

REPLs All The Way Up

Avdi Grimm

<https://avdi.codes>

Jessica Kerr

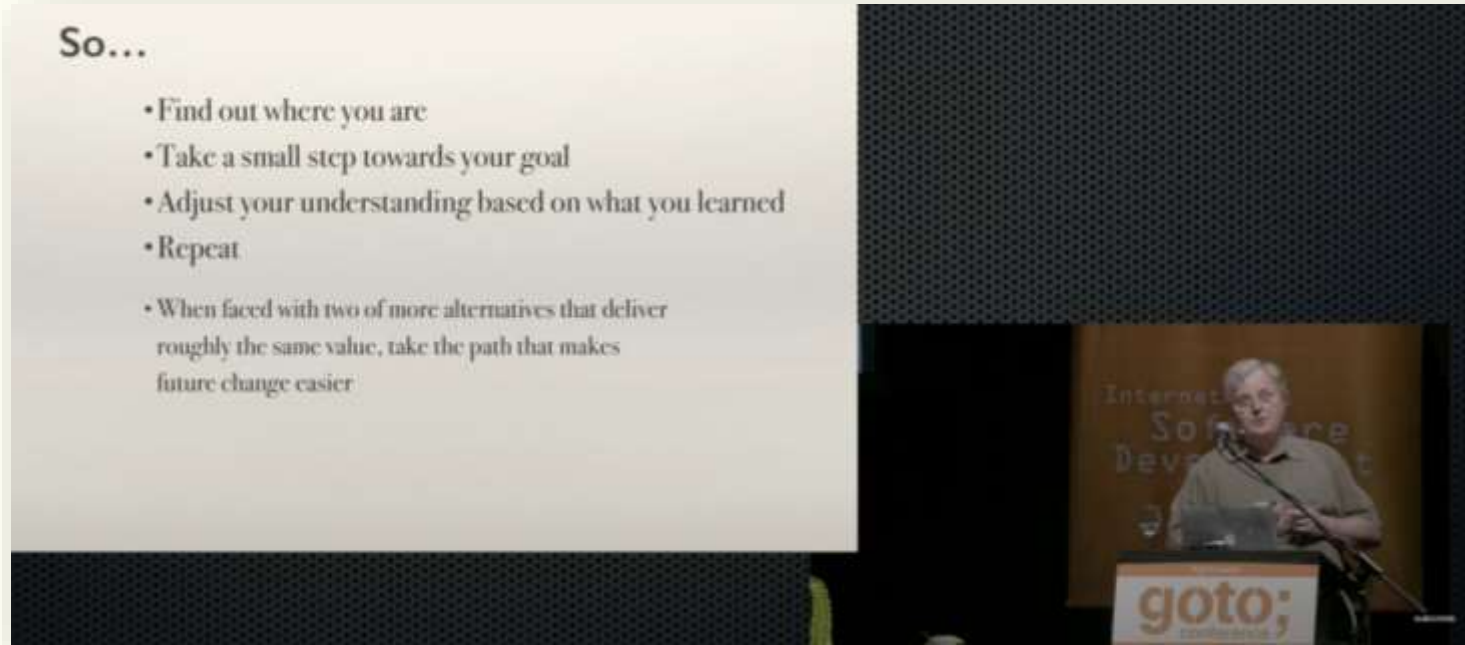
<https://jessitron.com>



Feedback Loops

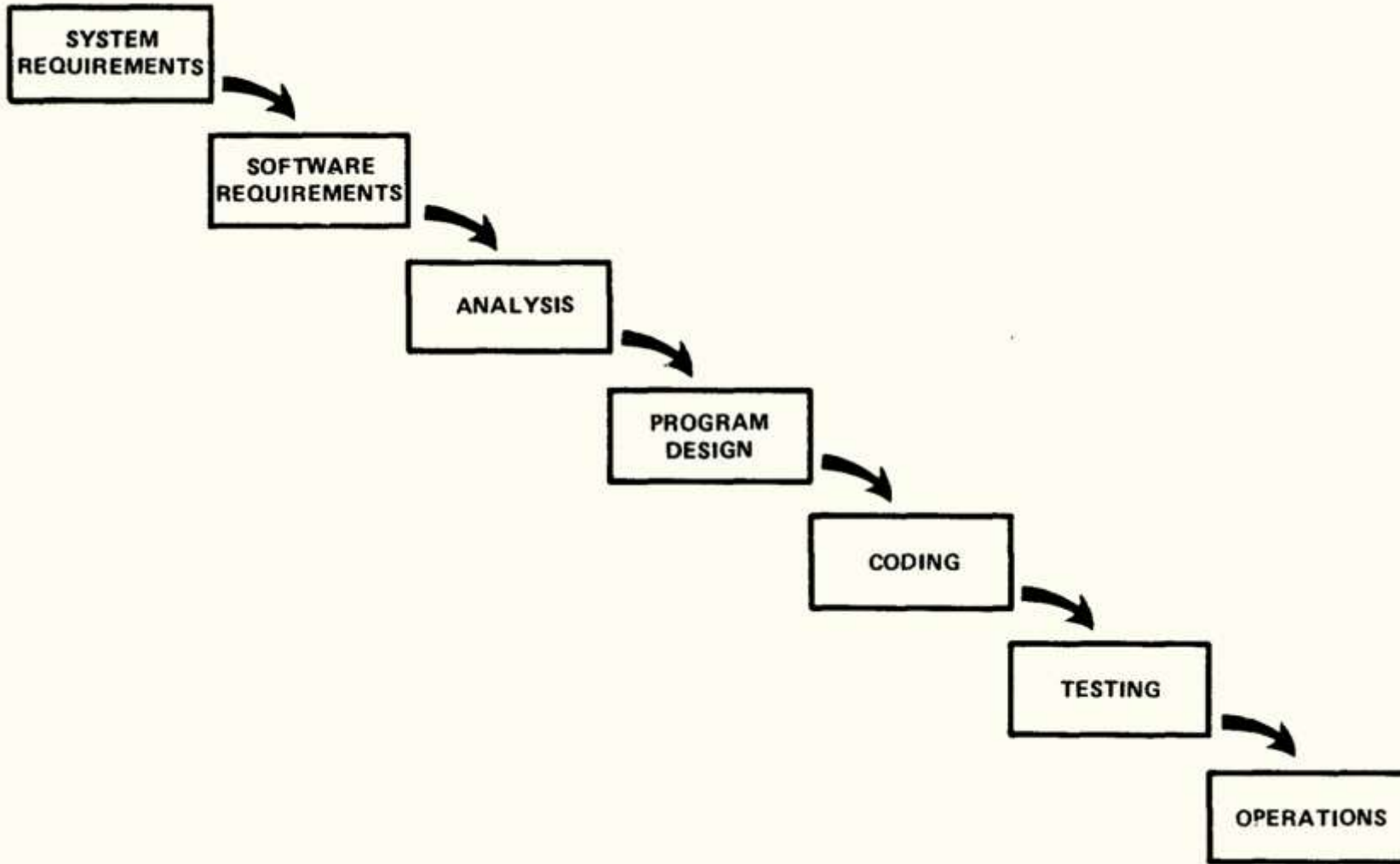
So...

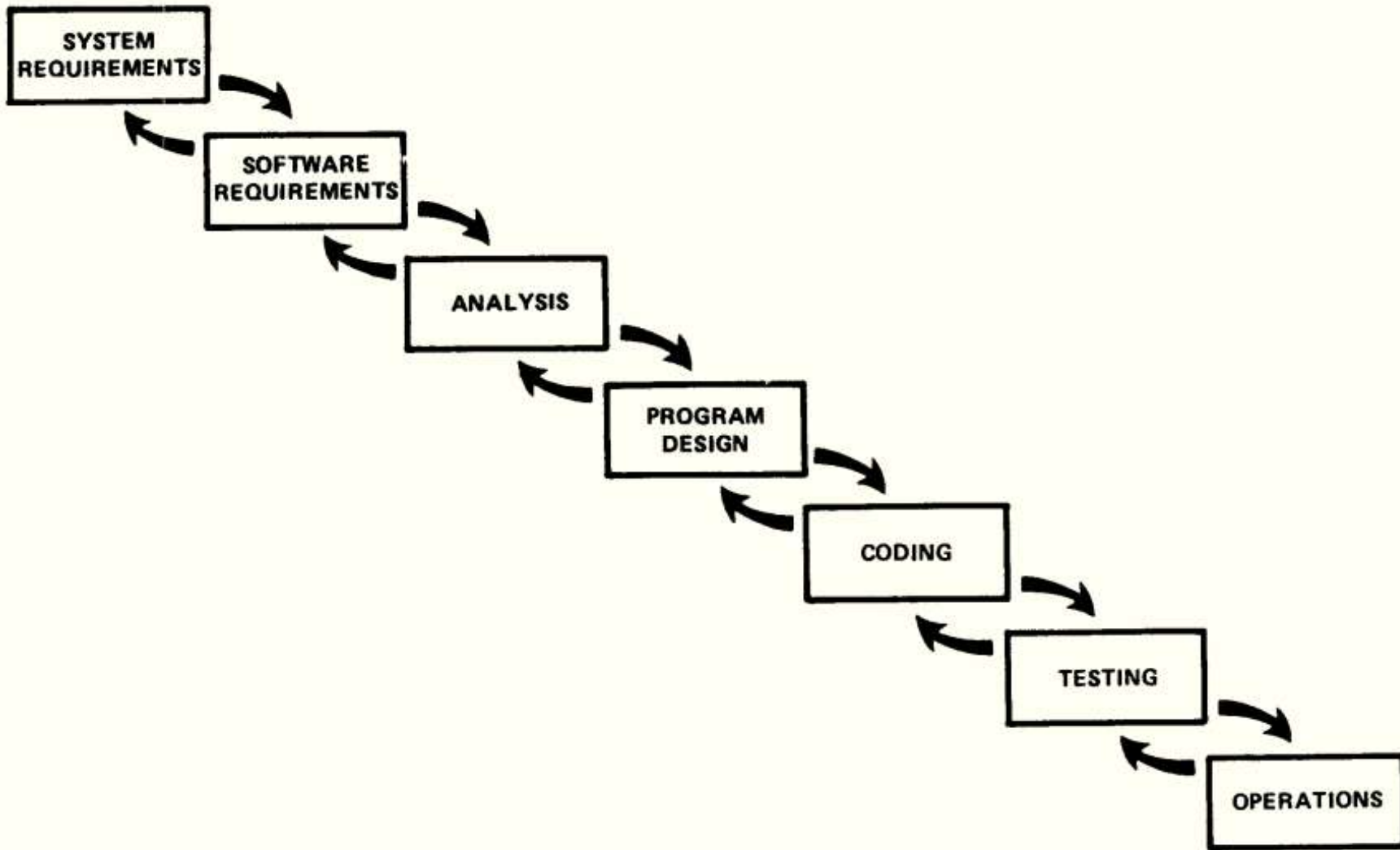
- Find out where you are
- Take a small step towards your goal
- Adjust your understanding based on what you learned
- Repeat
- When faced with two or more alternatives that deliver roughly the same value, take the path that makes future change easier

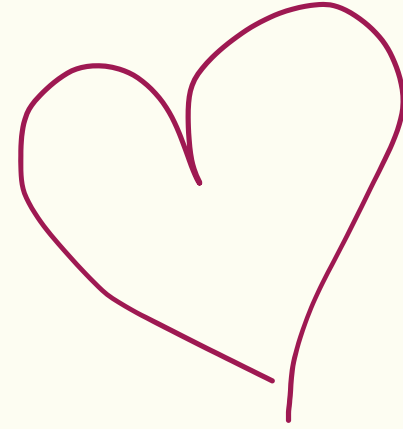


“Agile”

“Agile is Dead, Long Live Agility”: Dave Thomas, GOTO 2015



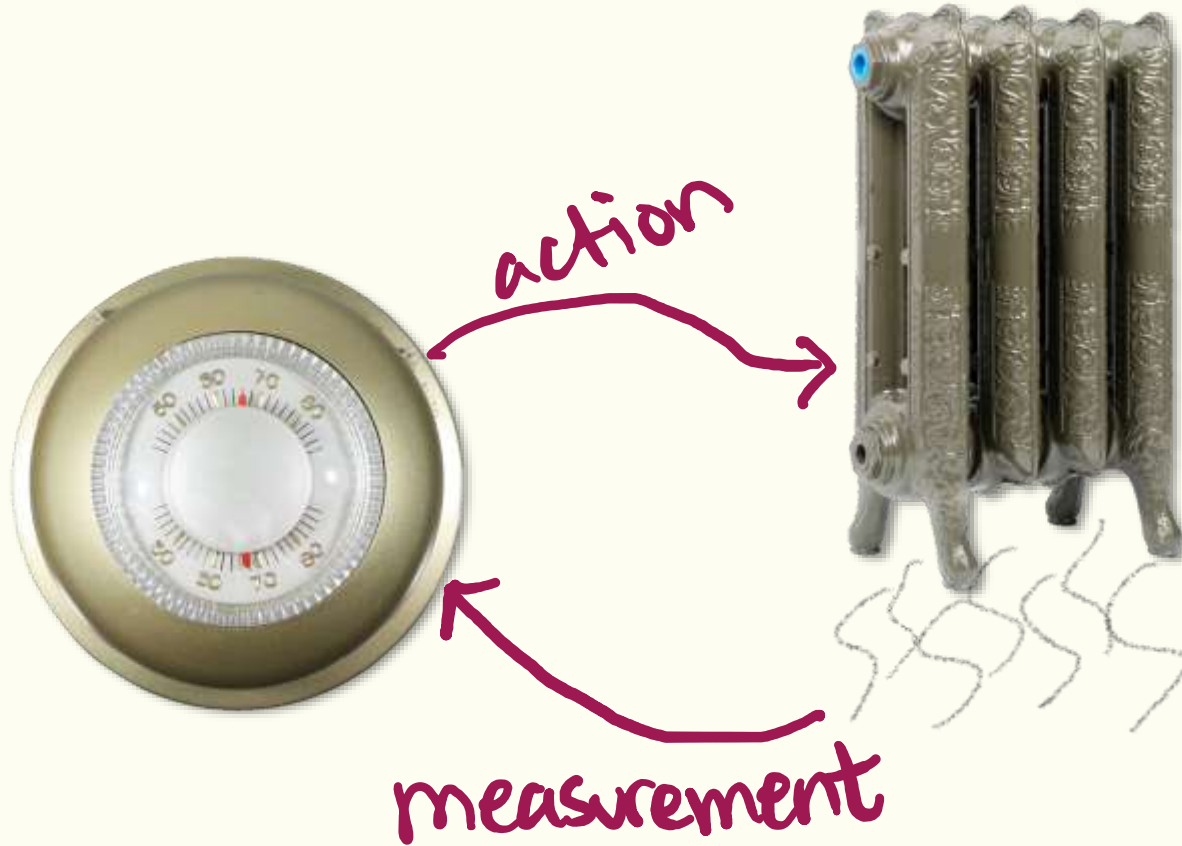




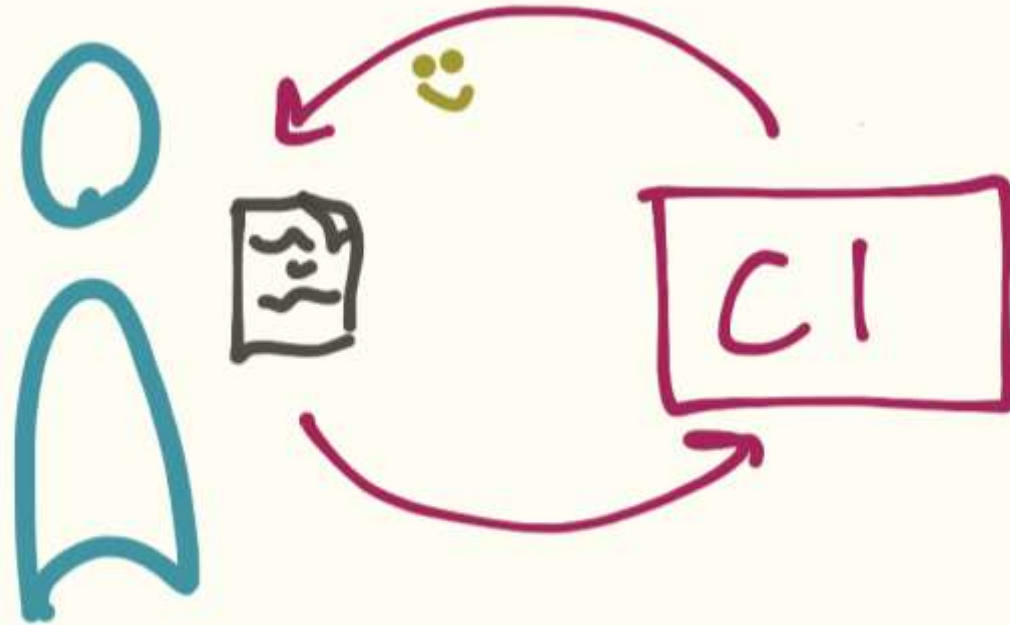
Feedback Loops

We like them!

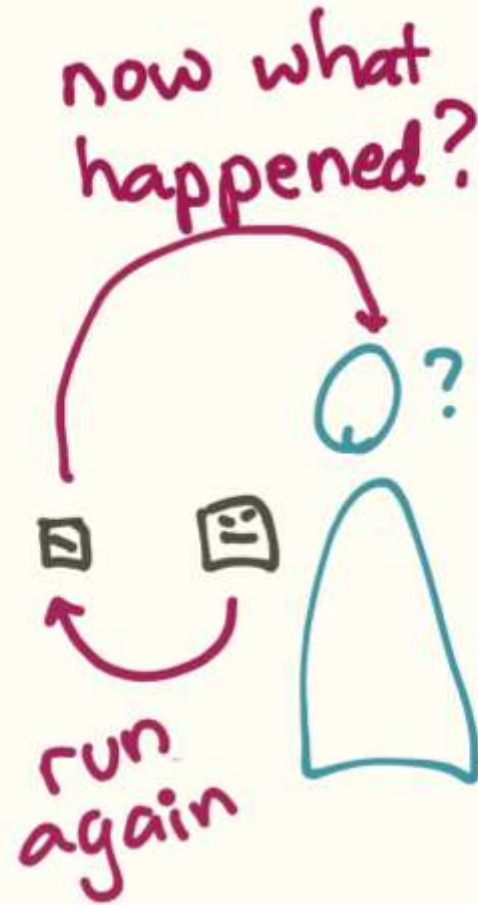
Control Loop



Control Loop



Exploratory Loop



Not all exploratory
loops are created equal



What do good exploratory
loops look like?



“Touch (Not Vision) is the Foundation of Human Cognition”

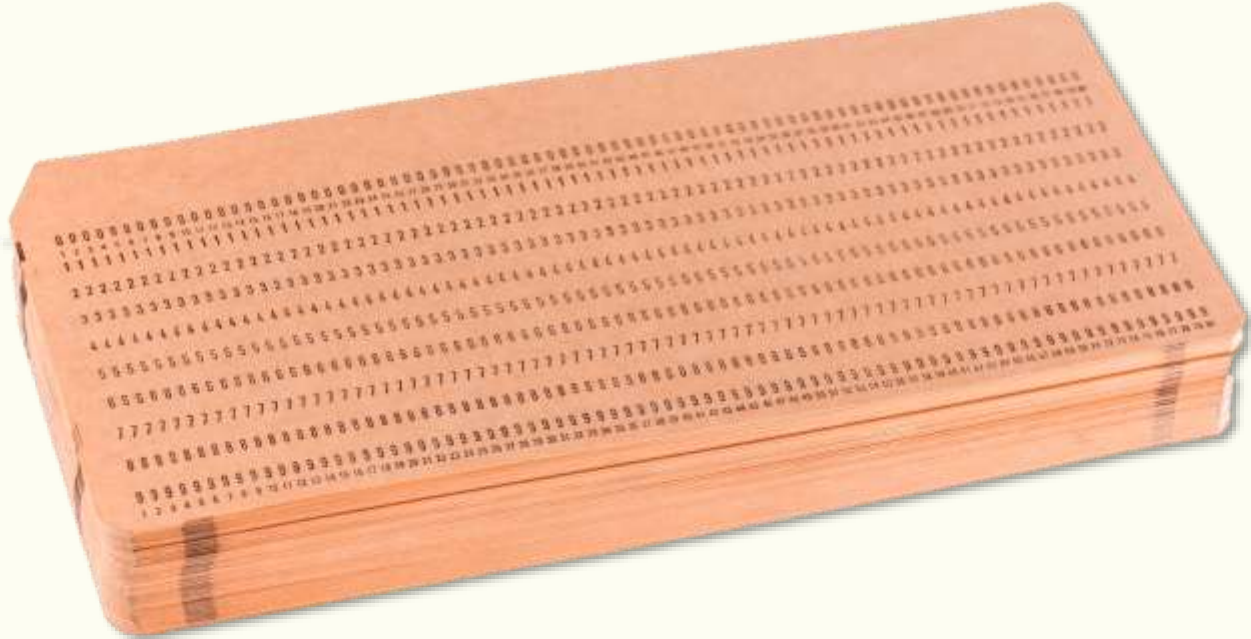
“At the foundation of human cognition is how we interact with our world. Perception is not passive, but rather it is an active conversation and this is fundamentally how we can ‘feel’ the world.”

-- Carlos E. Perez



What do good exploratory
loops **FEEL** like?

Once upon a time...



1964: LISP on the PDP-1



As soon as the basic PDP-1 LISP system is read into the computer, control stops at register 4. Turn up sense switch 5 for typewriter input; press CONTINUE; and the system enters a waiting loop which causes lamps to light in the program counter, looking like 1335. At this point, the LISP system is ready for manual typewriter input. As soon as the operator types, for example:

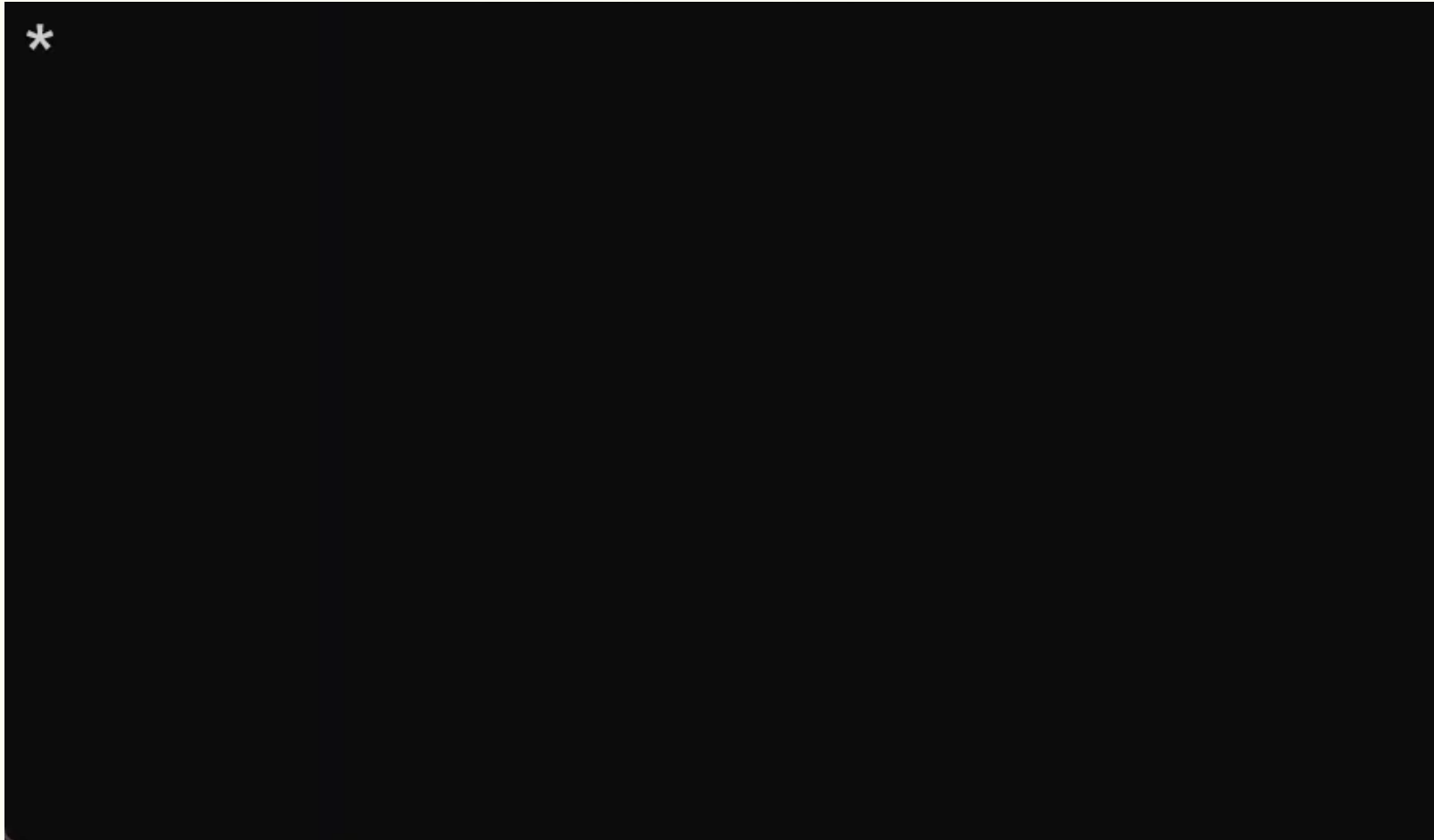
```
(CAR (QUOTE (A B C D)))
```

together with a final space at the end of the last right parenthesis, the computer takes control of the typewriter, impulses a carriage return, and then types out:

```
A
```

which of course is the correct answer. Similarly, for the other suggested test sequences in Table 2 below.

Read, Eval, Print, Loop



Late 1960s - Early 1970s: BASIC and the DTSS



**MY
COMPUTER
LIKES
ME ***

by BOB ALBRECHT



***when i speak in BASIC**

Dr. Dobb's Journal of

COMPUTER

Calisthenics **&** **O**rthodontia

Running Light Without Overbyte

Volume One People's Computer Company, Box E, Menlo Park, CA 94025 1976

A Reference Journal
for Users of
Home Computers



Volume One



People's Computer Company

SCR

```
10 PRINT "MY HUMAN UNDERSTANDS ME"  
20 END
```

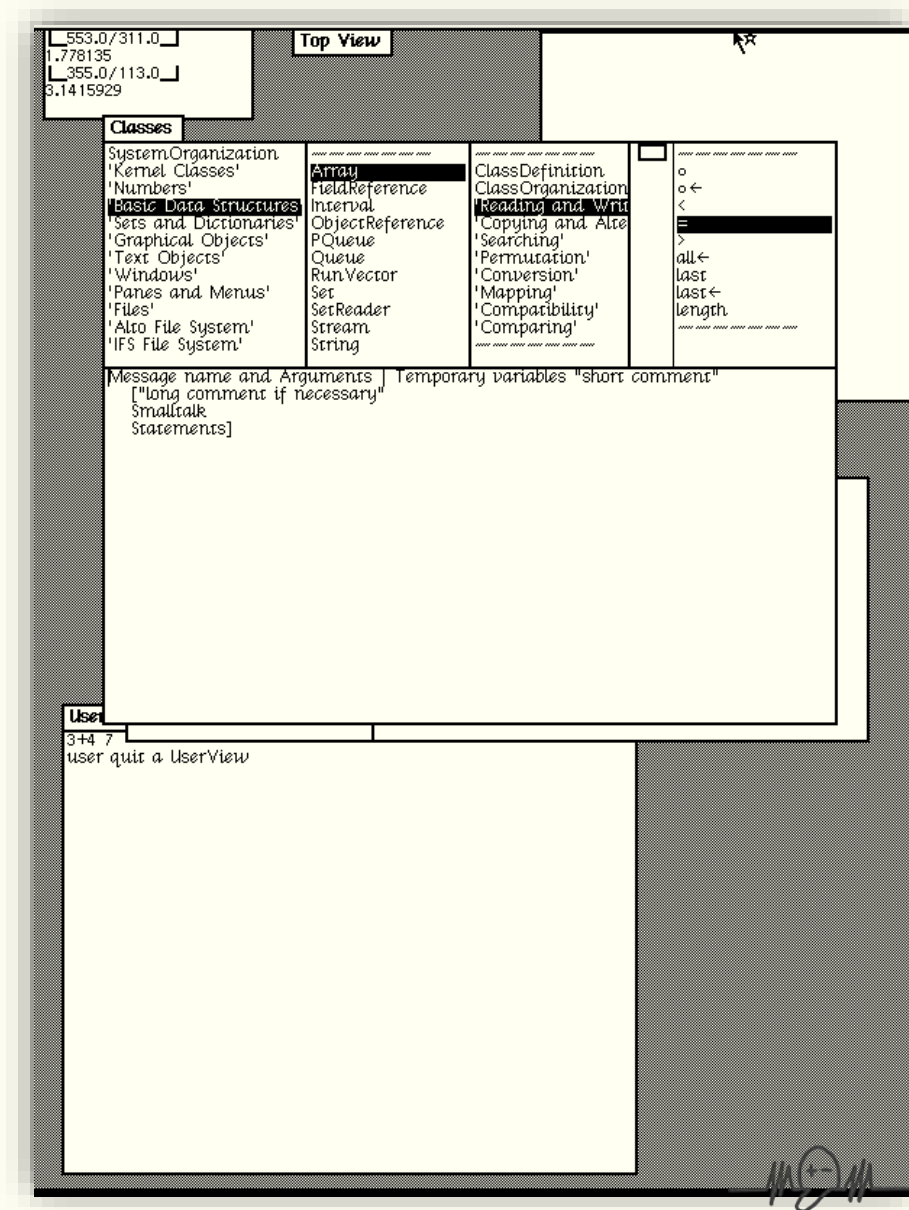
then type

RUN

and the computer will type

MY HUMAN UNDERSTANDS ME

1970s: Smalltalk

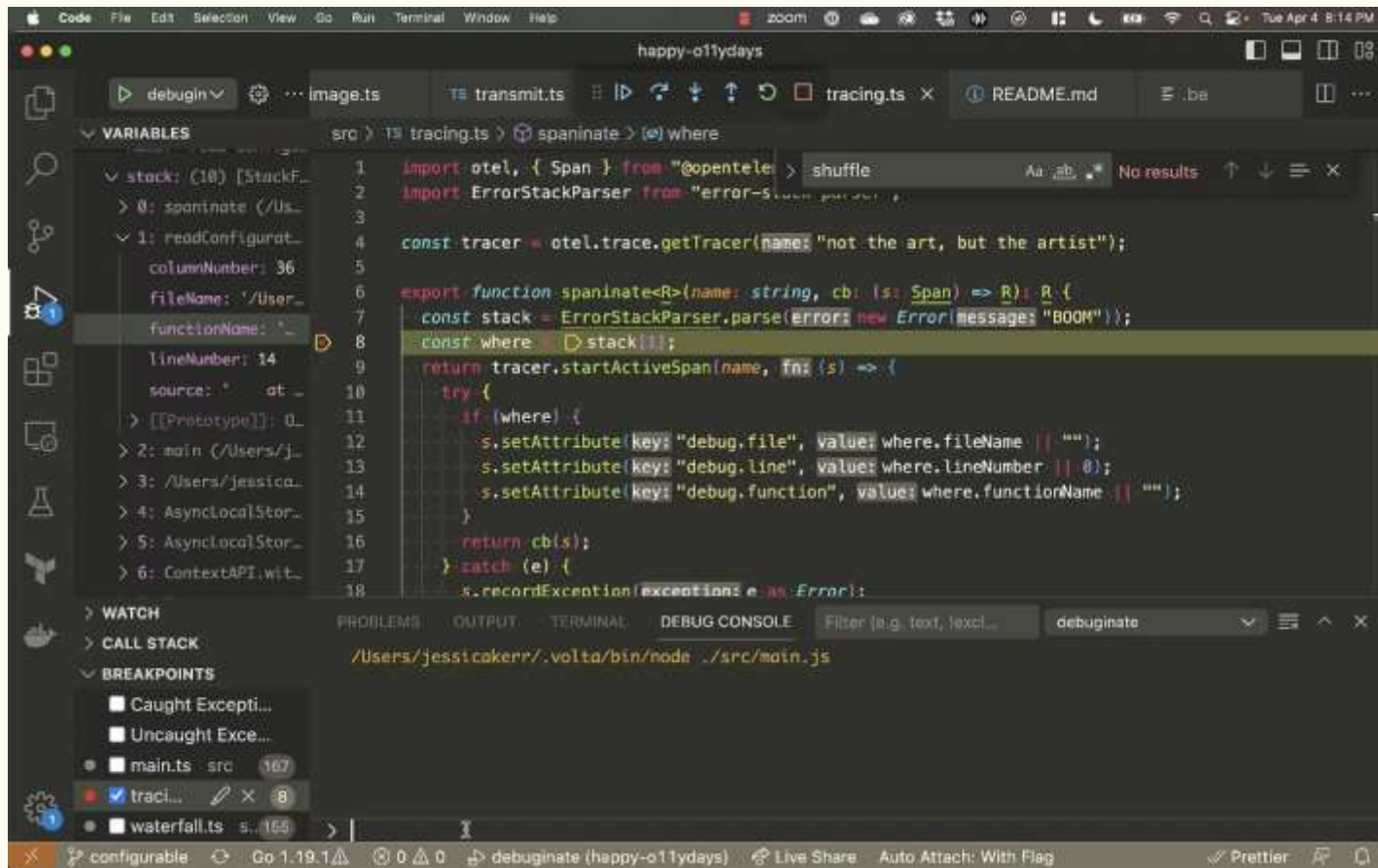


An hourglass with white sand is positioned in the center, resting on a calendar page. The calendar shows dates from 22 to 31. The entire scene is set against a light beige background.

1980-2000: C, C++, Pascal,
Java

Compile and Integration Times Increase

Debugger-Driven Development



~2000: Test-Driven Development!

```
app_test.rb M test/app_test.rb/...
require "+frobozz/app"
require "json"

describe Frobozz::App do
  include Rack::Test::Methods

  You, 22 minutes ago | 1 author (You)
  def app
    Frobozz::App.app
  end

  it "responds to a hello world request" do
    get "/hello"

    assert last_response.ok?
    assert_match(/hello, world/i, last_response.body)
  end
end
```

```
require "+robozz/app"  
require "json"
```

```
describe Frobozz::App do  
  include Rack::Test::Methods
```

You, 22 minutes ago | 1 author (You)

```
  def app  
    Frobozz::App.app  
  end
```

```
  it "responds to a hello world request" do  
    get "/hello"
```

```
    assert last_response.ok?  
    assert_match(/hello, world/i, last_response.body)
```

```
  end
```

You, 22 minutes ago • get started on tests ...

```
end
```

2000s:

“Scripting Languages”

TCL, Scheme, Perl, Python, Ruby, PHP,
JavaScript, Groovy...

TERMINAL

PROBLEMS

50

OUTPUT

DEBUG CONSOLE

PORTS

2

GITLENS

COMMENTS

irb - frobozz + ▾ □ 🗑️ ⋮ ▾ ✕

```
irb(main):007:0> Faker::Name.name  
=> "Earle Donnelly"  
irb(main):008:0> Faker::Name.name  
=> "Garrett Stanton PhD"  
irb(main):009:0> Faker::Name.name  
=> "Pete Feeney"  
irb(main):010:0> █
```

We keep trying to get back to this state of grace

- Because it's approachable
- Because it's satisfying
- Because it's productive
- Because it *feels good*

It feels good because it matches
how humans evolved to think

*“human interaction with software is embedded in **a task-artifact cycle**”*

— John M. Carroll, 1992

Ideas are formed: *“when expressed [in] conversation with...
representations, artifacts, or **objects-to-think with**”*

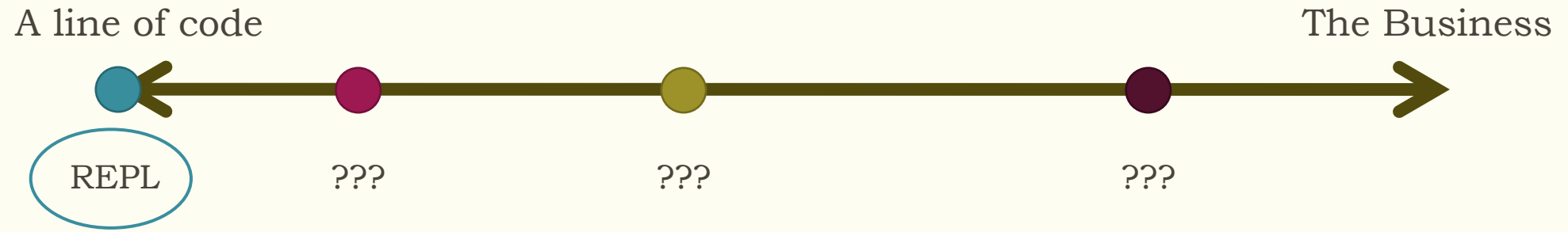
— Edith Ackermann (on Seymour Papert), 2001

**“Knowledge fluidly grows through action taken
in a complex system”**

— Kit Martin (z’1), 2020

REPL: The gold
standard of
exploratory
feedback in-the-
small

Can we have this feeling at larger scopes?

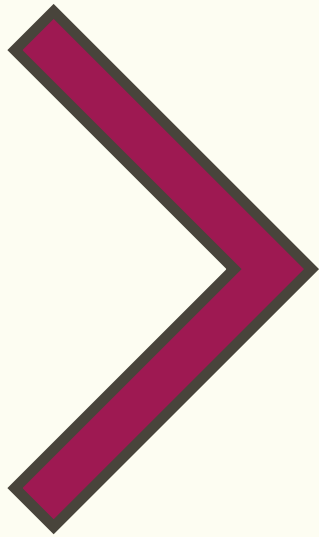


REPL Nature

What qualities
make REPLs feel so
good?

READ





—

There is a prompt, welcoming you

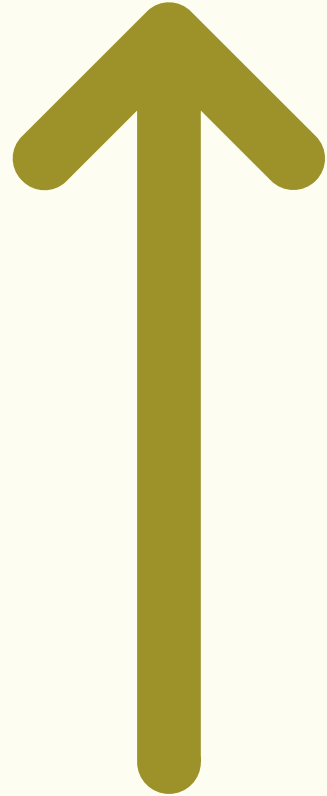
Invitational. Approachable.

What can you enter?
Anything!

It's *Open-ended*

Most REPLs
provide
autocomplete
or suggestions

Discoverable. Serendipitous.



You can
always hit
the up-
arrow

No pausing for
perfection. It's highly
incremental.

REPLs are
SAFE

This is a private,
intimate place to try
silly things.

EVAL



Use the language **you're** used to

No further abstractions.
REPLs are *familiar*,
concrete, and the
knowledge you build is
directly *transferable*.

```
irb:25:in `load'  
irb:25:in `<main>'  
0> Faker::Internet.email  
"avdi@cris.name"  
4:0> DB[:accounts].insert email: Fake  
15:0> DB[:accounts].to_a  
id=>2,  
"avdi@avdi.codes",  
id_hash=>"$2a$04$Z0zJ4xyf2rtgh18g0i3J..l  
:status_id=>1, :email=>"jamison.ledner@  
016:1* 100.times do  
017:1*   DB[:accounts].insert email: Fa  
018:0> end  
019:0> █
```

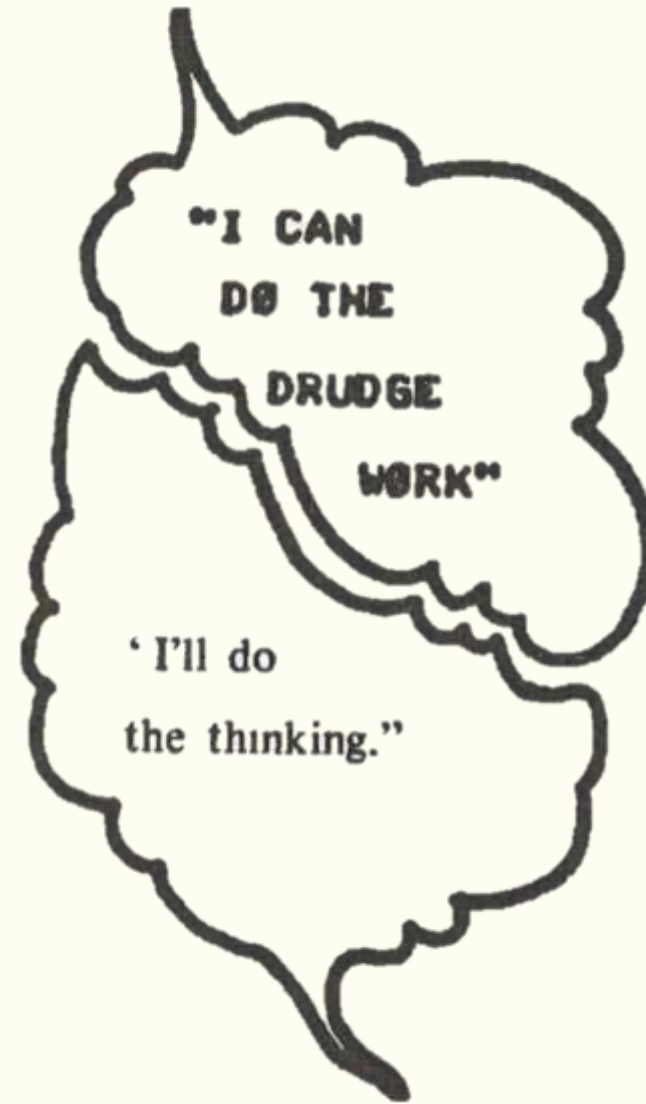


Poke the
system with
a stick

Don't just ask
questions, *perturb*
the system and
see what falls out.



REPLs are
AUTOMANUAL



PRINT



```
irb(main):012:0> Faker::Email
```

```
(irb):12:in `<main>': uninitialized constant Faker::Email (NameError)
```

```
  from /usr/local/lib/ruby/gems/3.2.0/gems/irb-1.6.2/exe/irb:11:in `<top (required)>'
```

```
  from /usr/local/bin/irb:25:in `load'
```

```
  from /usr/local/bin/irb:25:in `<main>'
```



Multi-Channel

Rich output: Not just the answers we asked for, but `STDOUT`, `STDERR`, and caught exceptions.

```
{:id=>73, :status_id=>1, :email=>"josphine.wilkinson@schuppe.io", :password_hash=>nil},
{:id=>74, :status_id=>1, :email=>"jarred@langworth.io", :password_hash=>nil},
{:id=>75, :status_id=>1, :email=>"reinaldo@veum.io", :password_hash=>nil},
{:id=>76, :status_id=>1, :email=>"stefania@hegmann.biz", :password_hash=>nil},
{:id=>77, :status_id=>1, :email=>"ron@wyman.com", :password_hash=>nil},
{:id=>78, :status_id=>1, :email=>"josue@schoen.co", :password_hash=>nil},
{:id=>79, :status_id=>1, :email=>"edison@miller.org", :password_hash=>nil},
{:id=>80, :status_id=>1, :email=>"jeanine_lehner@boyle.info", :password_hash=>nil},
{:id=>81, :status_id=>1, :email=>"erin.cummerata@sporer.info", :password_hash=>nil},
{:id=>82, :status_id=>1, :email=>"ward.hayes@dickinson.io", :password_hash=>nil},
{:id=>83, :status_id=>1, :email=>"joanna@parker.org", :password_hash=>nil},
{:id=>84, :status_id=>1, :email=>"joanna_labadie@goldner-murphy.net", :password_hash=>nil},
{:id=>85, :status_id=>1, :email=>"debbie@cremin.co", :password_hash=>nil},
{:id=>86, :status_id=>1, :email=>"leanora.marks@russel.net", :password_hash=>nil},
{:id=>87, :status_id=>1, :email=>"dusty.cartwright@baumbach-vandervort.org", :password_hash=>nil},
{:id=>88, :status_id=>1, :email=>"oswaldo@crooks-shields.org", :password_hash=>nil},
{:id=>89, :status_id=>1, :email=>"kiana_schultz@kautzer.biz", :password_hash=>nil},
{:id=>90, :status_id=>1, :email=>"kiana_schultz@kautzer.biz", :password_hash=>nil},
{:id=>91, :status_id=>1, :email=>"donny@hamill-durgan.net", :password_hash=>nil},
{:id=>92, :status_id=>1, :email=>"irvin@zulauf-koch.co", :password_hash=>nil},
{:id=>93, :status_id=>1, :email=>"kip@leannon.net", :password_hash=>nil},
{:id=>94, :status_id=>1, :email=>"henry_goyette@ritchie-kihn.name", :password_hash=>nil},
{:id=>95, :status_id=>1, :email=>"byron@stehr-hane.info", :password_hash=>nil},
{:id=>96, :status_id=>1, :email=>"ethelyn_funk@kunze.com", :password_hash=>nil},
{:id=>97, :status_id=>1, :email=>"waylon_bednar@casper-simonis.co", :password_hash=>nil},
{:id=>98, :status_id=>1, :email=>"brendon@reichel.co", :password_hash=>nil},
{:id=>99, :status_id=>1, :email=>"agripina.leannon@oconnell-bosco.com", :password_hash=>nil},
{:id=>100, :status_id=>1, :email=>"angel_simonis@mosciski-kshlerin.biz", :password_hash=>nil},
{:id=>101, :status_id=>1, :email=>"lauryn.mills@olson.org", :password_hash=>nil},
{:id=>102, :status_id=>1, :email=>"marcos_labadie@pfeffer.net", :password_hash=>nil}]
```

Arbitrary length output

Enables *gradual refinement*.

Scroll-back history


A story, but an
ephemeral one. A
narrative.



```
irb(main):015:0> DB[:accounts].to_a  
=>  
[{:id=>1,  
  :status_id=>2,  
  :email=>"avdi@avdi.codes",  
  :password_hash=>"$2a$04$Z0zJ4xyf2rtgh18g0i3J.  
{:id=>2, :status_id=>1, :email=>"jamison.ledne
```

Serialized
data
structures

This is *exposed ductwork*: it gives you a clue about further questions you might ask.

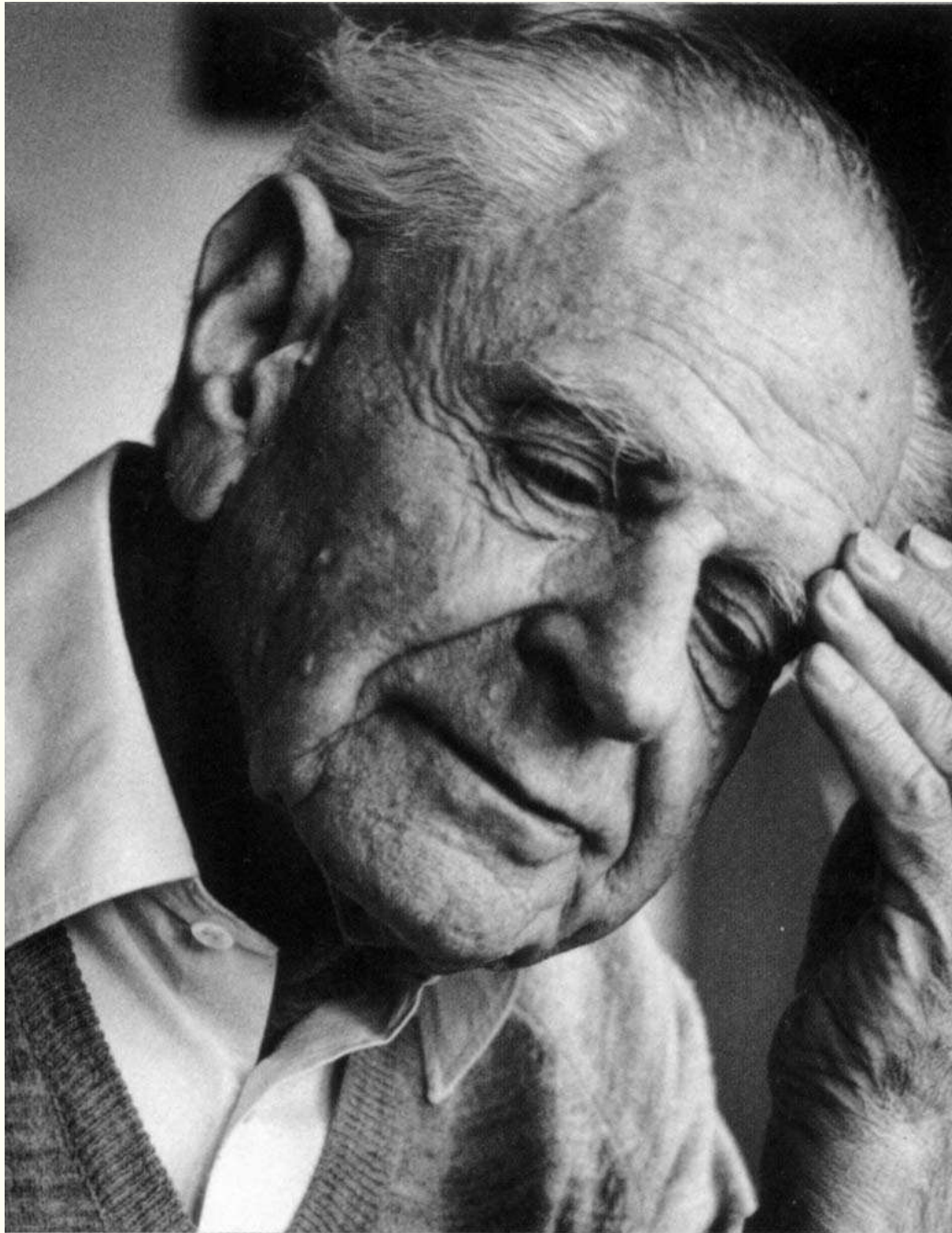
A red balloon floats in the center of the frame against a grey, overcast sky. The balloon is positioned between two tall, multi-story brick buildings that line a city street. The buildings have many windows and air conditioning units. The overall scene is a classic urban street view.

You get stuff you
didn't expect

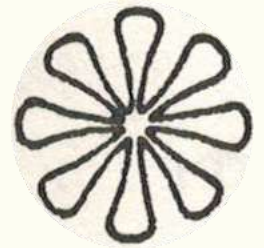
The REPL is open to surprise.



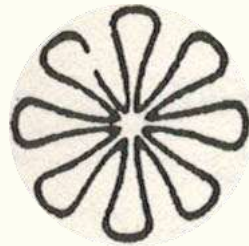
Are REPLs science?



Science progresses
with falsifiable
theories.



**SCIENCE PROGRESSES
WITH SURPRISE!**





REPLs are
EXPLORAMENTAL!

LOOP



Try Again



REPLs have infinite depth





REPLS
build up
context
over time

Variables, helpers,
changes to the data.



REPL Interactions are Lossy

REPLs are fast

This is not “must go faster”



REPLs ~~are fast~~ move
at the speed of your
questions

REPLs are *responsive*:
they provide a pacing of
iteration appropriate to
the scope.



REPLs are
CONVERSATIONAL

REPL NATURE

READ

- Invitational
- Approachable
- Open-Ended
- Discoverable
- Incremental
- Private

Safe

EVAL

- Familiar
- Transferable
- Concrete
- Reflective
- Reactive

Automanual

PRINT

- Ductwork Exposed
- Gradually Refined
- Rich
- Narrative
- Open to Surprise

Exploramental

LOOP

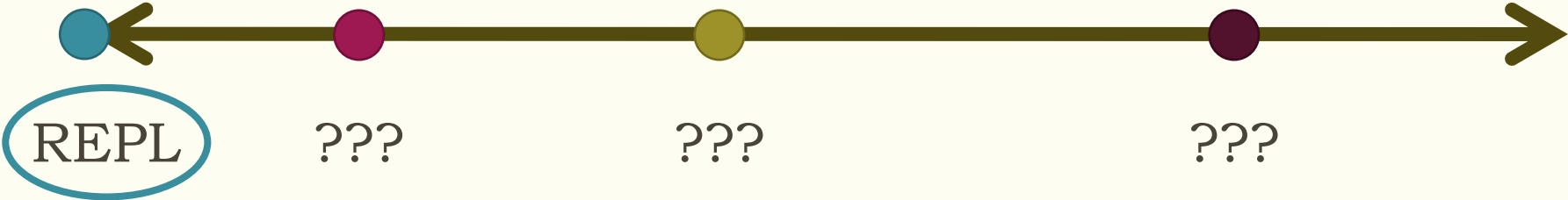
- Contextual
- Ongoing
- Infinitely Deep
- Lossy
- Responsive

Conversational

Movin' on up...

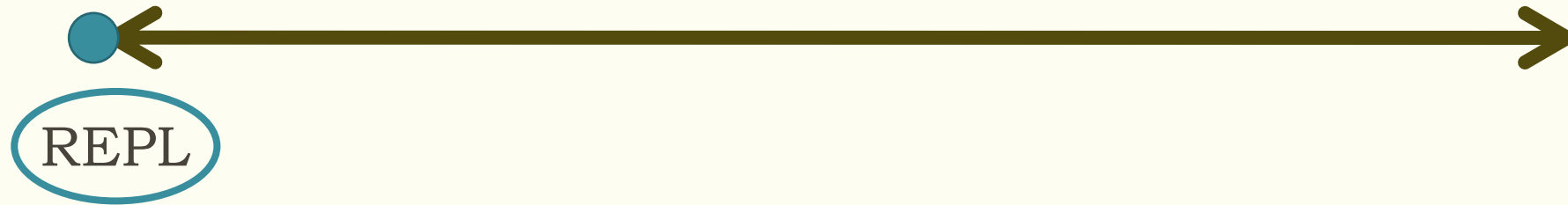
A line of code

The Business



A line of code

The Business



REPL

TERMINAL PROBLEMS 50 OUTPUT DEBUG CONSOLE PORTS 2 GITLENS COMMENTS

irb - frobozz + - - - X

```
irb(main):007:0> Faker::Name.name  
=> "Earle Donnelly"  
irb(main):008:0> Faker::Name.name  
=> "Garrett Stanton PhD"  
irb(main):009:0> Faker::Name.name  
=> "Pete Feeney"  
irb(main):010:0> █
```

A line of code

The Business



A Unit

Practice test-driven design

```
app_test.rb M test/app_test.rb/...
require "frobozz/app"
require "json"

describe Frobozz::App do
  include Rack::Test::Methods

  You, 22 minutes ago | 1 author (You)
  def app
    Frobozz::App.app
  end

  it "responds to a hello world request" do
    get "/hello"

    assert last_response.ok?
    assert_match(/hello, world/i, last_response.body)
  end
  You, 22 minutes ago • get started on tests _
end
```

Read your test failures and stack traces

```
include Rack::Test::Methods

You, 26 minutes ago | 1 author (You)
def app
  Frobozz::App.app
end
```

```
it "responds to a hello world request" do
  get "/hello"
```

1) Failure:

```
Frobozz::App#test_0002_saves and returns new rooms [/workspaces/frobozz/test/app_test.rb:26]:
```

```
Expected: [{:name=>"Kitchen"}]
```

```
Actual: {"name"=>"Kitchen"}
```

```
2 runs, 4 assertions, 1 failures, 0 errors, 0 skips
```

```
rake aborted!
```

```
Command failed with status (1): [/usr/local/bin/ruby -Ilib:test:. -w -e 're...]
```

```
/usr/local/rvm/gems/default/gems/minitest-5.18.0/lib/minitest/test_task.rb:164:in `block in define'
```

```
/usr/local/rvm/gems/default/gems/rake-13.0.6/exe/rake:27:in `'
```

```
irb frobozz
```

```
rake: tes... ❌
```

A line of code

The Business



A Feature

Beware of Sherlock Holmes



Tracing

Another view. Below is a **TRACE** of the program. (*Trace? Sure! A TRACE traces the path the computer takes through the program it is working on, and shows what values are assigned to the variables at any step in the program. Ain't it obvious?*) In the column marked **N** we show the value of **N** after the statement on the same line has been carried out by the computer.

STATEMENT	N	REMARKS

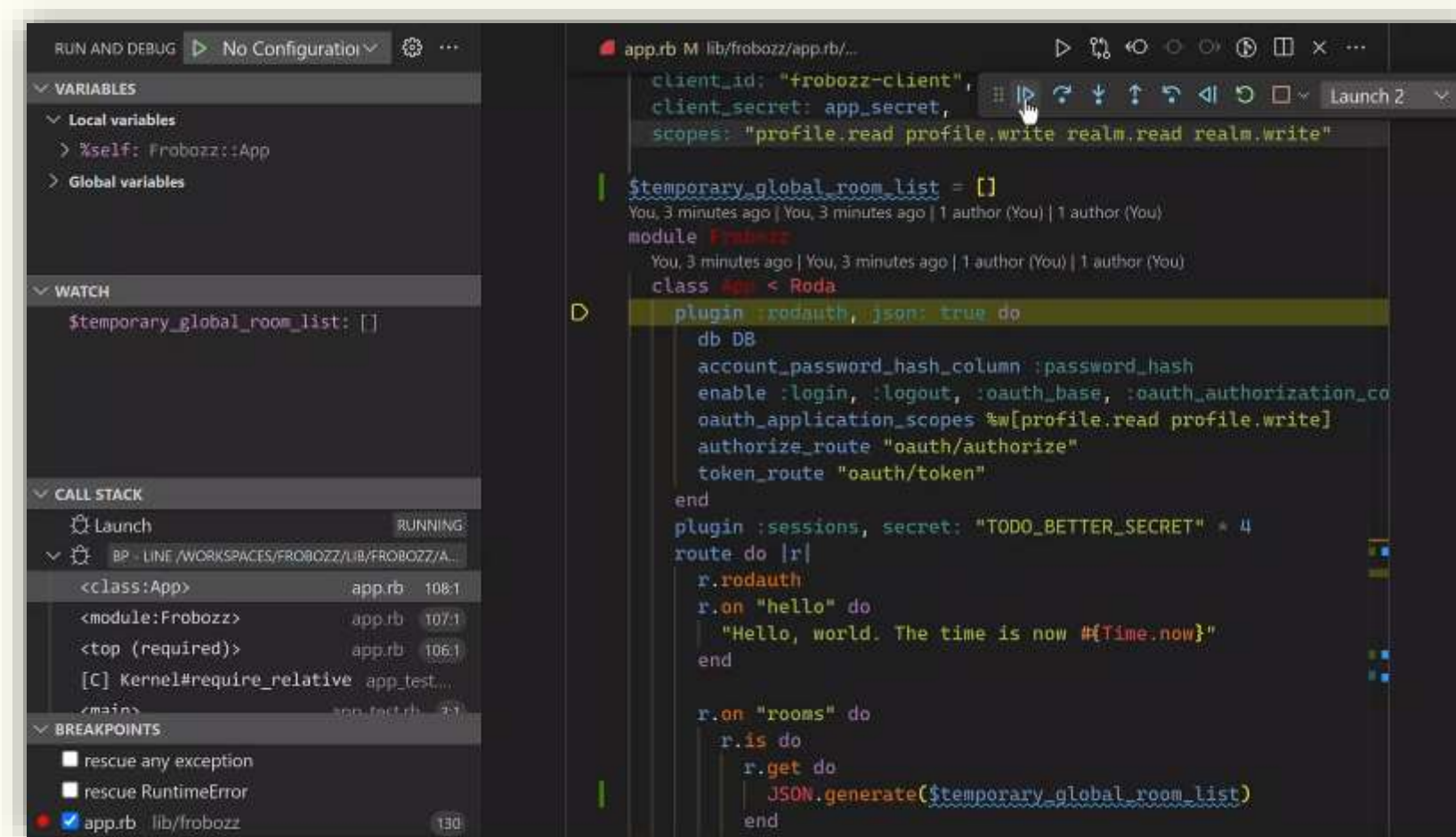
10 LET N=1	1	
20 PRINT N	1	Print the value of N.
30 LET N=N+1	2	Increase N by 1 (add 1 to the old value of N).
40 GO TO 20	2	Go to beginning of loop.
20 PRINT N	2	Print the value of N.
30 LET N=N+1	3	Increase N by 1.
40 GO TO 20	3	Go to beginning of loop.
20 PRINT N	3	Print the value of N.
30 LET N=N+1	4	Increase N by 1.
40 GO TO 20	4	Go to beginning of loop.

```
root@9677044702e7:/workspaces/examples# █
```



```
root@9677044702e7:/workspaces/examples# strace -e open bundle 2>&1 | less
root@9677044702e7:/workspaces/examples# strace -f -e open bundle 2>&1 | less
root@9677044702e7:/workspaces/examples# strace -f -e openat bundle 2>&1 | less
```

Use the debugger (judiciously)



RUN AND DEBUG

No Configuration



VARIABLES

Local variables

> %self: Frobozz::App

Global variables

WATCH

\$temporary_global_room_list: []

CALL STACK

Launch RUNNING

BP - LINE /WORKSPACES/FROBOZZ/LIB/FROBOZZ/A...

<class:App> app.rb 108:1

<module:Frobozz> app.rb 107:1

<top (required)> app.rb 106:1

[C] Kernel#require_relative app_test...

<main> app_test.rb 3:1

BREAKPOINTS

 rescue any exception rescue RuntimeError app.rb lib/frobozz

130

app.rb M lib/frobozz/app.rb/...



```
client_id: "frobozz-client",
client_secret: app_secret,
scopes: "profile.read profile.write realm.read realm.write"
```

```
$temporary_global_room_list = []
```

You, 3 minutes ago | You, 3 minutes ago | 1 author (You) | 1 author (You)

```
module Frobozz
```

You, 3 minutes ago | You, 3 minutes ago | 1 author (You) | 1 author (You)

```
class App < Roda
```

```
  plugin :rodauth, json: true do
```

```
    db DB
```

```
    account_password_hash_column :password_hash
```

```
    enable :login, :logout, :oauth_base, :oauth_authorization_co
```

```
    oauth_application_scopes %w[profile.read profile.write]
```

```
    authorize_route "oauth/authorize"
```

```
    token_route "oauth/token"
```

```
  end
```

```
  plugin :sessions, secret: "TODO_BETTER_SECRET" * 4
```

```
  route do |r|
```

```
    r.rodauth
```

```
    r.on "hello" do
```

```
      "Hello, world. The time is now #{Time.now}"
```

```
    end
```

```
    r.on "rooms" do
```

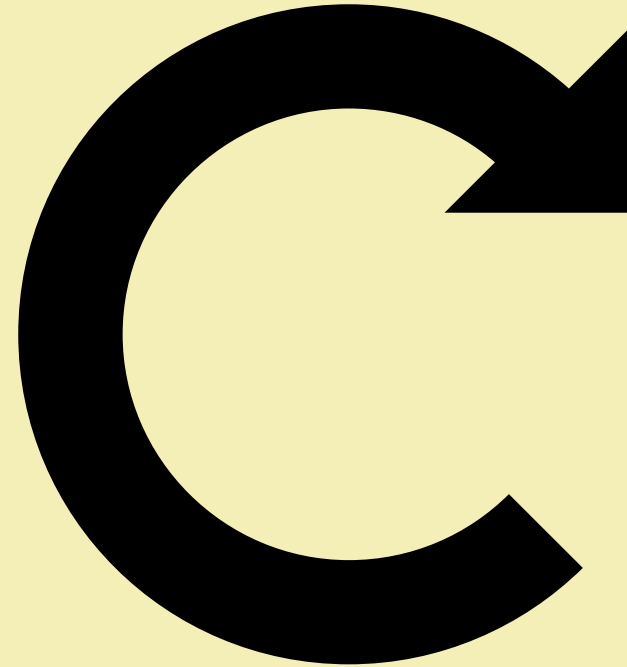
```
      r.is do
```

```
        r.get do
```

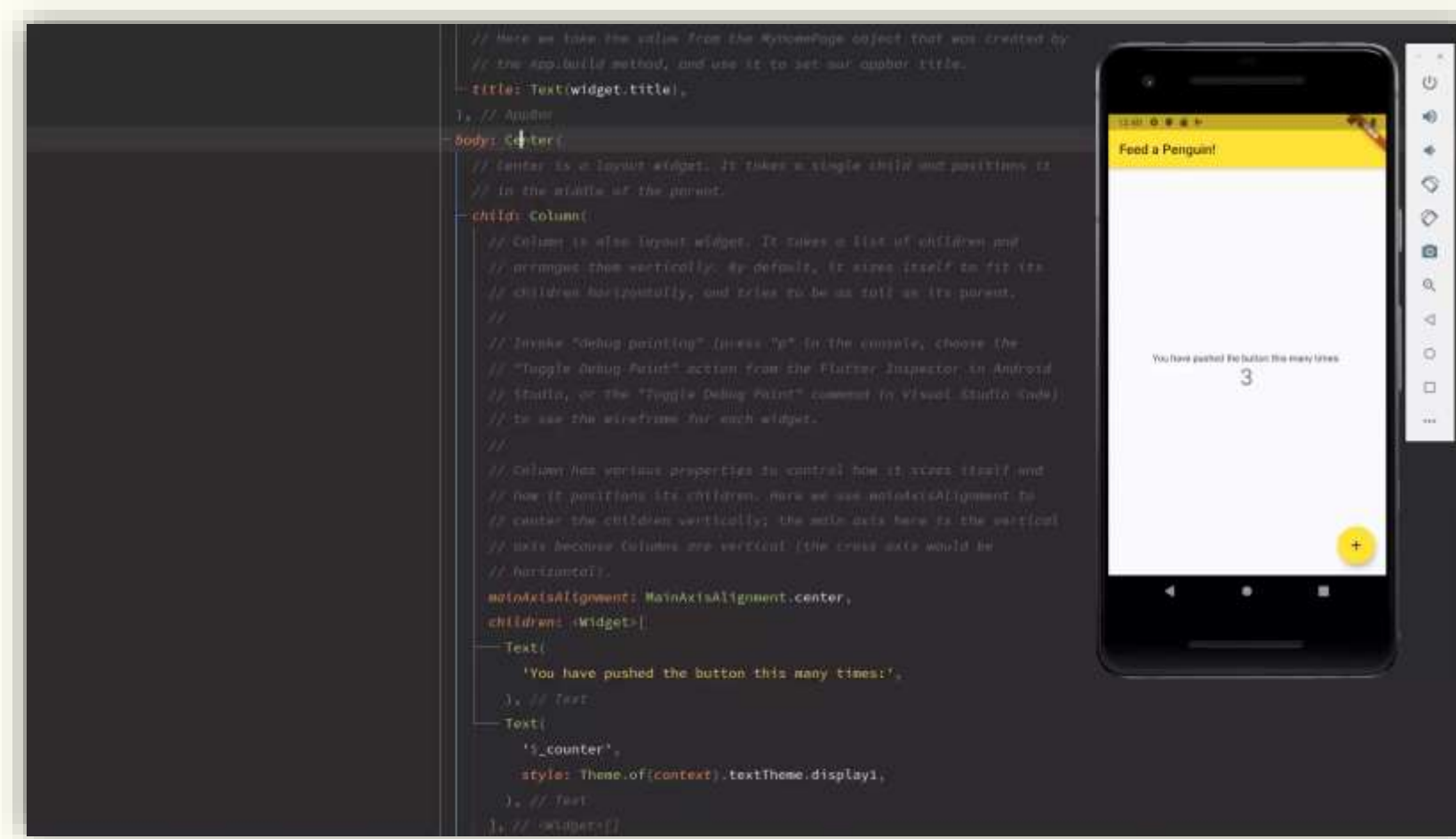
```
          JSON.generate($temporary_global_room_list)
```

```
        end
```

Reload-driven
development



Leverage hot-reload



A line of code

The Business



A Service

```
1 select * from wp_postmeta where meta_key like "wpf%";
2 |
3
```

gracefuldev: select * from wp... | gracefuldev: select * from wp... | gracefuldev: select * from wp... X

meta_id	post_id	meta_key	meta_value
126997	28154	wpf_settings_buddypress	a:1:{s:10:"apply_tags";a:1:{i:0;s:26:"gracefuldev:grouptype:team";}}
127009	28155	wpf_settings_buddypress	a:1:{s:10:"apply_tags";a:1:{i:0;s:30:"gracefuldev:grouptype:interest";}}
127021	28156	wpf_settings_buddypress	a:1:{s:10:"apply_tags";a:1:{i:0;s:29:"gracefuldev:grouptype:special";}}
128534	28322	wpf_settings_buddypress	a:0:[]
128548	28323	wpf_settings_buddypress	a:1:{s:8:"tag_link";a:1:{i:0;s:34:"gracefuldev:profiletype:instructor";}}
128562	28324	wpf_settings_buddypress	a:1:{s:8:"tag_link";a:1:{i:0;s:34:"gracefuldev:profiletype:pro-member";}}
109443	25668	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109637	26169	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109670	26172	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109711	26186	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109734	26193	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109774	26203	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109809	26209	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109978	25670	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
109997	25679	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
110016	25681	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}
110035	25683	wpf-settings	a:4:{s:8:"redirect";i:0;s:12:"redirect_url";s:0:"";s:11:"apply_delay";i:0;s:12:"lock_content";b:0;}

Make friends with SQL

A yellow shipping container is shown against a blue sky with a white cloud. The container has a corrugated metal surface and a door on the left side. The text "Develop in containers" is overlaid on the container in a white box with a blue border.

Develop in
containers

A line of code

The Business



An Application

Automate
scenario
reproduction

```
options:  
  courses:  
    shared_steps: true  
lessons:  
  - name: Lesson A  
  - name: Lesson B  
  - name: Shared Lesson  
  primary_course: Course A  
courses:  
  - name: Course A  
    lessons:  
      - Lesson A  
      - Shared Lesson  
  - name: Course B  
    lessons:  
      - Lesson B  
      - Shared Lesson  
students:  
  - name: Avdi  
    current_step: Lesson B
```

Automate
scenario
reproduction

**This does NOT imply
automating end-to-
end tests!*

```
options:  
  courses:  
    shared_steps: true  
lessons:  
  - name: Lesson A  
  - name: Lesson B  
  - name: Shared Lesson  
  primary_course: Course A  
courses:  
  - name: Course A  
    lessons:  
      - Lesson A  
      - Shared Lesson  
  - name: Course B  
    lessons:  
      - Lesson B  
      - Shared Lesson  
students:  
  - name: Avdi  
    current_step: Lesson B
```

Practice
exploratory
testing

The
Pragmatic
Programmers

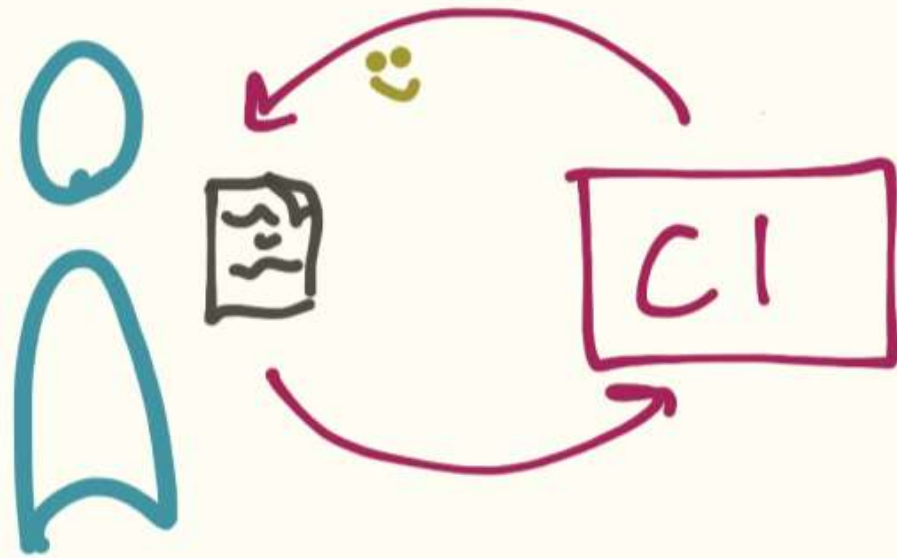
Explore It!

Reduce Risk and
Increase Confidence with
Exploratory Testing



Elisabeth Hendrickson

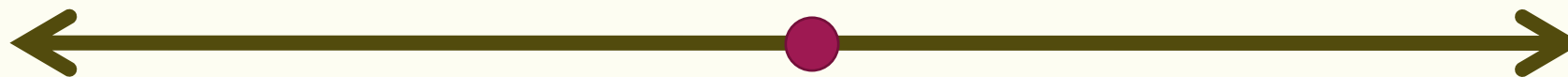
Edited by Jacquelyn Carter



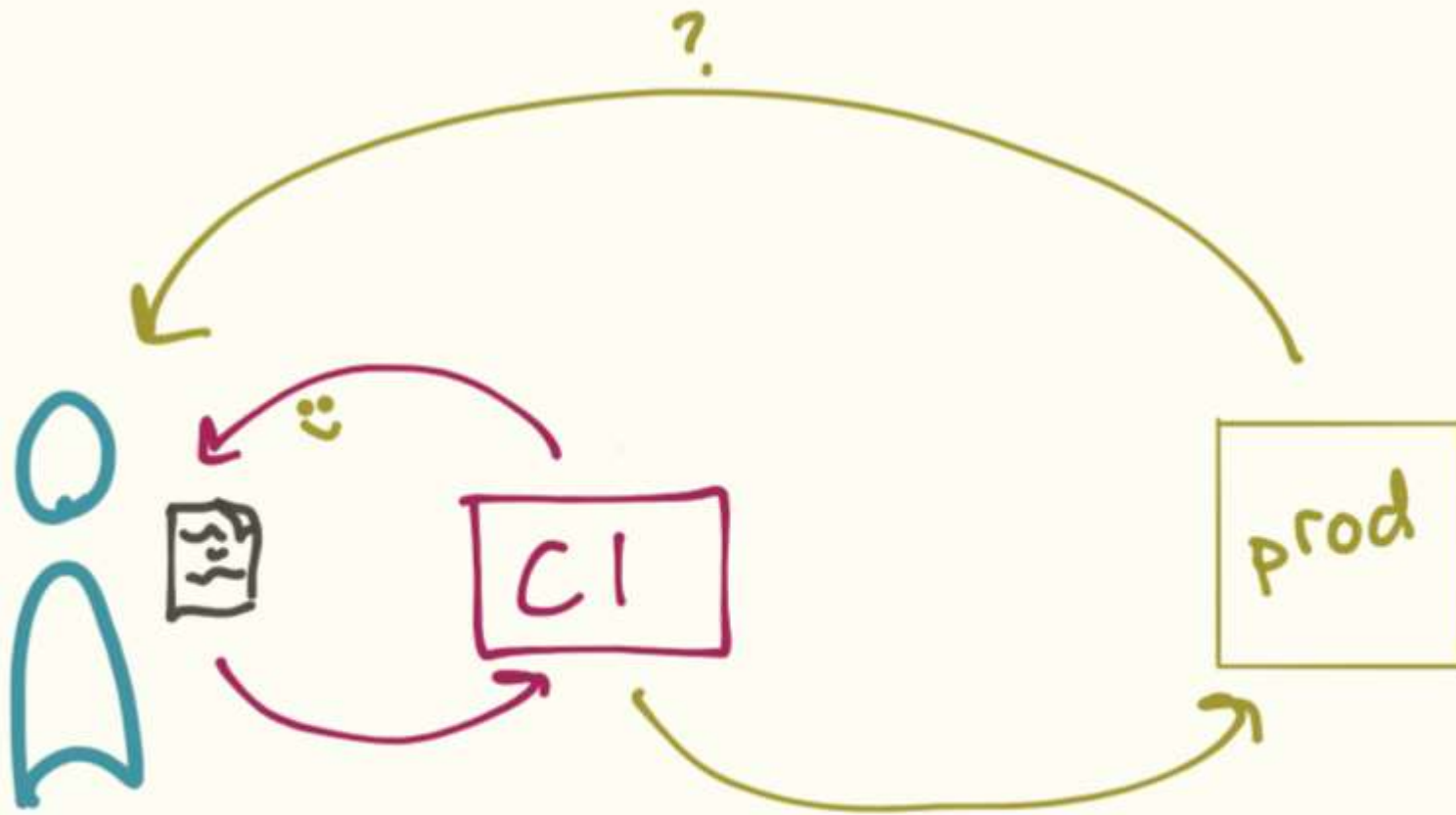
Use
continuous
integration

A line of code

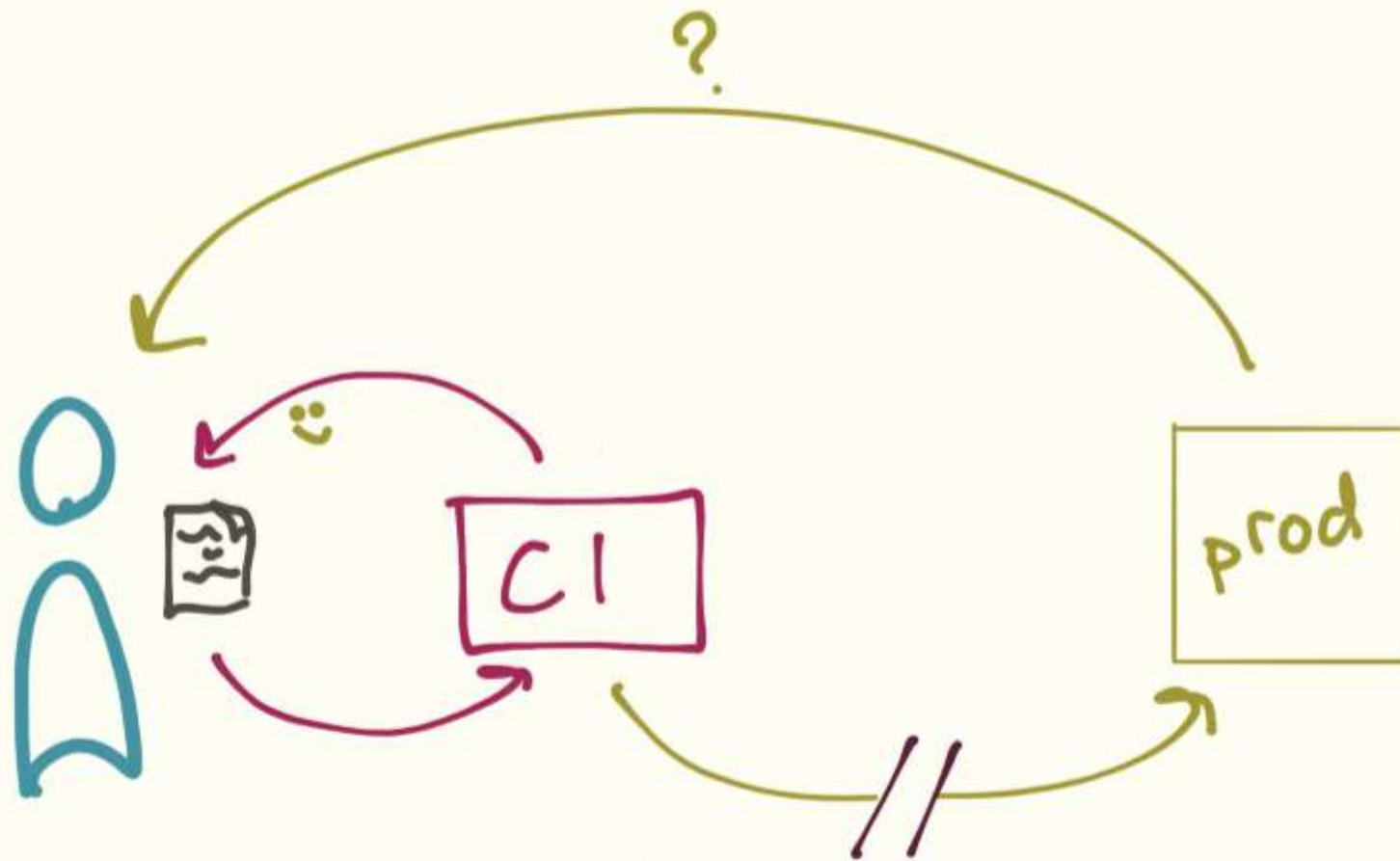
The Business



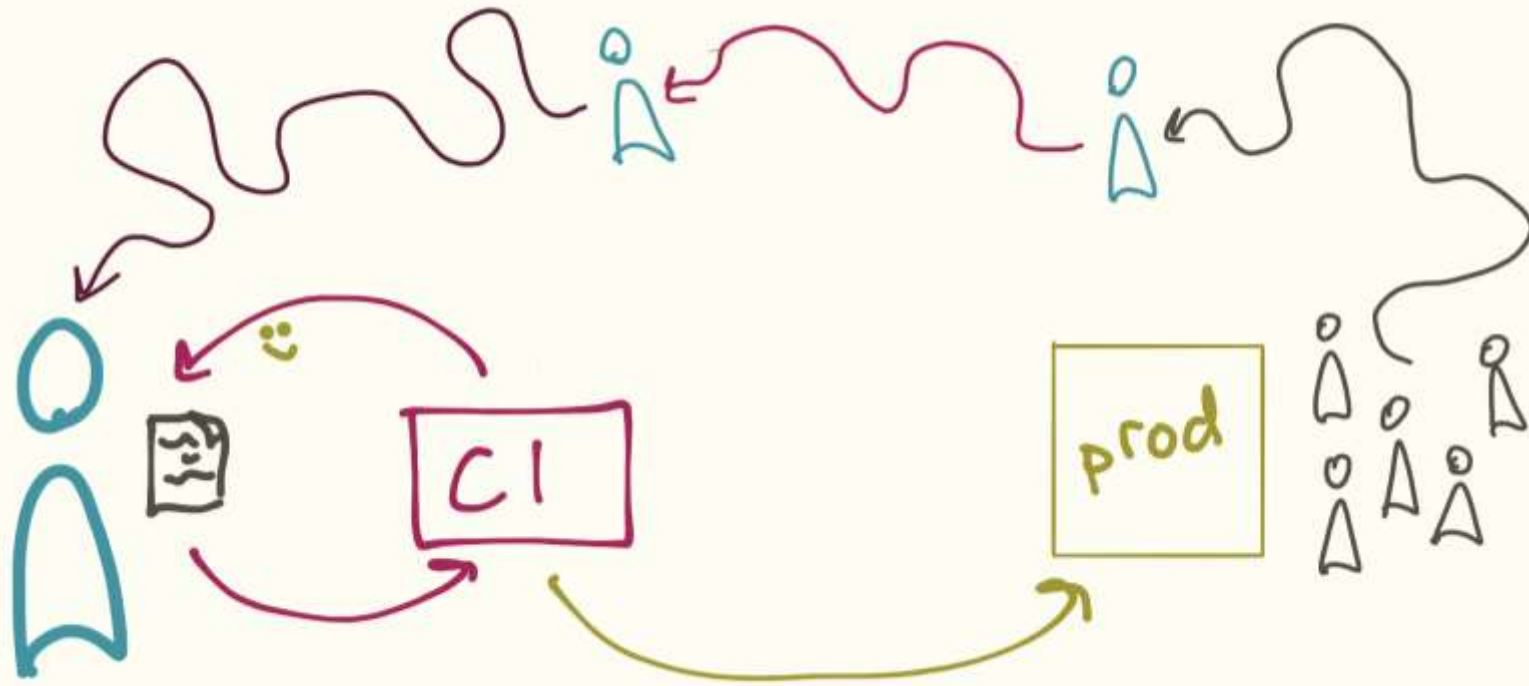
App-In-Production



Use
continuous
deployment



...but beware!



...also
beware!

The screenshot displays the Honeycomb web interface for creating a new query. The browser's address bar shows the URL `ui.honeycomb.io/modernity/environments/demo`. The left sidebar contains navigation links: Home, New Query, Datasets, Boards, History, Triggers, SLOs, Service Map, Search, Usage, and Account. The main content area is titled "New Query" and includes a "VISUALIZE" section with "COUNT" and "HEATMAP(duration_ms)", a "WHERE" clause with "trace.parent_id does-not-exist", and a "GROUP BY" dropdown menu. The dropdown menu is open, showing a list of column names: "name", "db.name", "filename", "host.name", "http.server_name", "instrumentation_scope.name", "k8s.cluster.name", "k8s.deployment.name", "k8s.namespace.name", "k8s.node.name", and "Create derived column". A "Run Query" button is located on the right side of the interface.

Use distributed tracing

All datasets in demo

Last 2 hours

New Query

VISUALIZE

COUNT HEATMAP(duration_ms)

WHERE

trace.parent_id does-not-exist

AND

GROUP BY

name

Run Query

Clear | Cancel

+ ORDER BY

+ LIMIT

Suggested Queries

Overall distribution of latencies

Slowest traces

Trace volume

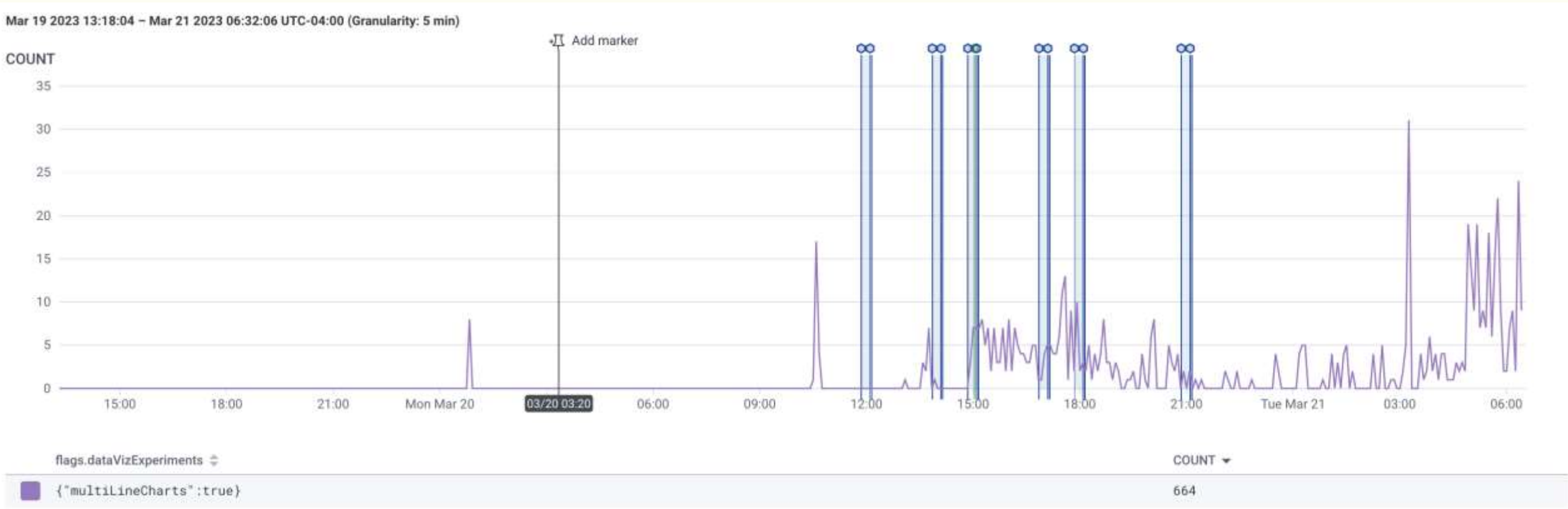
Trace duration distribution

- name
- db.name
- filename
- host.name
- http.server_name
- instrumentation_scope.name
- k8s.cluster.name
- k8s.deployment.name
- k8s.namespace.name
- k8s.node.name
- Create derived column

Run Query

Showing 1-4 of 5

Release with feature flags



A line of code

The Business



Product Team

A group of ten diverse people are captured in a candid, joyful moment, dancing and celebrating in what appears to be an office or meeting room. They are dressed in casual business attire, and their expressions and gestures convey a sense of fun and camaraderie. The background is a plain, light-colored wall, and the floor is covered with a blue patterned carpet. The overall atmosphere is one of positivity and teamwork.

Make friends with
customer support



Make friends with your salesperson

**“Embed the
customer”**

A line of code

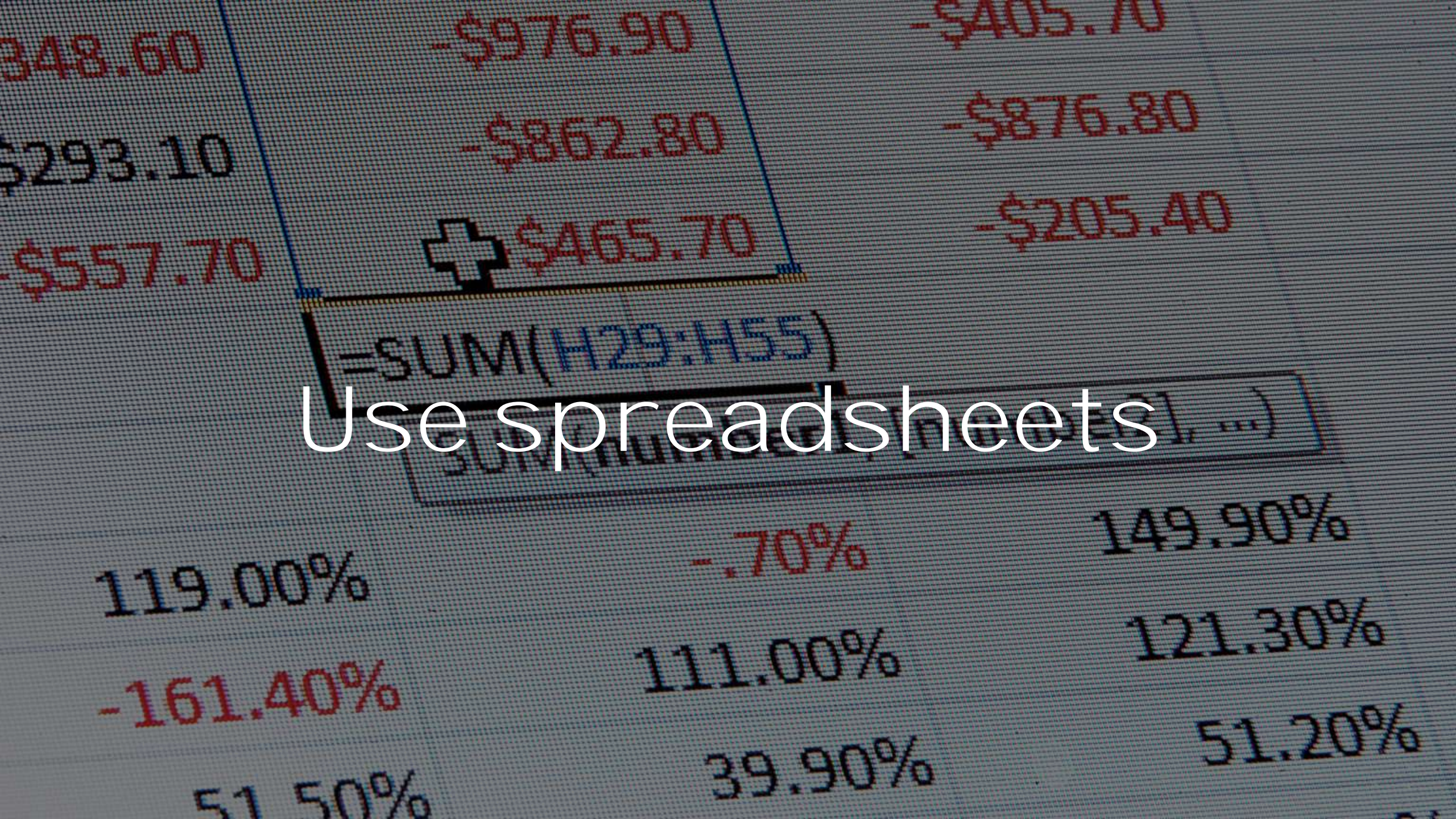
The Business



Facilitate REPL Nature

(because it's not all about you)





Use spreadsheets



WORDPRESS

Commodity
Software for
Commodity
Needs

Stop making
engineering the
bottleneck for
business
experiments!

Engagement overview

Average engagement time ?

0m 35s

Engaged sessions per user ?

0.58

Average €

0m 2

USERS IN LAST 30 MINUTES

1

USERS PER MINUTE

TOP PAGES & SCREENS

Frog and toad,... - avdi.codes

Support business metrics and analytics

01 Apr

02

03

04

05

06

07

Views

1.1K

Event count

4.3K

Event count by Event name

Seek Exploratory Feedback

READ

- Cultivate safe ways to play

Safe

EVAL

- Automate the repetitive, leave the rest open-ended

Automanual

PRINT

- Hope for surprise
- Manufacture serendipity

Exploramental

LOOP

- Keep trying
- Seek new questions, not mere answers.

Conversational



Chase new questions.

You'll get better answers.

TYPE STOP AND PRESS RETURN.
IF THAT DOESN'T WORK, HOLD THE
CTRL KEY DOWN AND PRESS THE C
KEY. THEN LET GO AND PRESS
RETURN. IF THAT DOESN'T WORK,
PRESS ESC OR ALT MODE. IF
THAT DOESN'T WORK, YELL FOR
HELP!

Good luck!

GRACEFUL  DEV

graceful.dev/phillyete2023/

References

- Albrecht, Bob 1972: [*My Computer Likes Me When I Speak Basic*](#)
- Carrol, John M. 1992: [*Making Errors, Making Sense, Making Use*](#). In *Software Development and Reality Construction*
- Deutch, L. Peter and Berkely, Edmund 1964: [*The Lisp Implementation for the PDP-1 Computer*](#)
- [DTSS Emulator](#)
- [Flutter](#), by Google
- Hendrickson, Elizabeth 2013: [*Explore It!: Reduce Risk and Increase Confidence with Exploratory Testing*](#)
- Martin, Kit; Horn, Michael; Wilensky, Uri 2020: [*Constructivist Dialogue Mapping Analysis of Ant Adaptation*](#)
- Papert, Seymour and Harel, Idit 1991: [*Situating Constructionism*](#)
- Perez, Carlos 2018: [*Touch \(Not Vision\) is the Foundation of Human Cognition*](#)
- [Project Jupyter](#)
- Rankin, Joy Lisi 2018: *A People's History of Computing in the United States*
- Royce, Winston 1970: [*Managing the Development of Large Software Systems*](#)
- Thomas, Dave 2015: [*Agile is Dead, Long Live Agility*](#)

Credits

- Photo of a PDP-1: Alexey Komarov, CC BY-SA 4.0 via Wikimedia Commons
- Event storming example process: Henning Schwentner, CC BY-SA 4.0, via Wikimedia Commons