

# JSer.infoの作り方

# 去年のまとめ

- JavaScript情報ってなんだっけ?
- 未来に関する悩みが増えた
- 情報の安定性を見る方法について
- コミュニティの問題が表面化
- 1つだけではなく複数の情報から判断しよう

A large, stylized green letter 'Q' with a white outline, positioned on the left side of a white rectangular area.A large, stylized red letter 'A' with a white outline, positioned on the right side of a white rectangular area.

JavaScript情報って  
なんだっけ?

[JSer.info](#) 5周年記念イベント

# はじめに

- このスライドには再現性がありません
  - JSer.info 特有の考え方が多く含まれます
- 技術的な内容ではありません
- 各自、適当に見てね

# アジェンダ

- JSer.infoの作り方
- 記事公開までのワークフロー
- 継続的にやるために考えること

# テーマ

- 何か作る上で次のようなワークフローを辿る
  - 見る(Watch)
  - 調べる(Search)
  - 学ぶ(Learn)
  - 考える(Think)
  - 作る(Create)
  - 共有(Share)

## 知識

調べる  
Search

見る  
See

## 行動

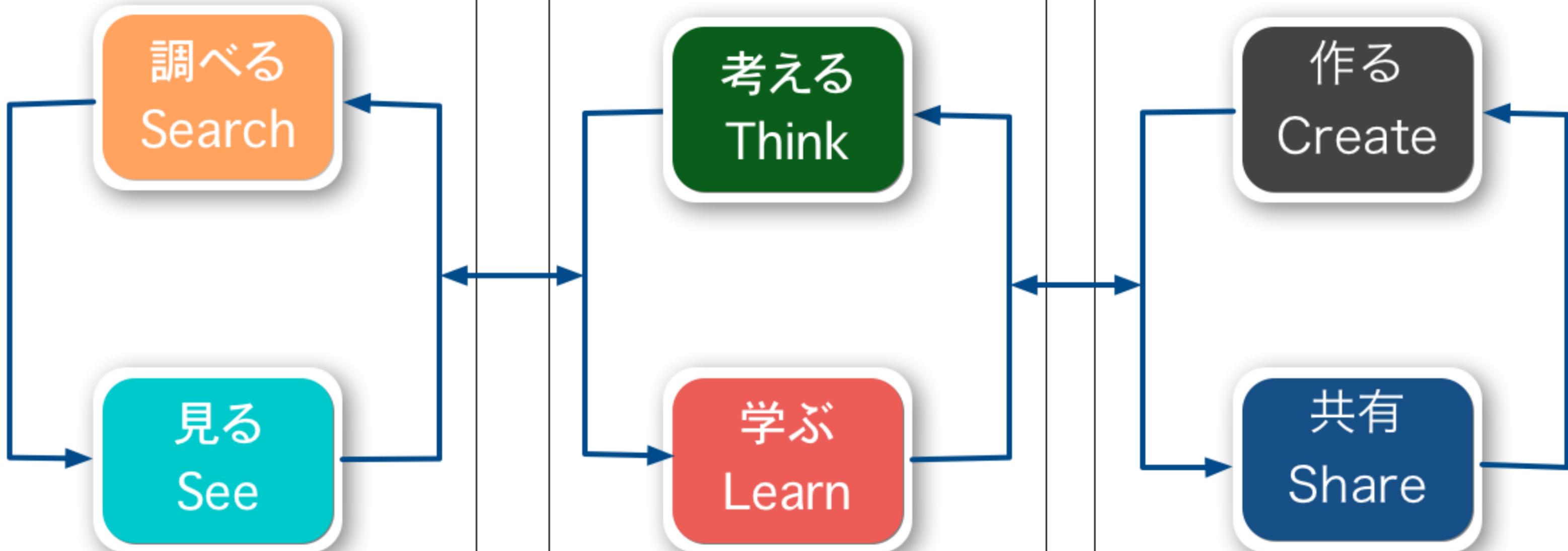
考える  
Think

学ぶ  
Learn

## 目的

作る  
Create

共有  
Share



# 目的

- **知識**を元に**行動**することで**目的**を達成する
- 大体のものごととは同じようなフローを辿る
- JSer.infoにおける、このフローの詳細を見ていく話

JSer.infoの場合

# JSer.infoの目的

なぜこのようなサイトを始めたのかというと、現在のインターネットはJavaScriptの情報が溢れていて自分の周りだけの情報で全てだと錯覚してしまうほど情報量だと思います。

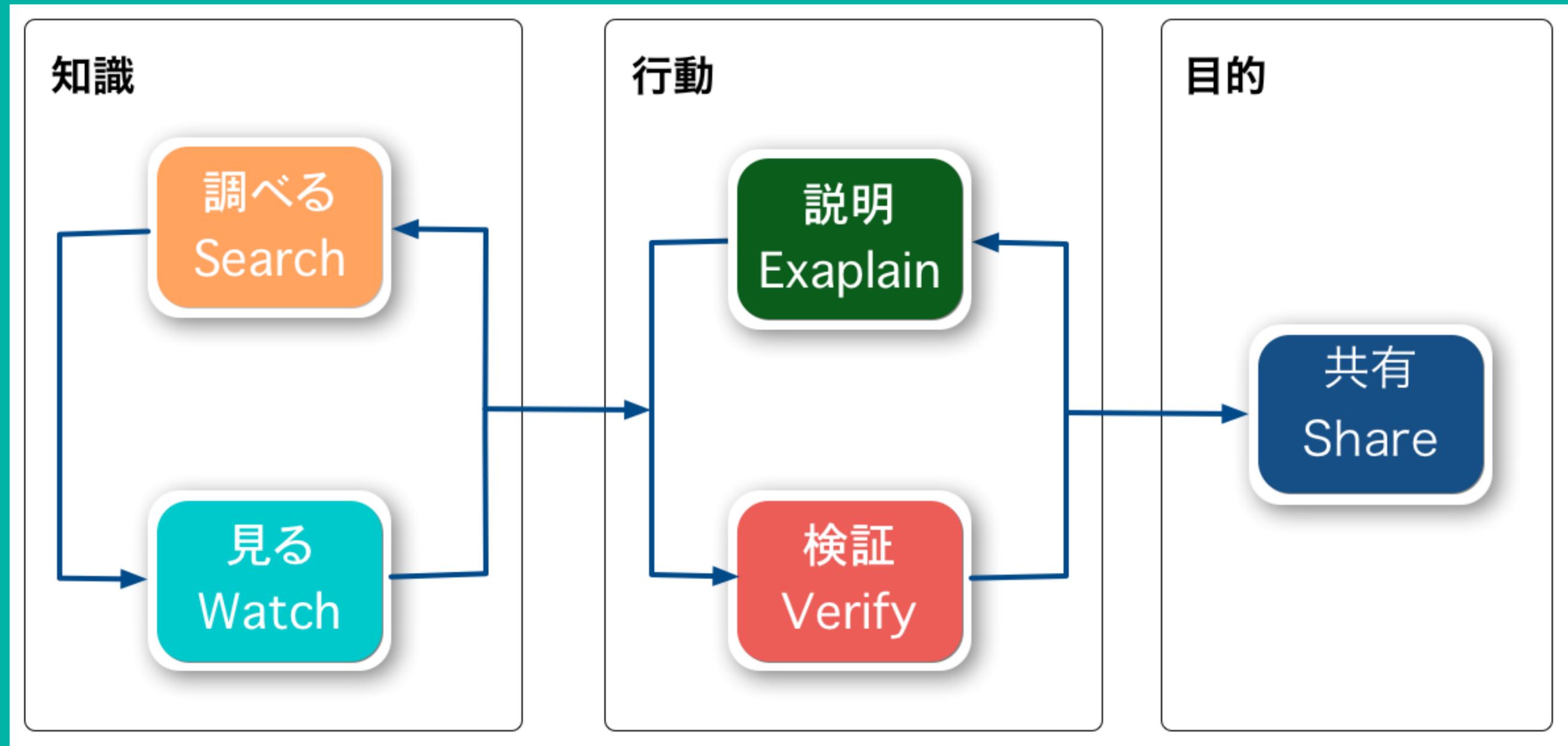
...ある程度の内容に絞った情報を提供する場所が必要ではないかと考えました。幸いにも私は情報を集めることが好きなので、そのような人間が少し整理した情報を提供することでより良い流れが作れるのではないかと考えています。改めて、このサイトは言語問わないJavaScriptの情報を紹介するサイトです。しかし、真の目的はJavaScriptの情報を”紹介”ではなく”知ってもらおう”事にあるため、継続的な活動が必要となるでしょう。

– <https://jser.info/about/>

# JSer.infoのゴール

- JSer.infoのゴール = 方向
- 「JavaScriptに興味がある人にもっとJSを知ってもらおう」
- 「JavaScriptの情報を整理して伝える」
- 「JavaScriptの現状を正確に伝える」
- 「更新コストを小さくして、継続できる形を作る」

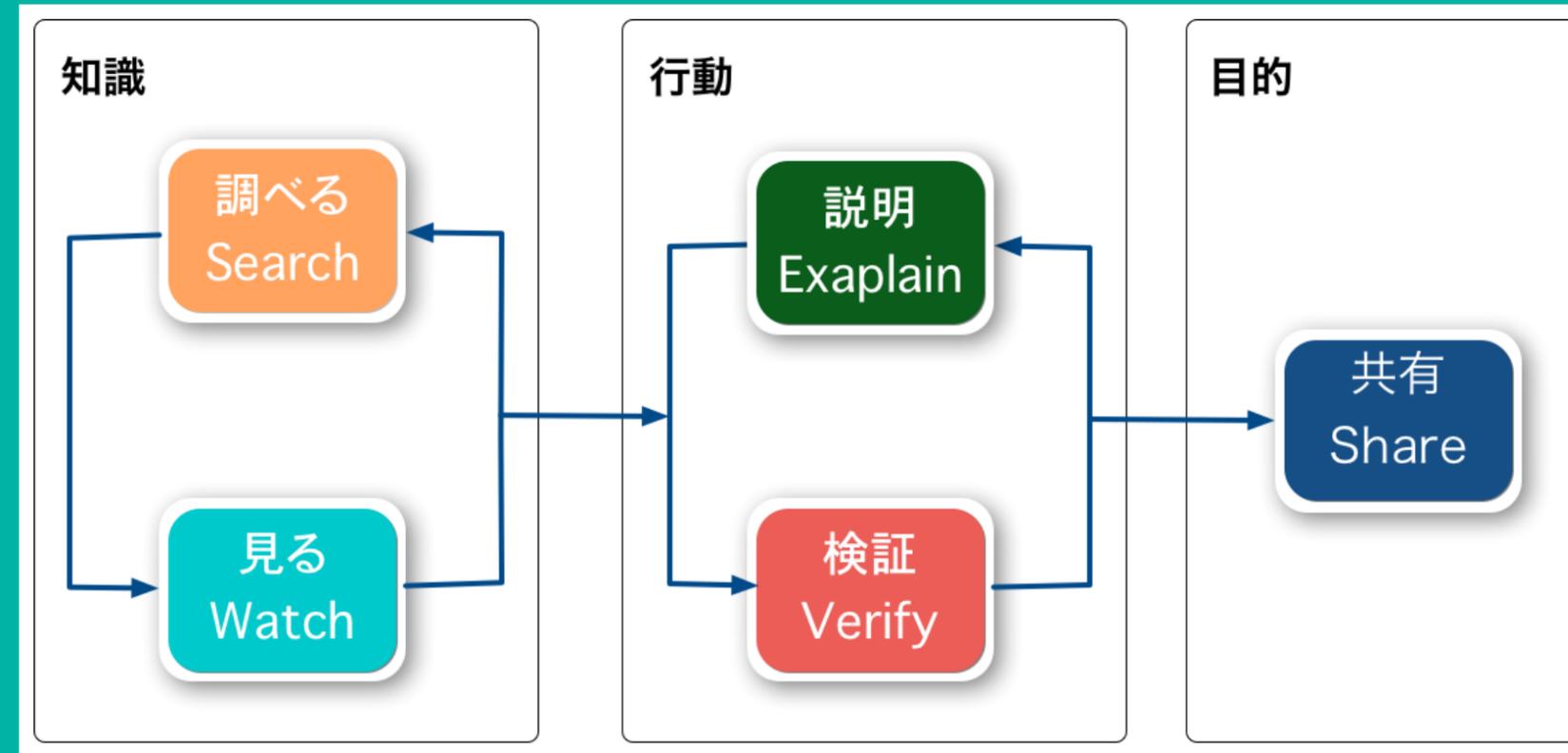
# JSer.infoのワークフロー



# JSer.infoのワークフロー

- 1つ1つの記事ごとに次のワークフローを通す

1. 見る
2. 調べる(省略可能)
3. 検証(省略可能)
4. 説明
5. 共有



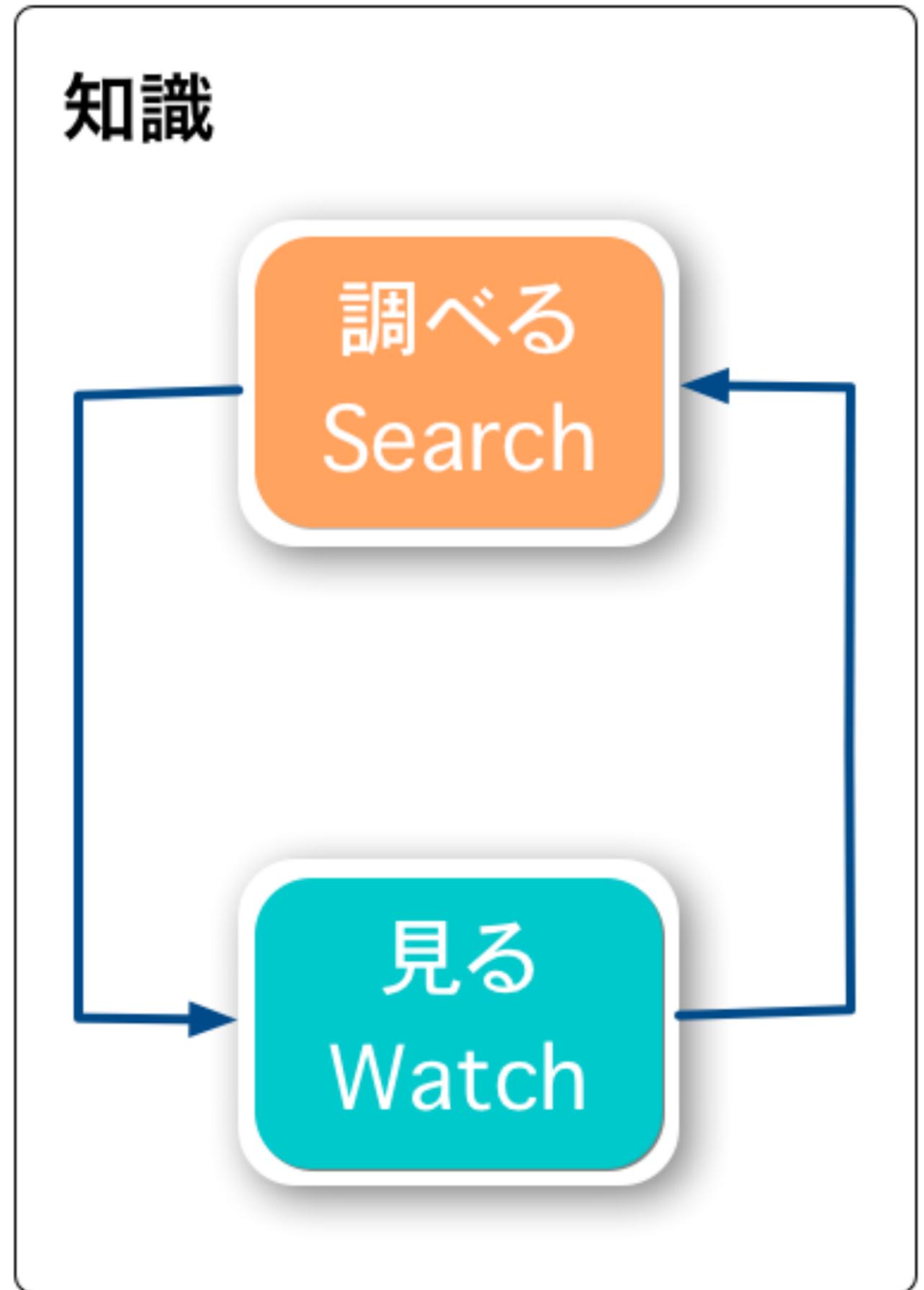
## 目的を満たすために

- JavaScriptの情報を整理して正確に伝えるために
- 誇張や誤った情報は避ける必要がある(Verify)
- また伝えるには言葉(文字)を使うので、嘘をつかないように説明する必要がある(Explain)
- 人にとって「良い」という言葉の解釈が異なるので難しい
- この問題に対してどのようなアプローチをとっているかについて

具体的にどう行動するか  
まずは情報がないと始まらない

# 見る(Watch)、調べる (Search)

- どのプログラミング言語でも大体同じやり方が通じる
- どちらも"情報収集"
- 調べるは能動的、見るは受動的な情報収集
- 情報化社会を この先生きのこるためには / Layzie@Frontrend in Kanazawa // Speaker Deck



# 見る(Watch)

- JSer.infoでは、とにかく情報を見る/気づくことが重要
- 気づくために色々なアンテナを貼る
- 「見る」で重要なのは、自分が見る所に集約すること
  - 見ない所に集めても結局見ない
  - 情報不足、情報過多は混乱を生む = 自分に合うバランスが必要
- @azu がもっと見るのはRSSリーダーとTwitter



# GitHubを見る

- しかし、GitHubのタイムライン(通知)は破綻してる
- そのため色々なツールやアプリなどを書いたり
- GitHubでライブラリのリリースを見ていくためのツールや方法 | Web Scratch
- Githubのタイムラインや通知を見るアプリをnode-webkitで作った | Web Scratch

🔔 Notifications	👁 Watching
<b>Unread</b>	<b>49271</b>
Participating	0
All notifications	
<hr/>	
Microsoft/TypeScript	4770
nodejs/node	3972
facebook/react	2085
webpack/webpack	2009
yarnpkg/yarn	1764
eslint/eslint	1749
vuejs/vue	1703
facebook/flow	1455
facebook/jest	1436
google/WebFundamentals	1291
Microsoft/ChakraCore	1242
babel/babel	1217
angular/protractor	932
whatwg/html	900
Fyrd/caniuse	716
lodash/lodash	564
moment/moment	544
ReactiveX/rxjs	542

# GitHub

- リポジトリをWatch 
  - アクティブに興味があるリポジトリをWatch
- リポジトリをStar 
  - 特に意味なくStarする
  - starWatchkerで補足
- リポジトリのリリースをRSSで購読
  - Feedlyに溜め込む => IFTTT -> Twitterへ投げる

# GitHubをTwitterで見る

- AWS lambdaでGitHubのアクティビティをTwitterで読む用に投稿する | Web Scratch
- BotでGitHubのNotification(Watch)、アクティビティを流す
- 結果、Twitterで次の情報が見れる
  - Watch/アクティビティ(フォロワーが何をStarしたとか)/リポジトリのリリース情報

The screenshot shows a Twitter feed with three tweets from GitHub. The top tweet is from 4 minutes ago, from the account [Microsoft/TypeScript], with the text "Latest NTypeScript build broken". It includes a profile picture of a man and a link to "Issue #9039". The second tweet is also from 4 minutes ago, from [Microsoft/TypeScript], with the text "Fix #6319: Add support for '--t: es5' and '--m es6'". It includes a profile picture of a man and a link to "Fixes #6319". The third tweet is from 6 minutes ago, from [facebook/react], with the text "Fix IE11 placeholder textContent value bug.". It includes a profile picture of a grid of blue squares and a link to "Fix IE11 placeholder textContent value bug. by jimf...". Each tweet has a row of interaction icons (reply, retweet, like, and more) below it.

# 見逃したくない情報は何重にも出す ⚠

- RSSリーダーでキャッチする
- リポジトリのリリース情報はRSSにもTwitterにも流す
- あとで読むはPocketに入れる
  - Pocket ExposeでPocketの中身がRSSで流れる
- それでもタブに溜め込みがち
  - [jser/ping](#)へ投げる
  - 該当のIssueはメールとRSSリーダーに通知される(流れにくい)

# RSSリーダー

- LDRを使ってる
- 現在の購読フィード数は 3212
- 購読するフィードは気にせず追加する
- 更新されなくなるフィードも多いため、結果的に帳尻は合う

# 「知る」まとめ

- 人によってやり方は様々
  - 自分が最も見る所に集約するのが簡単
- 情報は少なくても、多くても混乱する
  - 自分のニーズにあったやり方を作ることが大事
- 情報の意味についてはJavaScript情報ってなんだっけ?を参照
- JSer.infoの目的の一つは多すぎる情報を整理することにある

調べる(Search)

# 調べる(Search)

- 調べるは能動的な行動
- とりあえずググる
- とりあえずGitHub検索する
- とりあえずTwitter検索する
- とりあえず専門サイトを探す

# Google検索

- 英語、英単語で検索
- 日付を絞って検索
- URLを検索
- Fix Google Search Options
- 情報化社会を この先生きのこるためには / Layzie@Frontrend in Kanazawa // Speaker Deck



# GitHub検索

- GitHubの検索オプションは充実
- 検索方法も充実
  - リポジトリ/コミット/ソースコード/Issue/PR
- githubのissue, pull request 検索オプションをハックしてチートシート作った

## 関連を検索する

- URLで検索するというのは結構大事
  - 関連する/参照している情報にたどり着きやすい
- あるもの見つけて利用している「単語」を見つけてもう一度検索
  - 検索できる語彙をふやせる

# 例) テキストから感情を検出したい

1. テキストがポジティブ or ネガティブかを判定したい
2. 感情 判定 で検索
  - 単語感情極性対応表の辞書の存在を知る
3. URLでググる
  - 類似研究を見つける
4. 感情極性 というキーワードがあるらしい
5. ちょっと辞書が古そう(2005年 ~ 2008年)
6. Word2Vecとか関係あるのでは(既存の知識から)
  - 最近の研究でも同様の辞書が採用されてた。現役っぽいことがわかった

# 相対は関連を検索して知る

- 見つけたものが古いものなのか(もっと新しいものがあるか)を知りたい場合
- "<見つけたもの> compare related alternative migrate move to insteadof"とかで検索する
- GitHubで検索するのが簡単な方法(コミットメッセージも検索できる)
- 作る側も古いものを参照(リンク)していることは多い
- なぜなら
  - 古いものと一緒に作ったものがどう違うのかをREADMEに書いている
  - 比較を書くことで、作ったものにどういうメリットがあるかを明示したいから

# 専門の検索エンジンを知ると便利

- 単純な発見はGitHubとかGoogleで十分
- 比較やメタ情報を扱う場合は専門のサイトが強い
  - [Libraries.io](#)
  - [npm trends](#)
- どうやって専門のサイトを見つけるか?
  - GitHubで[awesome list](#)からを見つける
  - 昔ながらのディレクトリ検索 や 人に聞くのが有効な手段

知る -> 行動



# 人は知りすぎていると動けなくなる

実際に行うことよりも、知り続けることを優先すると、ある時点から混乱が増してきます。

## — 今日からはじめる情報設計

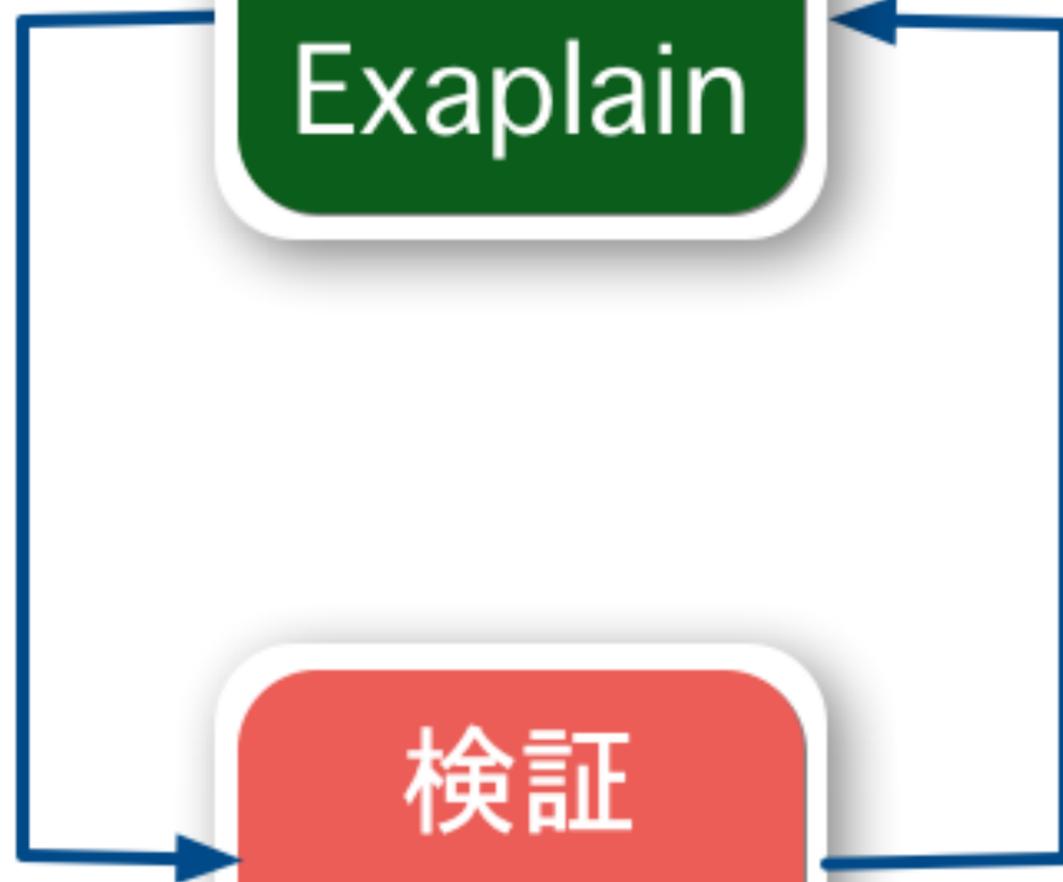
- 「知り続ける」だけだと、次の行動に手間を取るようになってしまう
  - 行動(変化)は今の知識を過去に追いやる行為であるため
- 類語: 選択肢が多すぎると、結局何も選べない<sup>p88</sup>
- Input/Outputのバランス大事

行動

行動

説明  
Exaplain

検証  
Verify



# 検証する(Verify)

- 「正しく」説明するには「検証」する必要がある
  - 紹介するものに誇張表現が含まれているかもしれない
- なぜ高速なのか、どういう仕組みなのか、どのような議論が行われたのかなど

# The Mechanism is not the Mental Model

– *[Dave Herman]*

# なぜ検証する必要があるのか

- 必ずしも直感は正しくない
  - 未知の分野、規模が大きい場合など、直感は間違っていることがある
- 作者による説明(主張)が正しいとは限らない
- 例) 「jQuery-compatible API」と書かれている
  - 検証するとjQueryと同じような機能を持っているだけだった
  - 正しくは「jQuery-like API」

# どうやって検証するのか

- ソースコードやIssueを読む
  - テストやサンプルコードを読む/動かす
- 実際に動かしてみると分かることもある
  - JavaScriptのトレンドを素振りして確認する方法 - Qiita
- パフォーマンス系は数値のマジックが多いので、必ずチェック
  - 検証すると多くはエッジケースにおける改善

# ソースコードをCloneして動かすSnippet

```
ghq get "https://github.com/jser/jser.info.git" --update
ghq look "https://github.com/jser/jser.info.git"
yarn --pure-lockfile # no-lock file
```

- [motemen/ghq](#)を使ってclone
- [Yarn](#)を使ってインストール<sup>shell</sup>
  - 大体使われるライブラリは偏るのでキャッシュが効くYarnは効果的

# 変更点を見つける

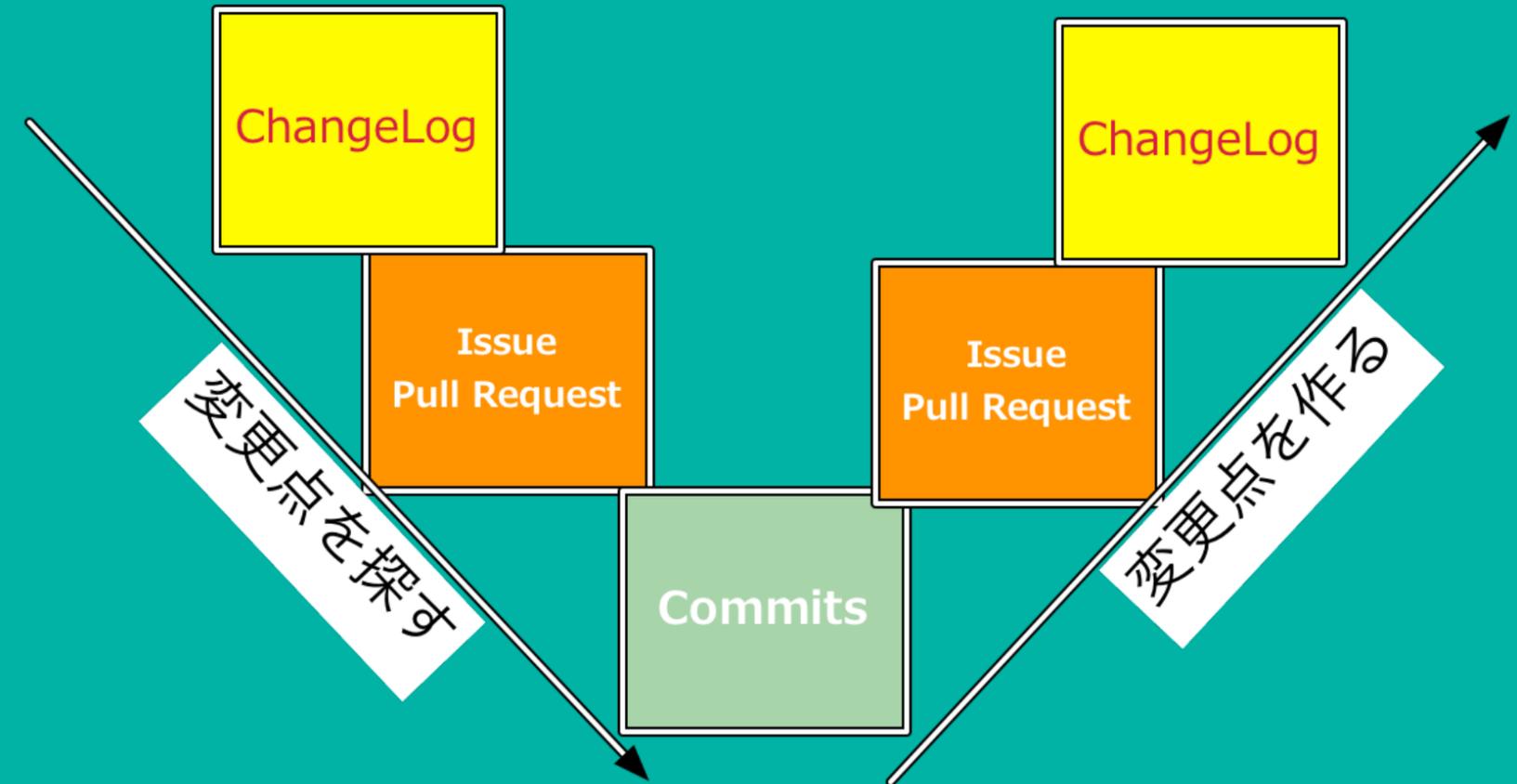
- 更新に対して一から見つけるのは時間の無駄
- 実際の変更点を見つけるのが時間短縮に繋がる
- われわれは、いかにして変更点を追うか
- CHANGELOG/ISSUE/PR/Commitsから実際のソースコードを探す

われわれは、いかにして  
変更点を追うか

ChangeLog/Issueを追う技術

# 変更点を見つける(要約)

1. CHANGELOG/リリースノートを見る
2. Issue/Pull Requestを見る
3. コミットを見る
4. ソースコードを見る



# すべては検証できない

- 時間的な制約、コスト的な問題がある
- 不確実性が含まれるならJSer.infoにおいてはスルーする
  - JSer.infoの目的に反してしまうため諦める
- 諦めて後回しにしておくことで、誰かが解決してくれるかもしれない
- 苦痛を感じるならやるべきではない

## 知識

調べる  
Search

見る  
Watch

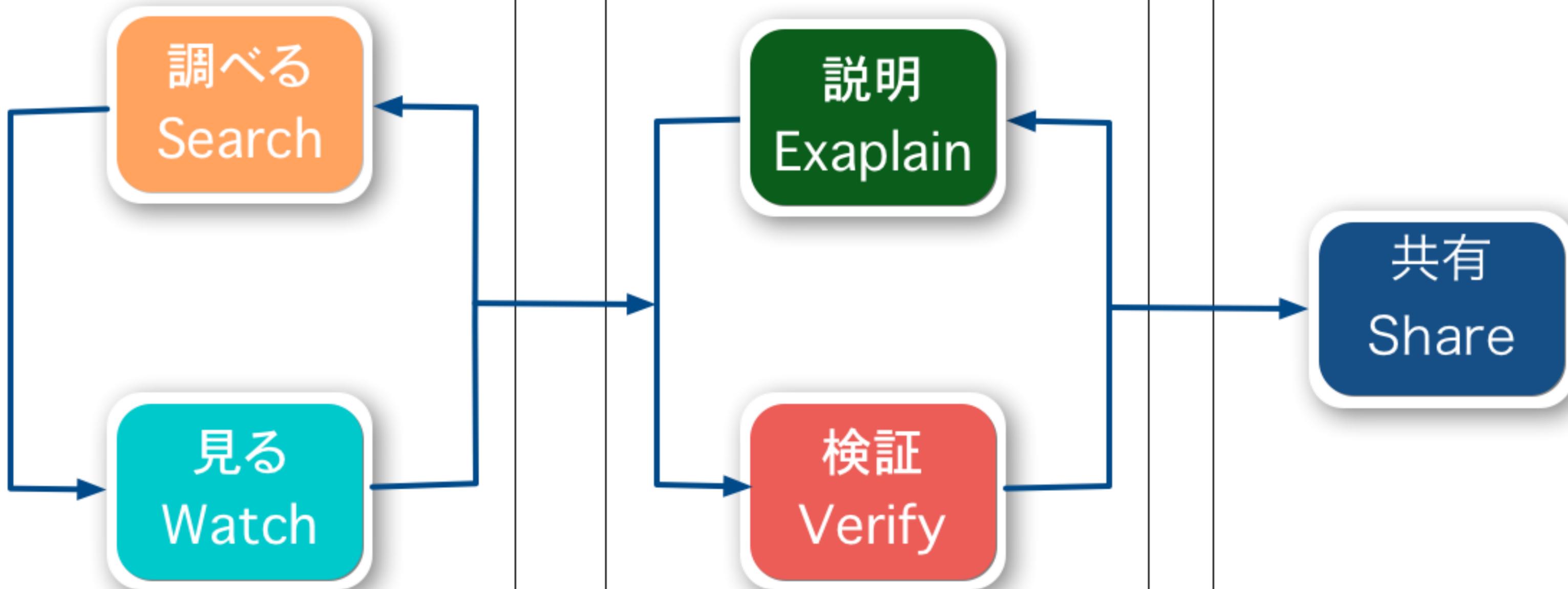
## 行動

説明  
Exaplain

検証  
Verify

## 目的

共有  
Share



# 説明する(Explain)

- 検証した結果を踏まえて「説明」する
  - タグ、説明、関連記事をpostemで書いて投稿
- jser/jsr.infoにpushされる
- 「説明」には「言葉」を使うけど...
  - 「言葉」は人によって認識が異なる
  - 言葉の使い方によっても受け取り方はことなる

## Title & URL

2017-01-04のJS: PostCSS概要、Chrome開発者ツール、

<https://jsr.info/2017/01/04/postcss-chrome-fusebox>

via URL

## Tags

× JavaScript × JSer

## Body

スペルチェック付きクロスポストアプリ。

<https://github.com/azu/postem>

× javascriptはJavaScript。

## Related Item

Add

Submit

# 言葉は意図

- 言葉とは意図を表明するもの
- JSer.infoにおいては次の意図を持たせる言葉(遣い)を優先する
  - 「整理」「正確」「現状」「中立」「関連性」「客観的」
- 逆を言えば使わない言葉を決める
  - 実際に使わない言葉の方が役立つ(機械的に落とせるため)

# JSer.infoで使わない言葉

紹介するサイトもこの内容が強すぎるのは避ける

- 煽りすぎている言葉
- 貶める言葉
- 批判する言葉
- 主張が強すぎる言葉
- 決めつける言葉

# JSer.infoで具体的に使いたくない言葉

単語そのものというよりは言葉遣いという視点から捉え方が難しいことば

- is Dead
- 最強
- 熱い
- 常識
- 知らなそう
- これだけ知って(おけ|れば)
- これがベスト/最も優れた

# Actual case

- [Javascript Fatigue – Medium](#)
- [Promise Cancellation Is Dead – Long Live Promise Cancellation! – Medium](#)
- [なぜ仮想DOMという概念が俺達の魂を震えさせるのか - Qiita](#)
- [Riot.js 2.0 を触ってみた – まだReactで消費しているの? - Qiita](#)



## Note: Actual problem

We apologize to everyone we upset with the JS Awards idea. It was a poor idea and has ended. Let's promote new stuff and ideas instead! ❤️🎉

— [twitter.com/JavaScriptDaily](https://twitter.com/JavaScriptDaily)

- JS Awards (JavaScript Weekly 主催) が中止された問題
- Awards in Open Source – Medium

# JSer.infoで使う言葉

- 代替え方法/手段/ライブラリ
- 特徴
- 目的
- と比較して
- 良いところ/良くないところ
- 可能
- 現状/ステータス/開発中

# 意図としてのCODE OF CONDUCT

- CODE OF CONDUCTはその場所をどう扱ってほしいかの表明
  - 利用者に強制力はないが、ヒントとして表明できる
  - 関連: 契約プログラミング
- [jser.info/CODEOFCONDUCT.md](https://jser.info/CODEOFCONDUCT.md)
  - Contributor CovenantやOpen Code of Conductも基本的にそういう意図を表明しているもの
  - [Introducing GitHub Community Guidelines](#)

We recommend projects consider adopting a code of conduct that fits their community.

— *Followup: Open Code of Conduct // TODO: Talk openly, develop openly*

オープンソースプロジェクトに興味のあるひとは、自分の居心地のよいプロジェクトを探してみよう。

– オープンソースプロジェクトとの距離のとりかた

# このイベントのCODE OF CONDUCT

- 公平な心をもって参加しましょう
- あるものをけなすような発言はぐっと抑えましょう
- 写真の撮影/アップロードなどは写ってる人に許可を貰ってから行いましょう

↗ 上記に反しないような主張や議論は歓迎

# 「言葉」の難しさ

- 正しいことを言っても、正しくは伝わらないことがある
  - 正しいことを言う != 正しく伝える
- 正しい事実だけではなく、正しく伝えることも大事
- 「使わない言葉」であげていたものは、正しく伝えるのが難しい言葉
- 主観的な言葉とも言い換えることができる

# 言葉とツール

- 事実を正しく伝えるのは難しい
  - そこにはコミュニケーションが存在する
- 障壁を下げるには機械を通すなど
  - いわゆるLintを通す場合はそこには人間関係が介在しにくい
  - 文字通り機械的にやり取りできる
- 例) `textlint`で説明をチェックする

# ツールによる検査の利点

人手による文書のレビューはコードのレビュー以上に人間関係を悪くしてしまう恐れがあります。

これに対して、自動検査ツールで結果を返すのは人ではなくツールですので執筆者は嫌がらせを疑う必要はありません。

– *Takahiko Ito*<sup>RedPen</sup>

# JSer.infoにおける機械的なチェック

- `textlint`で機械的なチェックが走ってる
- 機械的なチェックに引っかかったものは修正するか諦める
- 機械的に判断できるようにして、余計な時間を使わないように
  - 時間的なコストを削減するためにも、機械的なチェックは効果的

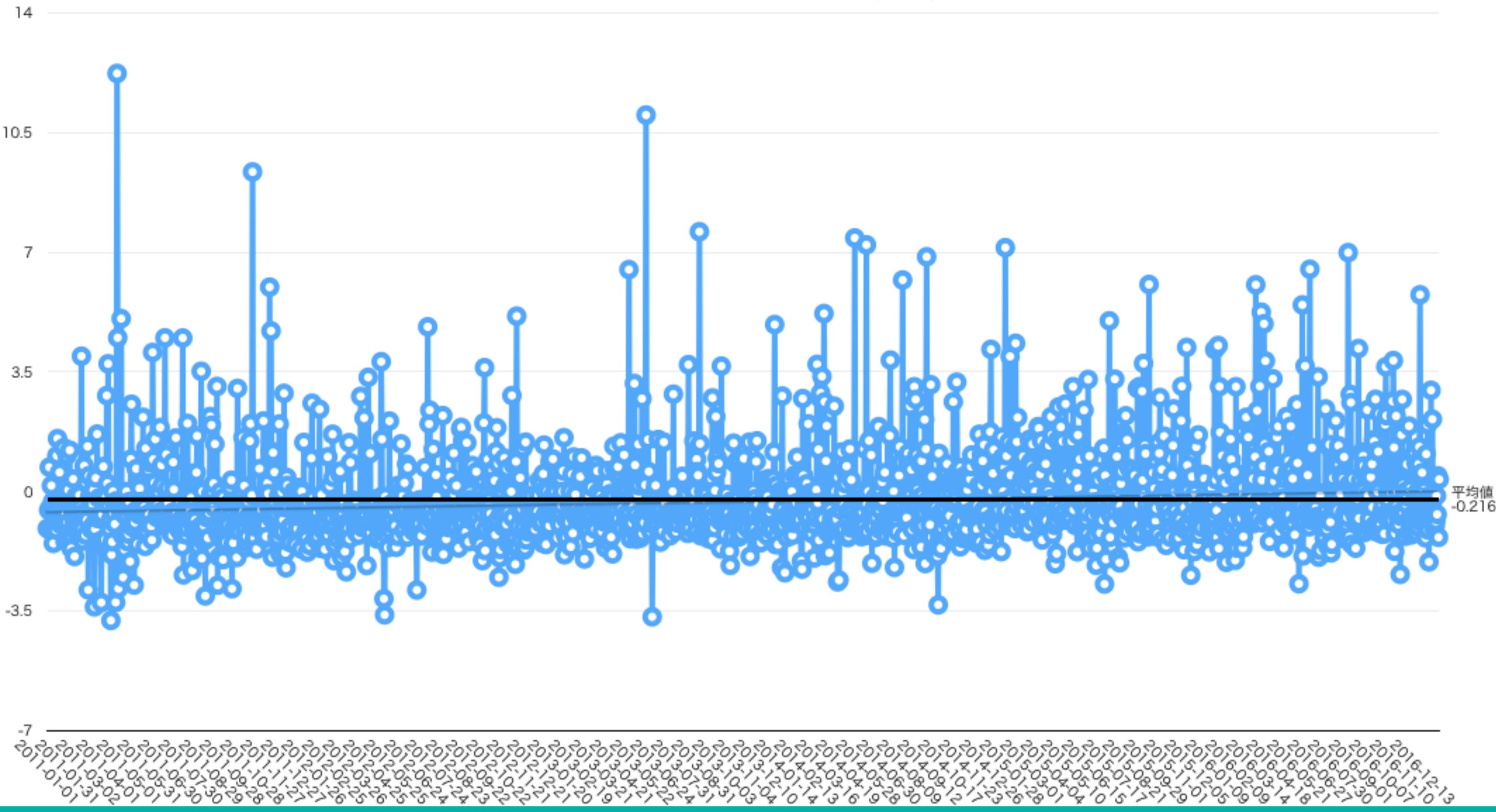
# JSer.infoの感情極性値

- JSer.infoの説明に感情的な単語がどれぐらい使われてるか
- 単語感情極性対応表
  - 単語ごと-1から+1までの値がつけられた辞書データ
  - ネガティブに近いかポジティブに近いか
- 説明文の各単語の感情値の合計値から平均を出してみる

## JSer.infoは中立的

- JSer.infoの意図は「中立」
- 0に近いほど目的に沿ってそう
  - ポジティブ、ネガティブのバランス
  - 記事ごとの説明文の感情極性値の平均を出してみる

○ アイテム毎の感情極性値(平均)



# 「説明」のまとめ

- 説明は場所/対象により正しさが異なる
- 注目を集めることは簡単だけど、事実を伝えることはより難しい
- JSer.infoでは 正しく説明  $\geq$  正しい事実
  - 正しく説明できなさそうなら諦める<sup>1</sup>
- 機械的なチェックを導入することで時間的に省略

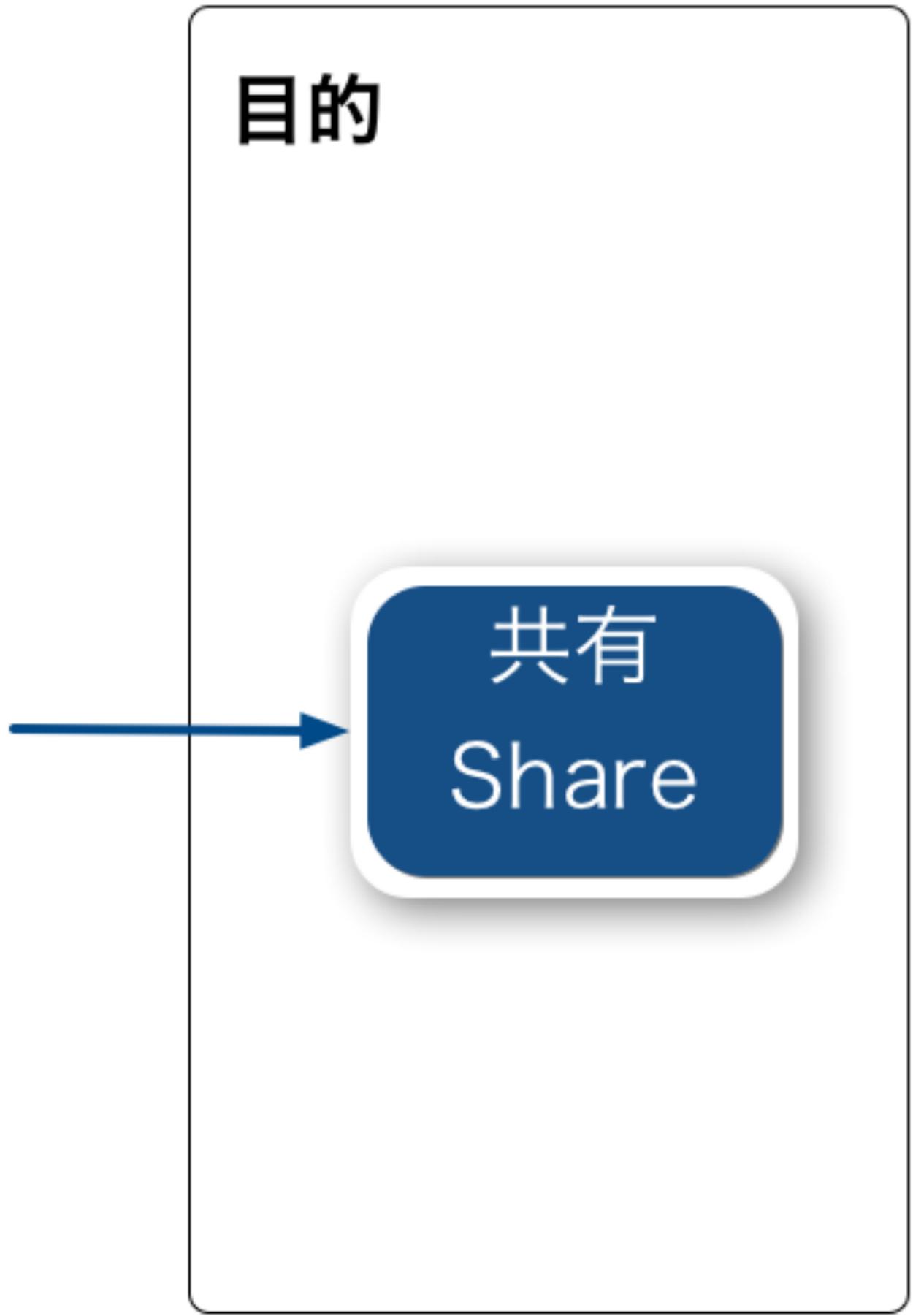
---

<sup>1</sup> 正しい事実が分かっているのに説明できないことは、専門外に多いので専門家に聞くのが良い

共有(Share)

目的

共有  
Share



# 共有(Share)

- 一定数、紹介サイトが溜まったら共有する
- 一定数はBotが教えてくれる

## jser/jser.info - Gitterの様子



bot-user @bot-user

そろそろ記事更新できそうですよ /cc @azu



azu @azu

はい。

# 共有(Share)

1. JSer.infoのアーカイブから記事を選ぶ

2. jser/jser.github.ioに記事を書く

- Jekyllで動いてるので、Markdownを追加してヘッドラインを書く

3. 記事が公開される

The screenshot shows the JSer.info website interface. At the top, there is a calendar for the year 2017, with the month of January (1月) selected. Below the calendar are navigation buttons: 'アーカイブ | Edit index.json on Github', 'FullScreen', 'Copy HTML', and 'Copy Markdown'. The main content area displays a list of articles:

- O'Reilly Japan - 初めてのJavaScript 第3版**  
<http://www.oreilly.co.jp/books/9784873117836/>  
JavaScript, book  
2017年1月20日発売 Learning JavaScriptの翻訳本。ES2015+に対応した内容  
• [初めてのJavaScript 第3版 サポートページ](#) — マーリンアームズ
- wheresrhys/fetch-mock: Mock http requests made using fetch (or isomorphic-fetch)**  
<https://github.com/wheresrhys/fetch-mock>  
Fetch, test, HTTP  
Fetch APIのモックライブラリ
- talyssonoc/structure: A simple schema/attributes library built on top of modern JavaScript**  
<https://github.com/talyssonoc/structure>  
JavaScript, library  
ES2015 classesのモデルに対してスキーマを定義できるライブラリ。スキーマを元にバリデーションやシリアライズを行える
- React Interview Questions**  
<https://tyiermcginnis.com/react-interview-questions/>  
React, JavaScript

On the right side, there is a sidebar with sections: 'ヘッドライン' (Node v7.4.0 (Current) | Node.js, NodeKit), 'アーティクル' (From Sass to PostCSS by Tyler Gaw, Flow Runtime), 'スライド、動画関係', and 'サイト、サービス、ドキュメント'.

----- ここまで -----

これを繰り返せば JSer.info の完成

# JSer.info

JavaScriptの最新情報を紹介する週刊ブログ

日本語 | 한국어

リアルタイム版 => [Realtime JSer.info](#)

## 投稿一覧

### [2017-01-11のJS: Node.js v7.4.0とnpm v4、PhantomJS 2.5.0 Beta、クリーンコード](#)

JSer.info #313 - Node v7.4.0がリリースされました。BufferやEventEmitterのパフォーマンス改善が含まれています。また、Node.js v7.4.0では、npm v4.0.5が同梱されるよう...

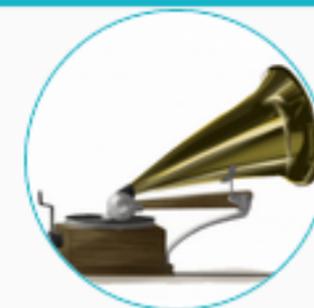
2017年01月11日

### [2017-01-04のJS: PostCSS概要、Chrome開発者ツール、FuseBox](#)

JSer.info #312 - PostCSS まとめ - Qiitaという記事では、PostCSSとはどのようなものなのかについて書かれています。PostCSSはCSSのツール基盤となるフレームワークで、CSSパーサやジェネレ...

2017年01月04日

[2016-12-27のJS: SharedArrayBufferとAtomic API、Node.js Stream、JavaScript](#)



[azu](#)



Enter Search



ページ

# 継続性

- JSer.infoの目標 
  - 「更新コストを小さくして、継続できる形を作る」

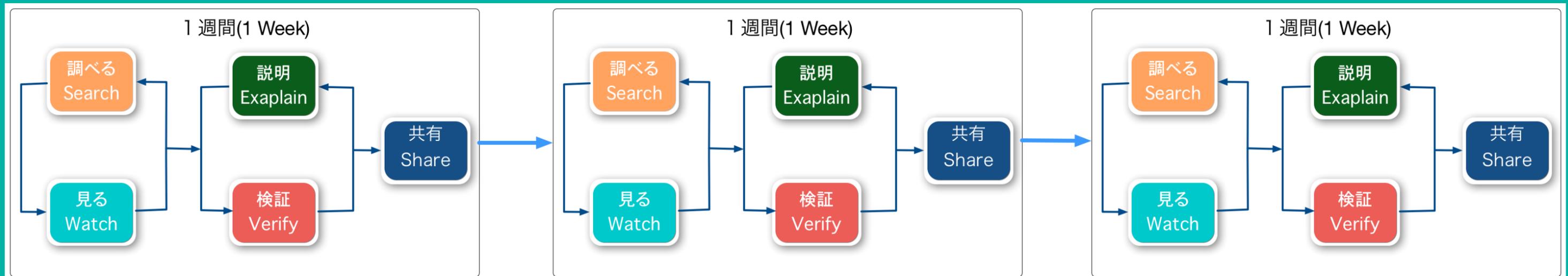
# ゴール(Goal)

- 継続的に行動するには、ゴールはできるだけ短く連続的に設定する
  - 「JSer.infoは1週間に1度ぐらい投稿する」というのがゴール
- ゴールと進捗は同じぐらい大事な指標
- [jser.info/status-of-post/](https://jser.info/status-of-post/)で現在の進捗を見られる
  - 記事の紹介文書きまでは分散して行われる(まとめる = 記事化は週一)



# 短く連続性のあるゴール

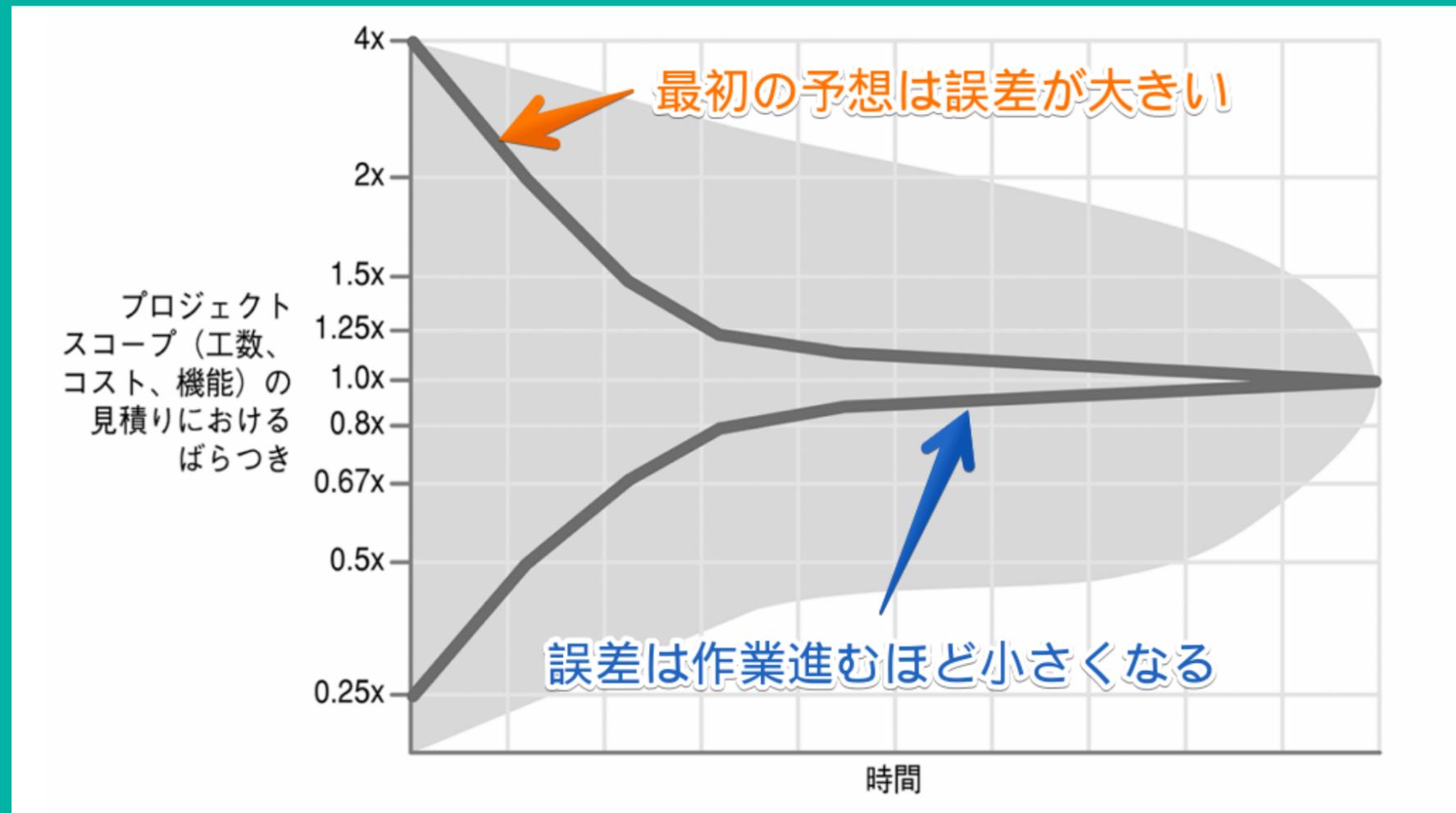
- 一度に遠くのゴールを目指すより、短い目標をちょっとずつクリアした方が継続する



# 長すぎるゴールの問題

- そもそも解決するのが難しい
- どれくらいの期間で終わるのか予想しにくい
- 直感的な見積もりは大体間違ってる

# 不確実性のコーン 見



# 直感と予想は一致しない

- 長期的なものを直感で予想するのは難しい
- 短い反復の方が、予想と実際の結果のバラつきは小さくなる
  - 計画実行の不確実性が減る
- 一度上手くいかなかったても、失敗の誤差も小さいため取り戻しやすい

# 再開に必要なコストは大きい

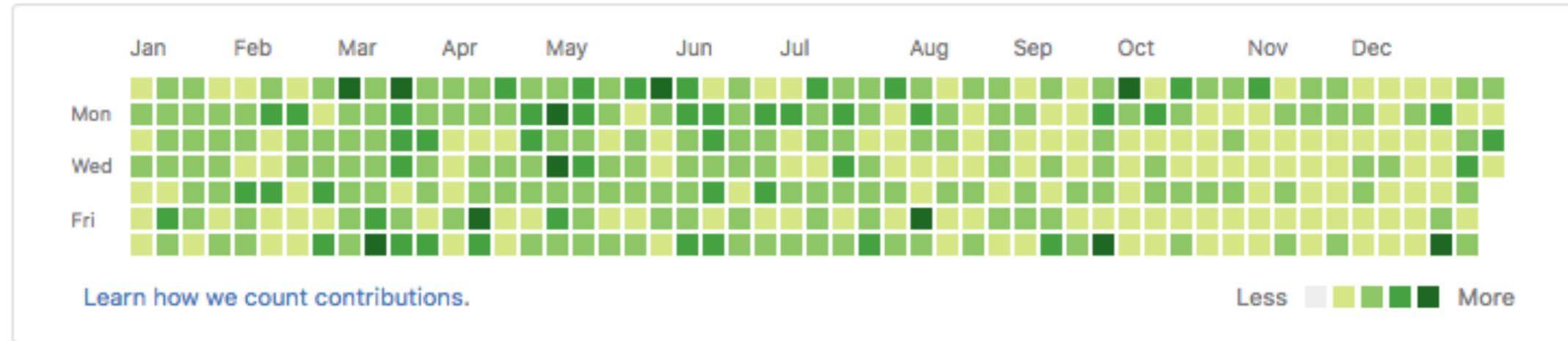
- 一度やめると再開するコストが大きい
  - 上手く達成できなかったタイミングで停止してしまうことが多い
- 1週間区切りなら上手く行かない週は、次の週で取り戻せる(リセットできる)
- 止まることは想定し、復旧することを前提にする
- 関連: [let it crashが生んだ誤解 - Qiita](#)

# 小さく作って小さく更新

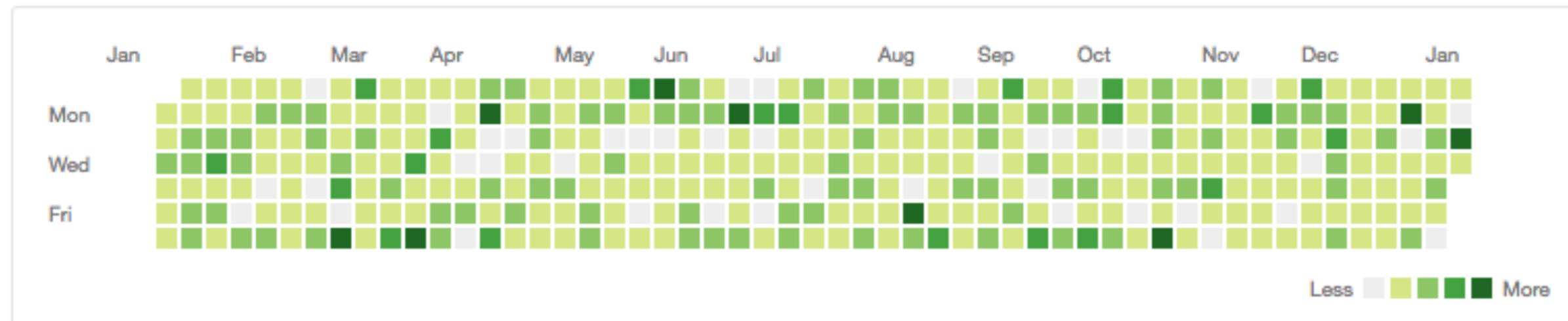
- 一日で一週間分をまとめるのは無理
- 小さくコミットして小さく続ける(VCS)
- 1紹介記事 = 1コミット
  - 一度にまとめてやるのは心理的コストが高い
  - 分散的に更新したものが、結果的にマージされていればいい

## 9,460 contributions in the last year

Contribution settings ▾

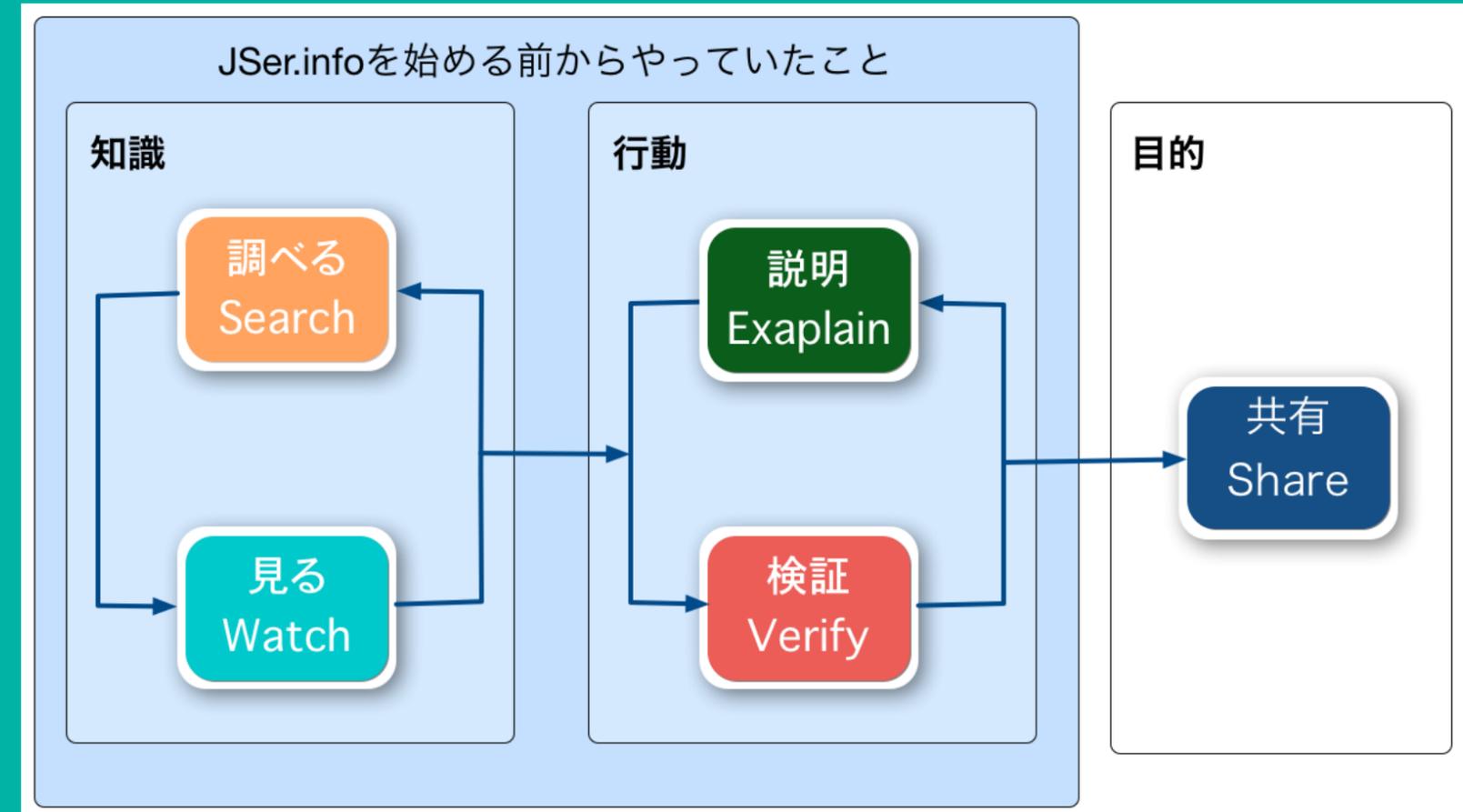


## 1075 items in the term



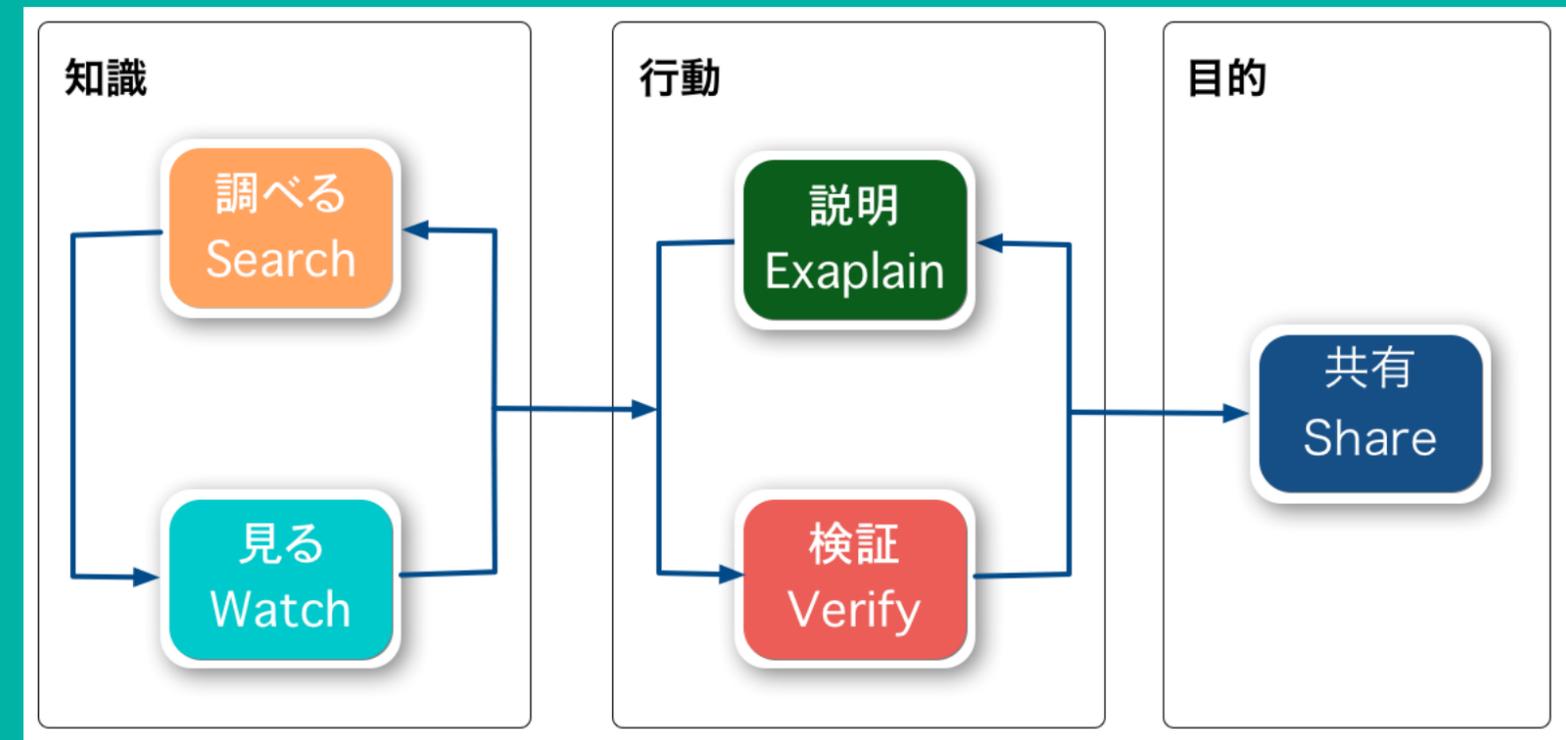
# 共有の位置づけ

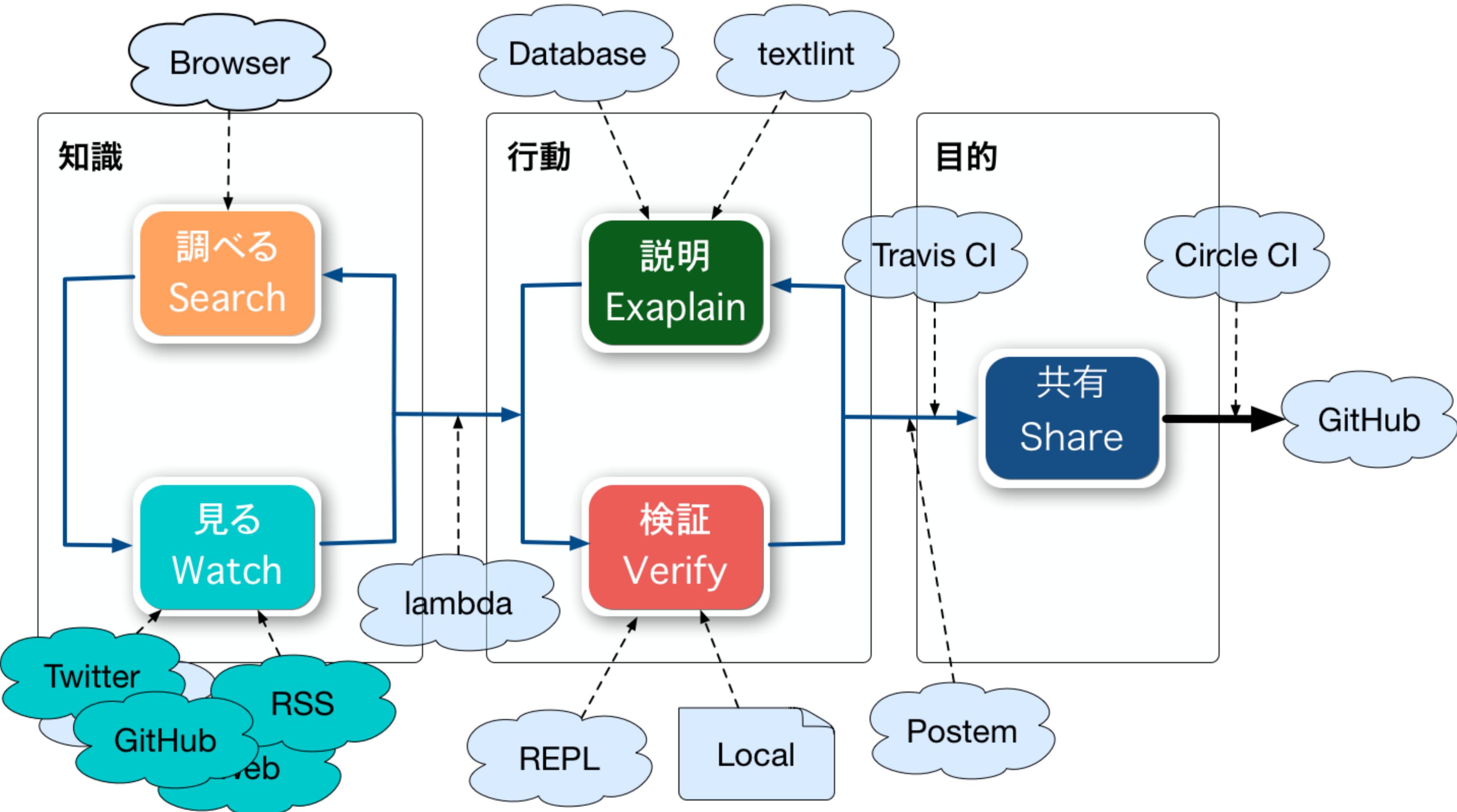
- 情報収集は元からの趣味
- 共有をするのはおまけ
- JSer.infoの更新(共有)を停止しても、情報収集自体は継続される
- 完全に停止はしないので、再開はしやすい



# イテレーションのコスト

- 無コストではないけど、そのコストは小さくするように努力する
- ワークフローはできるだけ一方通行で完了するようにする
  - Unidirection workflow
- textlint、CI、bot、API、投稿アプリ
  - その場その場で投稿の処理を行い、一つ一つのコストを小さくする





# あなたもJSer.infoをつくってみませんか？

- [jser/ping](#)にURLを投げる
  - 紹介するかは別として、とりあえず見ます。
- [jser/report](#)にアイデアを提案する
  - より密度が高いレポート形式でやる方法を考えています。
- [jser/jser.github.io](#)に寄稿する
  - 寄稿する仕組みはないですが、それがJSer.infoらしいならやるべきです。
- [jser/jser.info - Gitter](#)で議論をする

# まとめ

- その場における知識はそこまで重要じゃない
- 見るものは見る場所に集める
- 直感が正しくないときに、それを推し量る方法を持つ
- 自分の動けるスコープを決めて動くことが大事
- JSer.infoは1週間で小さな目標を達成出来るように動かしてる
  - 1つずつの紹介についてのタスクは細分化されている

# 参考 1

- JSer.info 1年を迎えて
- 世界のJavaScriptを読もう @ 2012
- The Mechanism is not the Mental Model
- プログラミング言語標準化のパターン
- 今日からはじめる情報設計
- ソフトウェア見積り 人月の暗黙知を解き明かす

## 参考 2

- [textlint](#) - pluggable linting tool for text and markdown
- [行動規範マナー動画](#) in ScalaMatsuri 2016 - YouTube
- [Open Resources/Japanese Sentiment Polarity Dictionary](#) - 東北大学 乾・岡崎研究室 / Communication Science Lab, Tohoku University
- [let it crashが生んだ誤解](#) - Qiita