You Have Agency

サンキュー、Ruby

@snoozer05 @kakutani

2026.10.29(Thu) - Rubyセミナー大阪

平凡な男が辿った数奇な人生

原作 スティーヴン・キング × 監督・脚本 マイク・フラナガンが
仕掛ける感動のミステリー

トム・ヒドルストン　キウェテル・イジョフォー　カレン・ギラン　ジェイコブ・トレンブレイ WITH マーク・ハミル AND

第49回
トロント国際映画祭
<観客賞> 受賞

サンキュー、チャック

5.1 fri

世界の終わりに明かされる、
愛すべき贈り物（サプライズ）とは？




平凡な男が辿った数奇な人生
@snoozer05 × @kakutani
仕掛ける感動のミステリー
トム・ヒドルストン　キウェテル・イジョフォー　カレン・ギラン　ジェイコブ・トレンブレイ WITH マーク・ハミル AND
You Have Agency
トロント国際映画祭
<観客賞> 受賞
サンキュー、
Ruby
2026.10.29(Thu) - Rubyセミナー大阪
世界の終わりに明かされる、
愛すべき贈り物（サプライズ）とは？

INTRODUCTION

2026年、AIが本格的にソフトウェア開発に参加し始めた年、島田浩二は一冊の本を訳し終えた。本の著者は、Dave Thomas。1999年、『達人プログラマー』を書き、2001年、アジャイルソフトウェア開発宣言に署名した人物。彼はその後、技術書の出版社Pragmatic Bookshelfを作り、『プログラミングRuby』や『Railsによるアジャイルwebアプリケーション開発』を始め、世に多くの技術書を届けた。島田が訳した書籍は、彼が2025年に11年ぶりに書き上げた『シンプリシティ』。
Daveが本書を執筆したのは、AIによる変化が本格化する少し前のことだった――

Dave Thomasの11年ぶりの新作『シンプリシティ』



- 現場開発者に向けた考え方が書かれている

Dave Thomasの11年ぶりの新作『シンプリシティ』



- 現場開発者に向けた考え方が書かれている
- 原書の刊行は2025年

Dave Thomasの11年ぶりの新作『シンプリシティ』



- 現場開発者に向けた考え方が書かれている
- 原書の刊行は2025年
  - AIによる変化が本格化する直前
  - AIへの直接的な言及などはない

Dave Thomasの11年ぶりの新作『シンプリシティ』



- 現場開発者に向けた考え方が書かれている
- 原書の刊行は2025年
  - AIによる変化が本格化する直前
  - AIへの直接的な言及などはない

→ **AI時代において読む価値は失われてしまったか？**

Dave Thomasの11年ぶりの新作『シンプリシティ』



- 現場開発者に向けた考え方が書かれている
- 原書の刊行は2025年
  - AIによる変化が本格化する直前
  - AIへの直接的な言及などはない

→ **AI時代において読む価値は失われてしまったか？**



訳者（本書を日本で一番読み込んだ人間）としては**本書の核となるメッセージは、AI時代だからこそ受け取るべきもの**だと考えています

今日のお話

- **『シンプリシティ』**に込められたメッセージを紐解きつつ
- それが、AI時代に重要な理由
- Rubyistである僕らは、それをどう受け取れるか

Act Three

THANKS, PROGRAMMERS

エージェントがコードを書く時代

- 開発者が書くコードの約半分（平均47%）は、エージェントが全部書いている

JetBrains Developer Ecosystem Survey 2026

- 「AIがあるから簡単に作れる。人はそんなに必要ない」

Gregor Ojstersek「Good Culture is the Biggest Productivity Hack, Not AI」


Pencils down
No more hand-written code

"LEGACY" PROGRAMMERS

80 GREAT YEARS!

Thanks Programmers!

The AI Impact Report 2026 - LeadDev

- 対象：エンジニアリングリーダー層が中心
- 質問：AI開発ツールの導入は、チームのエンジニアリング能力にどう影響したか
- 半数近く（42%）がチームの「AIなしでデバッグする力」が低下していると感じていると回答

AIを使った開発で、なぜ開発者のデバッグする力が低下するのか

デバッグする力とは

システムを理解し、仮説を立て、確かめ、選ぶ力

＝ **理解と判断の組み合わせ**

デバッグする力の下にある「理解と判断」が
チームの中から減っていると感じられている可能性

なぜ理解と判断が減るのか

- 開発者の理解や判断は、書く・直す・デバッグするという“作業”の副産物として従来得られていた
  - 例）触ったところ以外は壊れないはずと思ってコードを変更したところ、別の箇所でテストが落ちる → 暗黙的な結合箇所に対する理解が養われる
- つまり、作業（生成）だけを渡したつもりが、理解や判断を得る機会も手放してしまう

なぜ理解と判断が減るのか

- 開発者の理解や判断は、書く・直す・デバッグするという“作業”の副産物として従来得られていた
  - 例）触ったところ以外は壊れないはずと思ってコードを変更したところ、別の箇所でテストが落ちる → 暗黙的な結合箇所に対する理解が養われる
- つまり、作業（生成）だけを渡したつもりが、理解や判断を得る機会も手放してしまう

かつ、AIには判断や責任を委ねやすくする「認知的降伏」というフォースも存在する

認知的降伏（cognitive surrender）

- 人による判断を残していても、AIの出力をそのまま受け入れ、自覚のないまま判断を預けてしまう傾向
- 流暢で、自信ありげで、摩擦なく出力が届けられるときに起きやすい
- AIを使わなかった人よりも、AIを使った人の方が自信が上がる

Steven D. Shaw & Gideon Nave
”Thinking—Fast, Slow, and Artificial: How AI is Reshaping Human Reasoning and the Rise of Cognitive Surrender” (2026)



判断の形は残っているのに中身がない状態。認知的降伏は、そのときに人の内で起きていることに付いた名前。外から見た姿に付いた名前が「ミート・プロキシ（meat proxy、生成AIの出力を、検証・理解せずにそのまま他者に中継する人）」



Act Two

AGENCY FOREVER

Agile is Dead (Long Live Agility)

AGILITY PROCESS

In which I stick my neck out and initiate my Agile-is-Dead phase

On this page
Move to the Right
Back to the Basics

PUBLISHED
March 4, 2014

Thirteen years ago, I was among seventeen middle-aged white guys who gathered at Snowbird, Utah. We were there because we shared common beliefs about developing software, and we wondered if there was a way to describe what we believed.

It took less than a day to come up with a short list of values. We published those values, along with a list of practices, as the Manifesto for Agile Software Development:

|—:|:–|:—| | **Individuals and Interactions** | over | Processes and Tools | | **Working Software** | over | Comprehensive Documentation | | **Customer Collaboration** | over | Contract Negotiation, and | | **Responding to Change** | over | Following a Plan |

I was proud of what we did, both the process we followed and the result it produced. And I think that the existence of the manifesto has helped developers break free some some of the wasteful and soul-destroying practices of the '80s and '90s.

https://pragdave.me/thoughts/active/2014-03-04-time-to-kill-agile.html

You Have Agency

1 It's Your Life

I'm not in this world to live up to your expectations and you're not in this world to live up to mine.

Bruce Lee

It is *your* life. You own it. You run it. You create it.

Many developers we talk to are frustrated. Their concerns are varied. Some feel they're stagnating in their job, others that technology has passed them by. Folks feel they are under appreciated, or underpaid, or that their teams are toxic. Maybe they want to move to Asia, or Europe, or work from home.

And the answer we give is always the same.

"Why can't you change it?"

Software development must appear close to the top of any list of careers where you have control. Our skills are in demand, our knowledge crosses geographic boundaries, we can work remotely. We're paid well. We really can do just about anything we want.

But, for some reason, developers seem to resist change. They hunker down, and hope things will get better. They look on, passively, as their skills become dated and complain that their companies don't train them. They look at ads for exotic locations on the bus, then step off into the chilling rain and trudge into work.

So here's the most important tip in the book.

Tip 3 You Have Agency

Does your work environment suck? Is your job boring? Try to fix it. But don't

Agency

行為主体性

主体的に行動や選択を行い
環境に働きかけていける能力

『シンプリシティ』

小さく試し、学び、調整するアプローチ（**アジリティ**）を通じて、自分の仕事を理解しやすく、変更しやすく、余計な複雑さに振り回されない状態（**シンプリシティ**）に変えることを通して、開発における**agency**を持ち続ける方法の手本をDaveが示してくれている書籍

なぜ『シンプリシティ』のメッセージがAI時代に響くのか

Daveの課題意識

導入される制度やプラクティスとして「アジャイル」が使われる

→ 外側からやり方が与えられ、そこに作用できないことで、仕事の進め方が、働きかけるものから、従うものになってしまう

AI時代の課題

生成する役割としてAIが使われるようになっていく

→ 内側で手を動かす機会が失われることで、仕事する対象が、働きかけるものから、受け取るものになってしまう

仕事の進め方を支援・改善するために導入されたものが人と仕事の間に入り込み、

人が自分たちの仕事を動かしていく力を弱めてしまう



映画『サンキュー、チャック』https://www.youtube.com/watch?v=ABYLtRT62v4

Dance of Agency

Andrew Pickering『The Mangle of Practice』

- 人が世界に働きかけ、世界が返してくるものに応じて、人がまた働きかける。人の**行為主体性（agency）**は、その往復の中で形を得ていくという考え方
- ダンス…抵抗と調整の往復

抵抗（Resistance）

人は対象に働きかけるが、対象は必ずしも期待通りには応じない

調整（Accommodation）

人はその結果を受けて、方法を変え、ときには目標や意図も変えていく

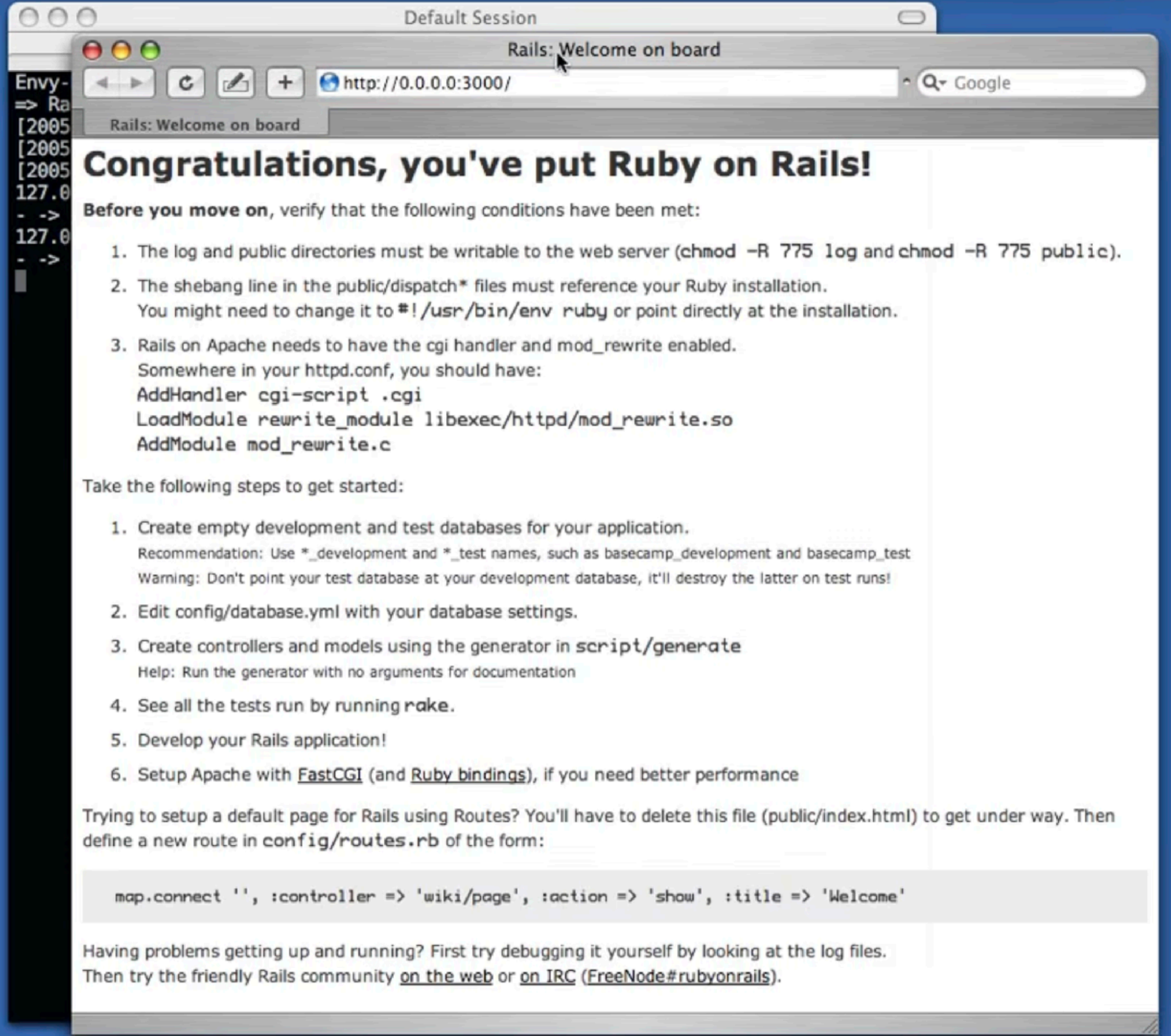
人の行為主体性とは、与えられるものでも、元から持っている素質でもなく、
世界との相互作用の中で発揮され、育まれていく



映画『サンキュー、チャック』https://www.youtube.com/watch?v=ABYLtRT62v4

何らかの装置を介して、世界と往復し
その往復の中でagencyを得ていく


RAILS


Default Session
Envy-
=> Ra
[2005
[2005
[2005
127.0
- ->
127.0
- ->
Rails: Welcome on board
http://0.0.0.0:3000/
Google
Rails: Welcome on board
Congratulations, you've put Ruby on Rails!
Before you move on, verify that the following conditions have been met:
1. The log and public directories must be writable to the web server (chmod -R 775 log and chmod -R 775 public).
2. The shebang line in the public/dispatch* files must reference your Ruby installation.
You might need to change it to #!/usr/bin/env ruby or point directly at the installation.
3. Rails on Apache needs to have the cgi handler and mod_rewrite enabled.
Somewhere in your httpd.conf, you should have:
AddHandler cgi-script .cgi
LoadModule rewrite_module libexec/httpd/mod_rewrite.so
AddModule mod_rewrite.c
Take the following steps to get started:
1. Create empty development and test databases for your application.
Recommendation: Use *_development and *_test names, such as basecamp_development and basecamp_test
Warning: Don't point your test database at your development database, it'll destroy the latter on test runs!
2. Edit config/database.yml with your database settings.
3. Create controllers and models using the generator in script/generate
Help: Run the generator with no arguments for documentation
4. See all the tests run by running rake.
5. Develop your Rails application!
6. Setup Apache with FastCGI (and Ruby bindings), if you need better performance
Trying to setup a default page for Rails using Routes? You'll have to delete this file (public/index.html) to get under way. Then define a new route in config/routes.rb of the form:
map.connect '', :controller => 'wiki/page', :action => 'show', :title => 'Welcome'
Having problems getting up and running? First try debugging it yourself by looking at the log files.
Then try the friendly Rails community on the web or on IRC (FreeNode#rubyonrails).

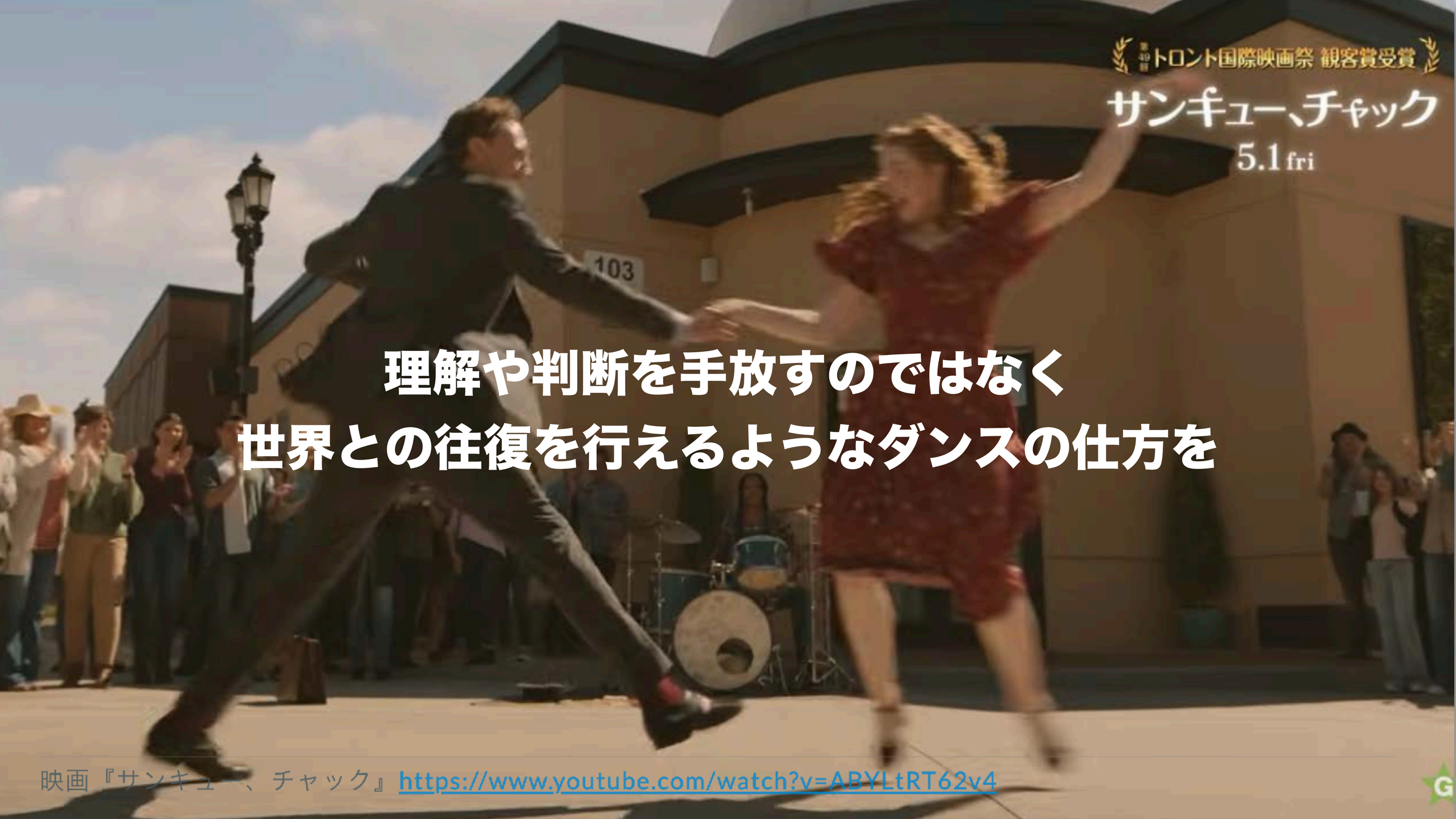
RailsによるアジャイルWebアプリケーション開発の往復から受け取ったもの

- ~~宣伝文句であった生産性~~
- 「Webについての知識がまだ十分でない自分でも、Webアプリケーションを作っていける」という感覚
  - Webアプリケーション開発という活動に対するagency

Railsとのダンス

- Webについての知識・理解がまだ十分でなくても、**最初はリードしてもらう**所からダンスを始められ……
  - rails g や「設定より規約」、ARなどを通じて、できないことをやってくれる相手
- Railsが用意してくれたコードをいじり、自分でも書いてみて、rails c でアプリケーションの状態を確認するという往復から、知識や理解を深めていき、気がついたら、**こちらがリードする側にも**回れる関係に……
- 「できないこと」をやってもらっていた相手から、「できること」を任せる相手へ

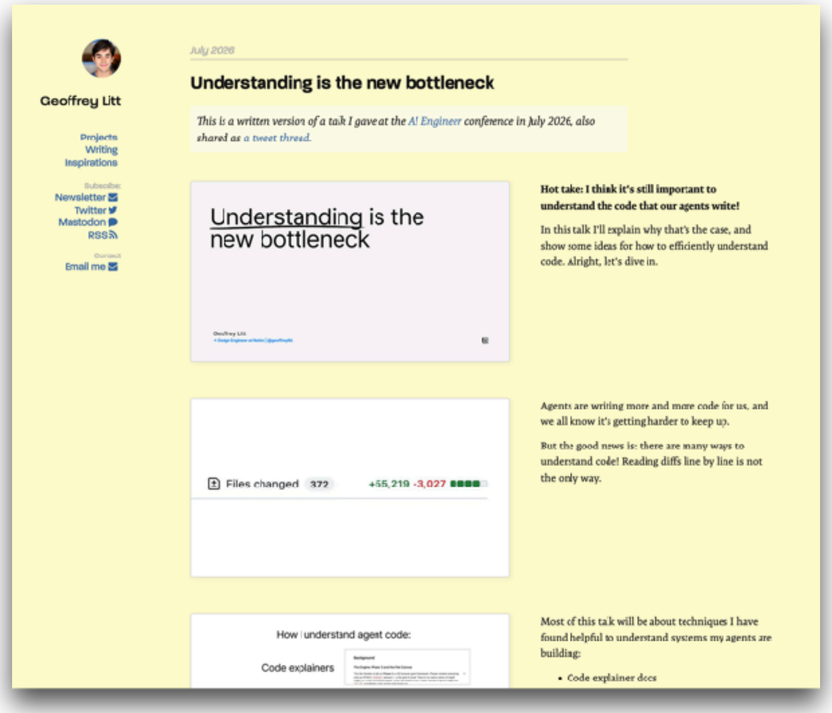
AIともRailsと同じような関係を作ればいい

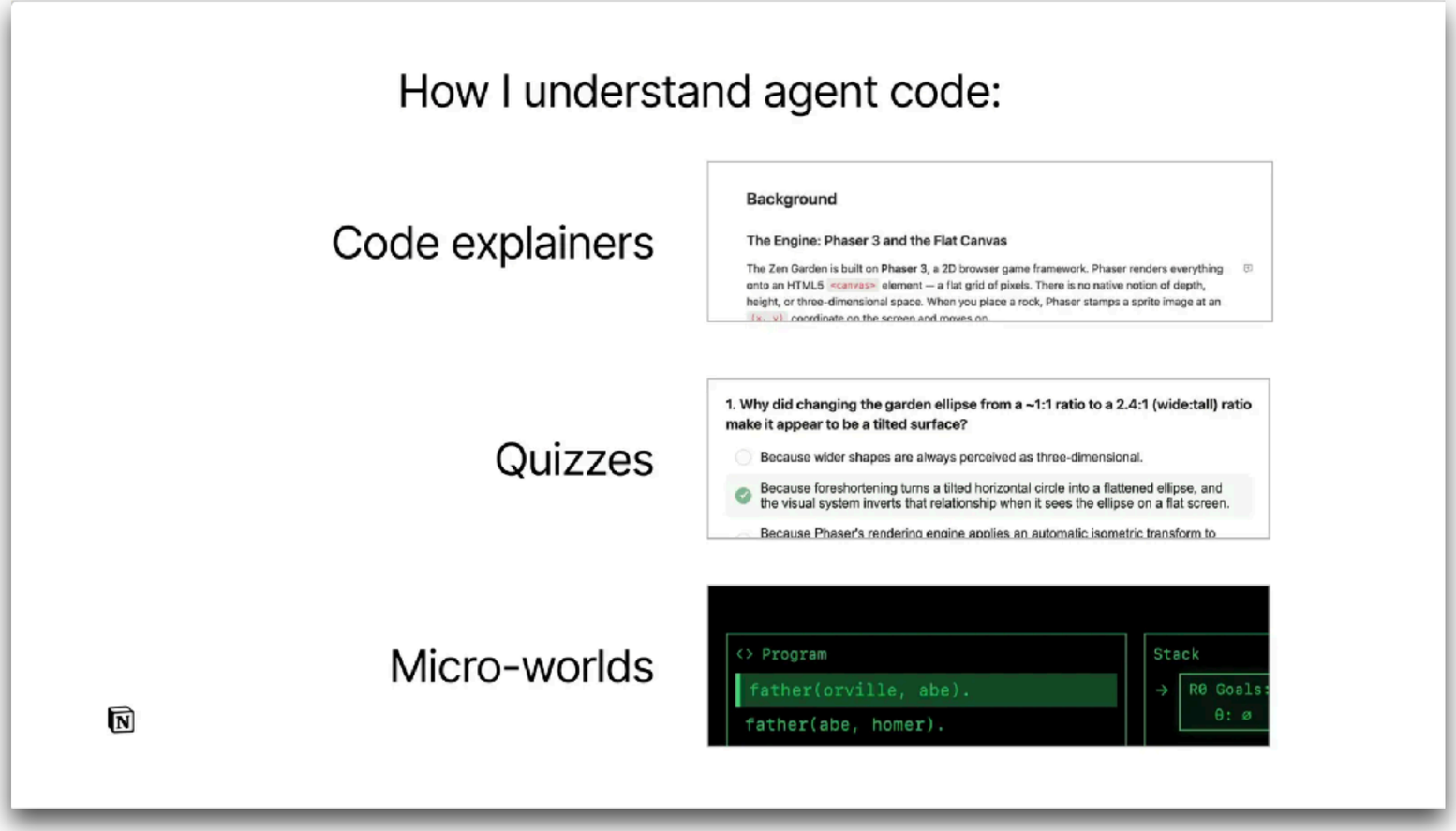


理解や判断を手放すのではなく
世界との往復を行えるようなダンスの仕方を

映画『サンキュー、チャック』https://www.youtube.com/watch?v=ABYLtRT62v4

手がかり





https://www.geoffreylitt.com/2026/07/02/understanding-is-the-new-bottleneck.html

理解のために、AIとやり取りする



Act One

I CONTAIN MULTITUDES

なぜRailsは人の側が意識しなくても
ダンスできる装置たりえたのか

Ruby

The Philosophy of Ruby
A Conversation with Yukihiro Matsumoto, Part I

by Bill Venners
September 29, 2003

Summary

Yukihiro Matsumoto, the creator of the Ruby programming language, talks with Bill Venners about Ruby's design philosophy, including design imperfection, the danger of orthogonality, and the importance of the human in computer endeavors.

Yukihiro Matsumoto, or "Matz," as he is known online, is the creator of the Ruby programming language. Ruby is an object-oriented language suitable for writing day to day scripts as well as full-scale applications. Matz began work on Ruby back in 1993, because he wanted a language that made him productive while being fun to use. Initially popular in Japan, Ruby has been finding its way into the hearts of programmers all over the world.

On September 24, 2003, Bill Venners met with Yukihiro Matsumoto at the JAOO conference in Aarhus, Denmark. In this interview, which will be published in multiple installments on Artima.com, Yukihiro Matsumoto discusses Ruby's design philosopy, the features of the Ruby language, and becoming a better programmer. In this initial installment, Matz waxes philosophic about design imperfection, the danger of orthogonality, granting freedom with guidance, the principle of least surprise, and the importance of the human in computer endeavors.

No Perfect Language

Bill Venners: Dave Thomas, coauthor of *Programming Ruby: A Pragmatic Programmer's Guide*, told me that you don't think a language design should be perfect. Why not?

Yukihiro Matsumoto: Language designers want to design the perfect language. They want to be able to say, "My language is perfect. It can do everything." But it's just plain impossible to design a perfect language, because there are two ways to look at a language. One way is by looking at what can be done with that language. The other is by looking at how we feel using that language—how we feel while programming.

Because of the Turing completeness theory, everything one Turing-complete language can do can theoretically be done by another Turing-complete language, but at a different cost. You can do everything in assembler, but no one wants to program in assembler anymore. From the viewpoint of what you can do, therefore, languages do differ—but the differences are limited. For example, Python and Ruby provide almost the same power to the programmer.

Instead of emphasizing the what, I want to emphasize the how part: how we feel while programming. That's Ruby's main difference from other language designs. I emphasize the feeling, in particular, how *I* feel using Ruby. I didn't work hard to make Ruby perfect for everyone, because you feel differently from me. No language can be perfect for everyone. I tried to make Ruby perfect for me, but maybe it's not perfect for you. The perfect language for Guido van Rossum is probably Python.

Orthogonal versus Harmonious

Rubyの哲学

- Rubyを使っているときにプログラマーが**どう感じるかを大事にする**
- プログラマーに「**選択の自由**」を与える
- **本質的なことに集中できる**ように簡潔に書けるようにする


この哲学がダンスへの誘い

Rubyという装置

- dance of agencyが起こりやすい哲学の元に作られている

Rubyは言語である

each_consして渡ってきたリストの各要素を
joinした文字列へとmapしてやる

Rubyは言語である

> “プログラミング言語はプログラマがプログラムを考えるためのものであって、既に考えたプログラムを書き下すためのものじゃない。

ポールグレアム『ハッカーと画家 コンピュータ時代の創造者たち』

A Ruby Programming Episode

Ruby is my glasses, my stationery, and my language.

島田 浩二(@snoozer05)



福岡Ruby会議02; Nov 25, 2017

https://speakerdeck.com/snoozer05/a-ruby-programming-episode-ruby-is-my-glasses-my-stationery-and-my-language

私にとって
Rubyがどんな言語であるか

ぼくにとってRubyとは

- **めがね**である: 問題の見方を与えるもの
- **ペン**である: 思考をスケッチするためのもの
- **言語**である: プログラマとして思考する言語


Ruby
Me

Rubyという装置

- **dance of agencyが起こりやすい哲学**の元に作られている
- だから、Rubyを書いていると、**自分の仕事を自分で動かしている感覚**が育つ

Rubyという装置

- **dance of agencyが起こりやすい哲学**の元に作られている
- だから、Rubyを書いていると、**自分の仕事を自分で動かしている感覚**が育つ
- この感覚こそが、Rubyistの言う「**たのしい**」の正体の一つでは

サンキュー、Ruby

…… そして長くRubyと踊っているうちに、Rubyから受け取ったものは、Rubyを使っているときだけのものではなくなり、ソフトウェア開発をしているときに常にある感覚となり、そして自分は「プログラマー」の一員になれたと感じています

できることを増やしていく
新しいダンスをAIとこれから

- AIの方には人に選択や判断を放棄させるほうに向かうフォースがある
- けれど、僕らは、何も持たずにAI時代に放り出されたわけではない
- 道具に助けてもらいながら、自分で考えることも手放さない踊り方はすでに知っている
- Rubyと同じようにはいかないもしれないけれど……



RubyやRailsと踊った記憶も、その根源にあるMatzの哲学も、そして、Daveをはじめとする先輩たちが示してくれている踊り方のヒント、そして先輩たちがコミュニティの中で踊ってきた“さま”も全て、全部私たちの中にあるから、私たちは大丈夫

We are wonderful

We deserve to be wonderful

And we contain multitudes

でもやるん
だよ!!!q

You have agency.
Thank you for the dance.

You Have Agency

サンキュー、Ruby

@snoozer05 @kakutani

2026.10.29(Thu) - Rubyセミナー大阪

参考文献

映画『サンキュー、チャック』（The Life of Chuck）マイク・フラナガン監督, 原作 スティーヴン・キング, 2024

Dave Thomas『Simplicity: Sustainable, Humane, and Effective Software Development』Pragmatic Bookshelf, 2025（邦訳『シンプリシティー 持続可能かつ人間的で効果的なソフトウェア開発』）

David Thomas, Andrew Hunt 『The Pragmatic Programmer 20th Anniversary Edition』（邦訳『達人プログラマー 第2版』）

Dave Thomas「Agile is Dead (Long Live Agility)」2014

GOTO Book Club「Simplicity」2025-06

JetBrains「How Much Code Do Developers Really Let Agents Write?」JetBrains Research Blog, 2026-08-26（Developer Ecosystem Survey 2026）

Gregor Ojstersek「Good Culture is the Biggest Productivity Hack, Not AI」Engineering Leadership, 2026-08-24

DHH「Rails World 2026 Opening Keynote」2026

LeadDev「The AI Impact Report 2026」

Steven D. Shaw & Gideon Nave「Thinking—Fast, Slow, and Artificial: How AI is Reshaping Human Reasoning and the Rise of Cognitive Surrender」PsyArXiv preprint, 2026

Niklas Gruhn「Don't be a meat proxy」2026-08-03

Geoffrey Litt「Understanding is the new bottleneck」2026-07-02

Andrew Pickering『The Mangle of Practice: Time, Agency, and Science』University of Chicago Press, 1995

島田浩二「A Ruby Programming Episode: Ruby is my glasses, my stationery, and my language」福岡Ruby会議02, 2017-11-25

ポール・グレアム『ハッカーと画家 コンピュータ時代の創造者たち』川合史朗 訳, オーム社, 2005

Dave Thomas, David Heinemeier Hansson ほか『RailsによるアジャイルWebアプリケーション開発』オーム社, 2006

Bill Venners「The Philosophy of Ruby: A Conversation with Yukihiro Matsumoto, Part I」Artima, 2003-09-29

David Heinemeier Hansson「The Rails Doctrine」

角征典「サンキュー、XP」