#DPE*

*#MONTREALJUG*

*SPEAKING.JBARU.CH*







# TEST DEPENDENCIES AS CODE



Testcontainers

WE ♥ OPEN SOURCE  
& EMPLOY MAINTAINERS

Testcontainers is an open source framework for providing throwaway, lightweight instances of test dependencies.

```
GenericContainer redis = new GenericContainer("redis:5.0.3-alpine")  
    .withExposedPorts(6379)
```



@JBARUGH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARU.GH





# Testcontainers is used by



Uber

NETFLIX



7Wise



INSTANA  
an IBM Company



@JBARUGH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUGH





Testcontainers

WE ♥ OPEN SOURCE  
& EMPLOY MAINTAINERS



## FOR YOUR ENTIRE STACK

Test against any database, message broker, browser...  
or just about anything that runs in a Docker container!



@JBARUCH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUCH



# Modules

Testcontainers modules are preconfigured implementations of various dependencies that make writing your tests even easier!














[Clear Filter](#)  Official Modules

## Languages

[All](#)[.NET](#)[Go](#)[Java](#)[Node.js](#)

## Categories

[All](#)

 <b>ArangoDB</b> NoSQL Database Java Node.js	 <b>Azure Cosmos DB</b> Cloud Java	 <b>Azure SQL Edge</b> Cloud .NET
 <b>Cassandra</b> NoSQL Database Java	 <b>ClickHouse</b> Relational Database Java	 <b>CockroachDB</b> Relational Database Java 
 <b>Consul</b> Other Java	 <b>Couchbase</b> NoSQL Database Java Go .NET	 <b>CouchDB</b> NoSQL Database .NET
 <b>DB2</b> Relational Database	 <b>Dynamalite</b> NoSQL Database	 <b>DynamoDB</b> NoSQL Database

[@JBARUCH](#)[@EALEYNER](#)[testcontainers.com/modules](https://testcontainers.com/modules)[#EDPE](#)[#MONTREALUG](#)[SPEAKING.JBARUCH](#)



Testcontainers

WE ♥ OPEN SOURCE  
& EMPLOY MAINTAINERS

## </> FOR ANY LANGUAGE

Testcontainers libraries exist for all popular languages including Java, Go, .NET, NodeJS, and more.



Java



Go



.NET



Node.js



Python



Rust



Haskell



Ruby



@JBARUCH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUCH





**Testcontainers**  
**Desktop** BY ATOMICJAR

# Simple Local Development with Real Dependencies

Testcontainers Desktop is the free companion app  
to open source Testcontainers libraries.



*@JBARUCH*

*@EALEYNER*

*#DPE*

*#MONTREALJUG*

*SPEAKING.JBARU.CH*



# Testcontainers Desktop

## The free companion app.

Debug your Testcontainers-powered dependencies

The app lets you proxy any service to a fixed port to easily inspect it with your favorite debugging tools.

Track and analyze your test sessions

Dashboards that provide you and your team with insights into your development and testing patterns.

Switch container runtimes and burst to the cloud

With Testcontainers Cloud, you can even run them in the cloud on demand, while saving your local resources.



**@JBARUGH**

**@EALEYNER**

**#DPE**

**#MONTREALUG**

**SPEAKING.JBARUGH**





# Testcontainers Cloud:

## Test without limits. Ship with confidence.



### Developer-first Testing

Test everything on your laptop  
without worrying about  
resources; no local docker  
daemon needed



### Effortlessly Fast CI

Run your ever-growing test suite  
without scaling your CI, and  
speed it up by running tests in  
parallel



### Reliable Test Suites

Enhance team efficiency by  
getting rid of flaky tests and  
ensuring consistency from dev to  
CI



**@JBARUGH**

**@EALEYNER**

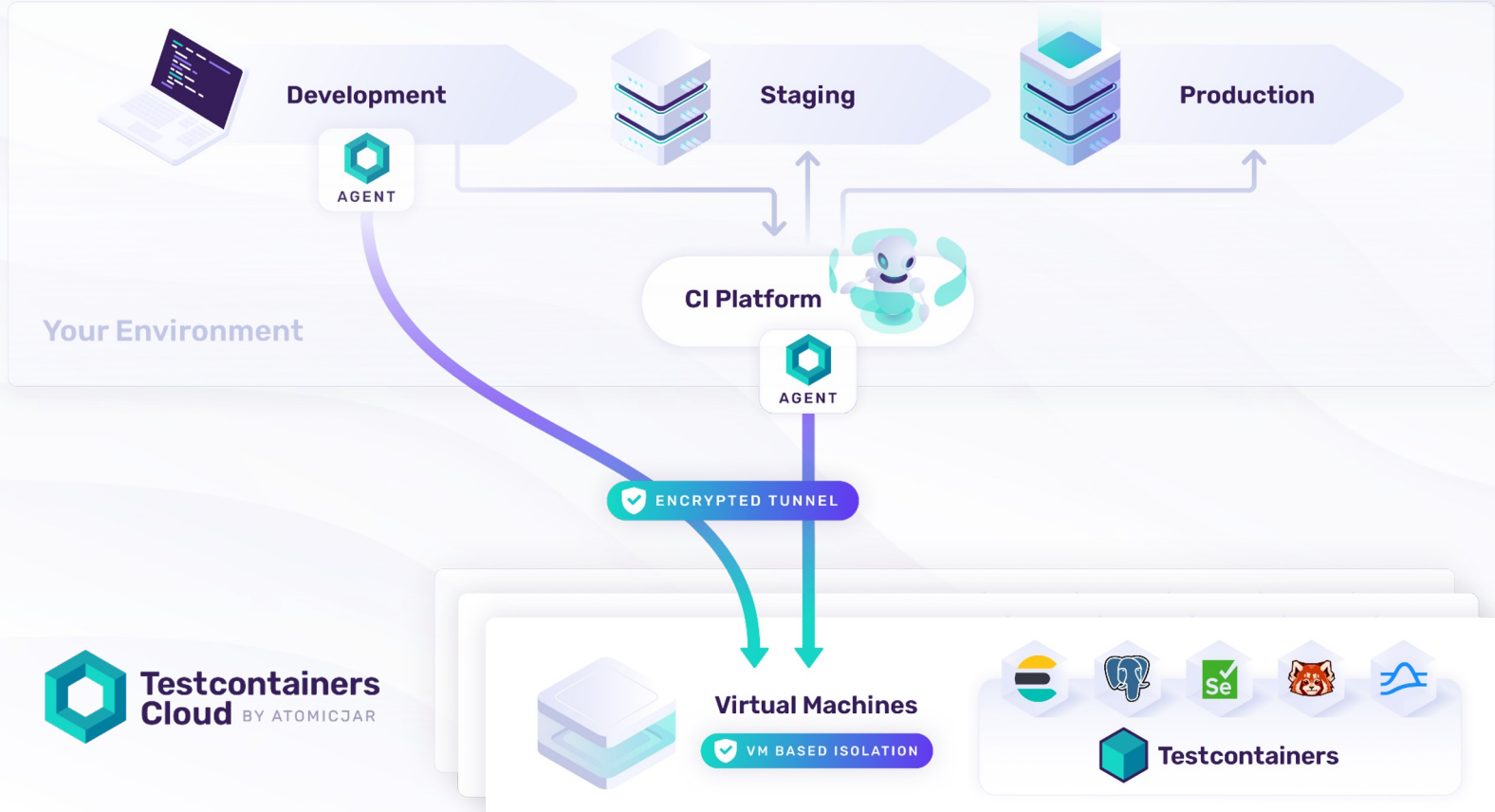
**#DPE**

**#MONTREALUG**

**SPEAKING.JBARU.GH**



# How it works



@JBARUCH

@EALEYNER

#DPE

#MONTREALUG

SPEAKING.JBARUCH



***THE GAINS ARE REAL!***



***@JBARUCH***

***@EALEYNER***

***#DPE***

***#MONTREALJUG***

***SPEAKING.JBARU.CH***



# DPE Dramatically Improves Productivity

Almost every surveyed IT organization agreed that “Since integrating Developer Productivity Engineering into our development process, the time savings we experienced on build and test cycle times have dramatically improved developer productivity.”



Source: TechValidate survey of 63 users of Gradle Enterprise



@JBARUGH

@EALEYNER

#DPE

#MONTREALUG

Gradle Enterprise

SPEAKING.JBARUGH



## DPE Fosters Developer Joy

84% of surveyed users agree that DPE's impact on their toolchain makes their job more enjoyable.



Source: TechValidate survey of 51 users of Gradle Enterprise



✓ Validated

Published: Jul. 2, 2023 TVID: 930-05A-A5F



[@JBARUGH](#)

[@EALEYNER](#)

[#DPE](#)

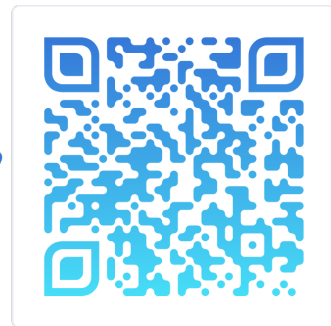
[#MONTREALUG](#)

[SPEAKING.JBARUGH](#)



## ***LEARN MORE AND TRY IT TODAY!***

- × Take the Gradle/Maven Speed Challenge!
- × Be DPE Agent of Change!
- × Read the DPE Handbook!
- × Watch the DPE Summit videos!



*Scan me*

***SPEAKING.JBARU.CH***



@JBARUCH

@EALEYNER

#DPE

#MONTREALJUG

SPEAKING.JBARU.CH



# Q&A AND

# SOCIAL ADS



- × @JBARUCH
- × @EALYENER
- × #MONTREALJUG
- × #DPE
- × SPEAKING.JBARU.CH