Webアクセシビリティを身近にする

アクセシビリティの捉え方と実践

@kuy / 2021.12.18 - Tech BASE Okinawa

自己紹介

- **@kuy / 小玉 祐輝**
- 株式会社ユーザベース B2B SaaS 事業 Fellow / INITIAL CTO
- フロントエンドが好きで6年前にReactを本番投入しました
- 最近はRustを書いてます

ユーザベース、経済情報に特化したオンライン番組配信事業「NewsPicks Stage.」を開始

2021.12.16

株式会社ユーザベースは、経済情報に特化したオンライン番組配信事業「NewsPicks Stage.」を開始いたします。2022年1月19日(水)には「SPEEDAトレンド #6 スマートシティで日本を変えるII 」の配信を予定

Webアクセシビリティ

Webアクセシビリティとは？

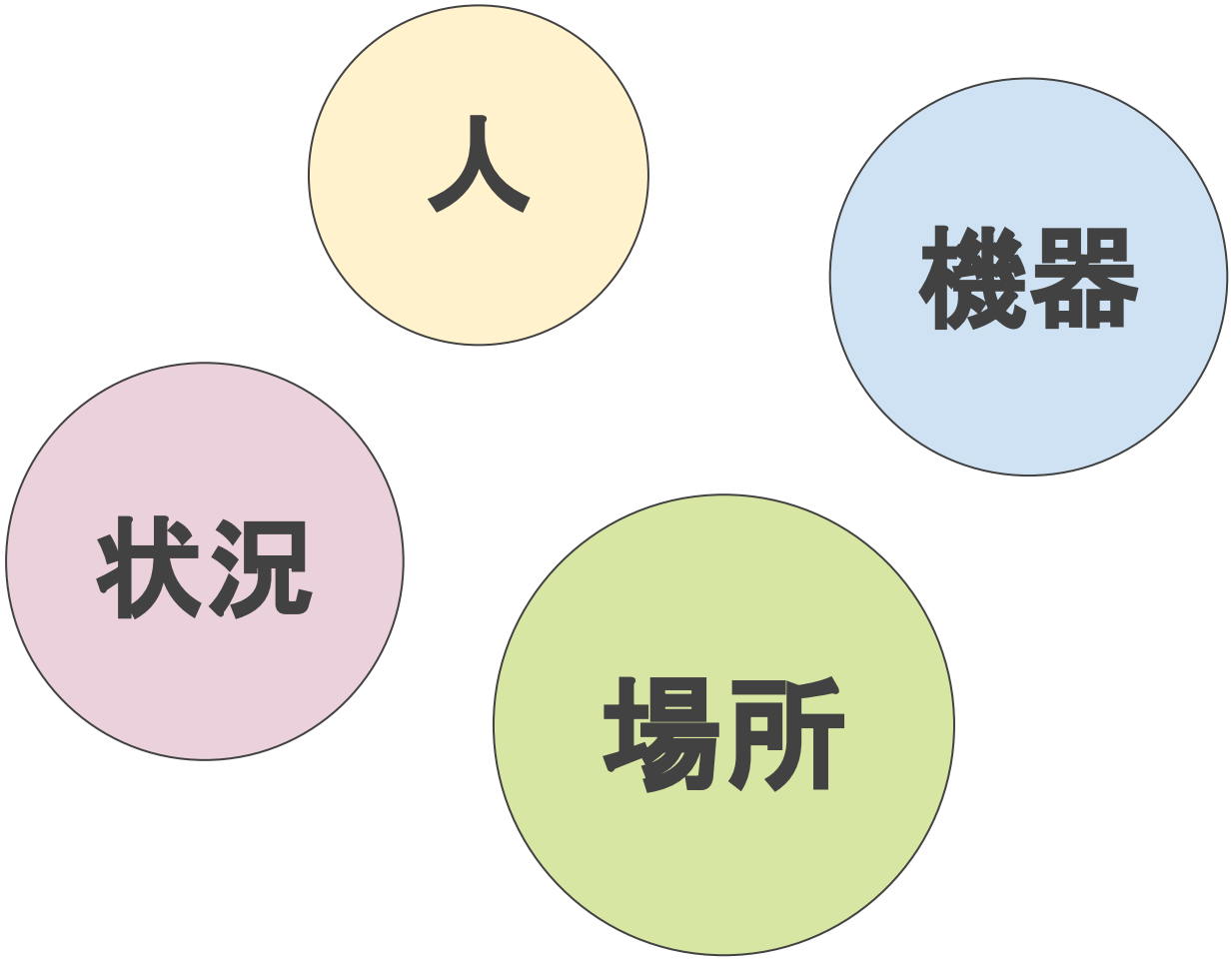
障がい者や高齢者がWebを利用できるようにすること。

Ritam Baishya via Unsplash

想像力を膨らませてみる

- 直射日光が当たる屋外でノートPCを操作する
- 視力が悪くて見づらい
- 加齢によって徐々に老眼になる
- マウスが使えない状況で正確にポイントできない
- 電車に乗ってて音を出せない
- 画面の小さいスマートフォンから閲覧する

様々な組み合わせ



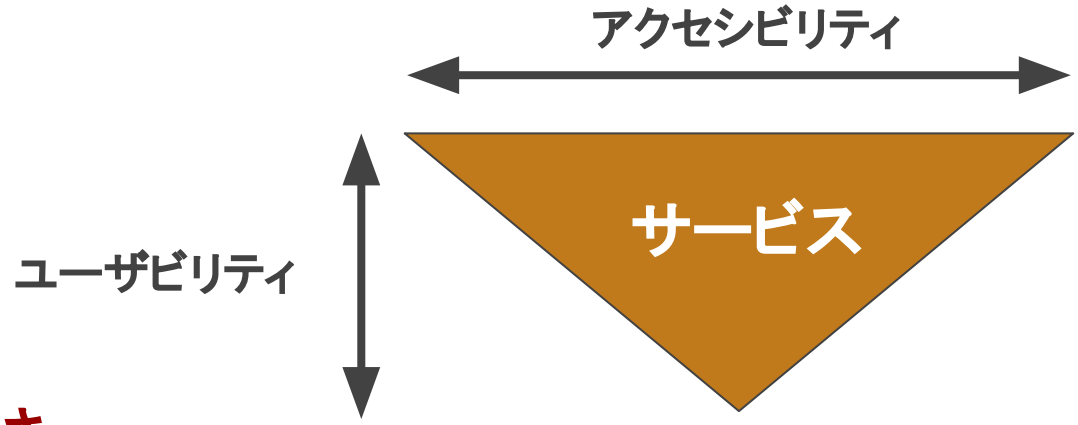
実はとても身近なこと

- アクセシビリティを「必要とする/しない」のゼロイチではなく、**グラデーション**になっている
- **誰もが**アクセシビリティの恩恵を受けることができる

Webアクセシビリティとは？

誰でも、どんな状況でも、どんなデバイスを使っていても利用できるようにすること。

ユーザビリティとの関係



- アクセシビリティは**広さ**
  - 利用できる人を増やすイメージ
- ユーザビリティは**深さ**
  - ISO 9241: 特定の利用状況において、特定のユーザーによって、ある製品が、指定された目標を達成するために用いられる際の、有効さ、効率、ユーザの満足度の度合い。
- アクセシビリティが高いと、ユーザビリティも高いとは限らない。逆もしかり。

なぜアクセシビリティに取り組むのか？

なぜアクセシビリティに取り組むのか？

どんなに素晴らしいサービス/コンテンツを作っても、ユーザーが利用できなければ意味がない。

https://initial.inc

なぜアクセシビリティに取り組むのか？

どんなに有用なデータや記事コンテンツがあったとしても、ユーザーが利用できなければ意味がない。

限られた人が使う  **より多くの人が触れる**

アクセシビリティテストの自動化

Q： Webの表示速度、測定したことありますか？

原体験: Webページの表示速度

- 古き良きWebの時代
- テキスト以外のコンテンツ（画像、動画、音声）
- 「Webページ」から「Webアプリケーション」へ
- SPA構成、フロントエンドヘビー

Webが遅い！

手軽に計測できるツールの登場

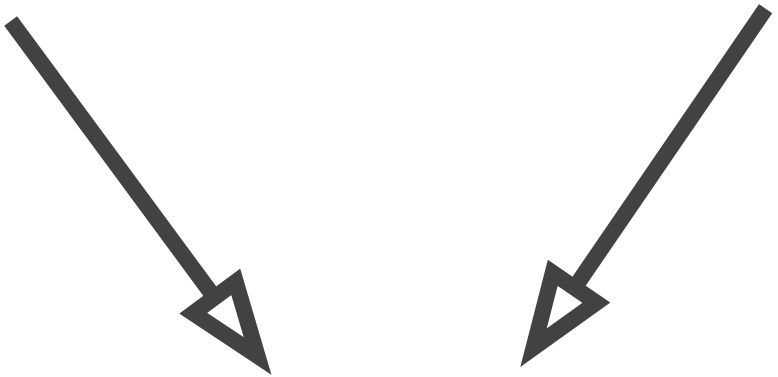
- YSlow
- Google PageSpeed Insights
- Google Lighthouse
    - 現在はChrome DevTools組み込み

どんどん手軽に

仮説

手軽に現状把握できる

繰り返し実行するのが容易



取り組む人が増える（カルチャーの醸成）
継続的な改善

自動化できそうなツール候補

- axe / Lighthouse / Chrome DevTools（DOMのみ）
- Visible: ACTベース、ドライバ変更可能
- Wave: Webサービスとして提供。
- Alfa: ACTベース
- **acot**

acot

Accessibility Testing Framework

https://github.com/acot-a11y/acot

- Puppeteerを使うことで動的な検証が可能
- ESLintライクなルール構成
- AOM (Accessibility Object Model) を活用

acot

手軽に使える

1. 開発サーバーを起動する (例: localhost:3000)
2. `npm install --save-dev @acot/cli puppeteer` でインストール
3. `npx acot init` で初期化
4. `npx acot run` で実行

最初からすべてを有効にする必要はないので、
基本的なルールから始めてもよい。

acot

`@acot/wcag/img-has-name`

- 非テキストコンテンツの代替テキスト
- アクセシビリティと言うと真っ先にこれを思い浮かべる人も多いはず
- SPAなど動的にDOMが生成されるアプリケーションだとつい忘れがち

最初からすべてを有効にする必要はないので、
基本的なルールから始めてもよい。

acot

`@acot/wcag/interactive-has-enough-size`

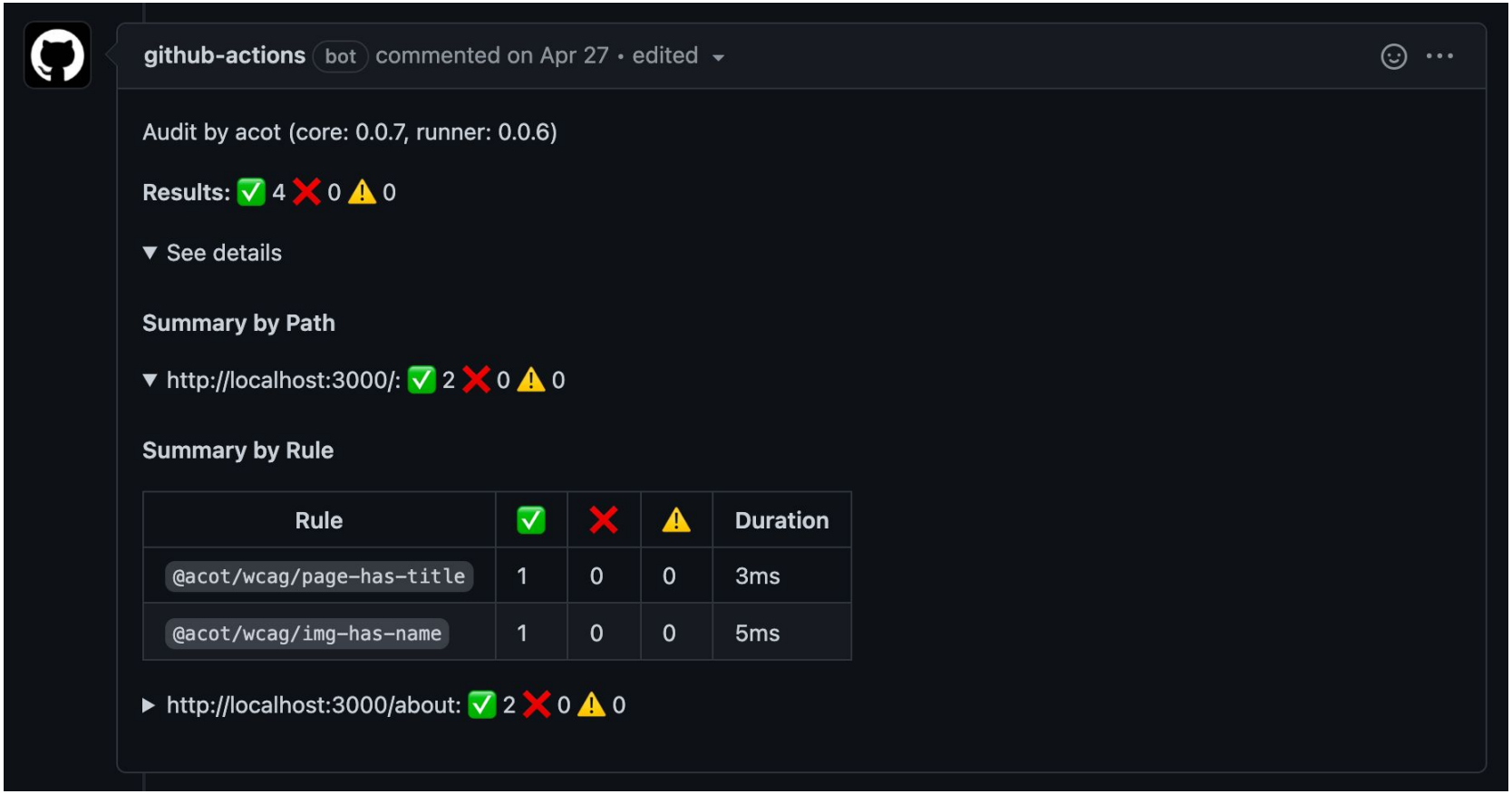
- ターゲットのサイズ
- 44x44が基準
- タッチパネルやスマートフォンなどポインティング精度の粗い機器で操作性を担保するために重要

GitHub Actionsに組み込む

- CIに組み込んでアクセシビリティが悪化してないか自動的にチェックできる
- GitHub Marketplaceからワンクリックで導入できるのが目標（手軽に試せる）

GitHub Actionsに組み込む

実行結果



Webアクセシビリティを高める

WCAG:
Web Content Accessibility Guidelines

いきなり **WCAG 2.1** から始めるのはハードルが高い・・・
WCAG 2.1 解説書 の「達成基準」は最初の入り口としてとてもよい。

達成基準: A, AA, AAA

- 代替テキスト: レベルA
- ターゲットのサイズ: レベルAAA

徐々にユーザーの対象範囲を広げていける。

セマンティックなマークアップ

- 意味のあるHTMLタグを使う
- WAI-ARIAの利用は最小限に留める
  - ARIA: Accessible Rich Internet Applications
  - Webアプリケーションをアクセシブルにする手段を提供

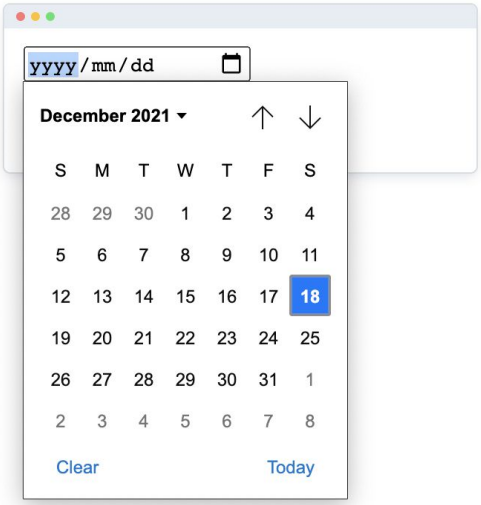
セマンティックなマークアップ

- 例1: ボタンのように振る舞うコンポーネントをdivタグとCSS、JavaScriptを駆使して作る
- 例2: 独自のスクロールバーを実装する
- 例3: リンクはaタグとhref属性を使う
    - ときどきマウスカーソルをホバーしてもURLがステータスバーに表示されないアプリケーションはこれを守れていない

セマンティックなマークアップ

ブラウザ側の実装を活用する

- ひと昔前までは自作するのが当たり前だった「カレンダーの日付選択」も `<input type="date" …>` を活用できる
- とはいえ、ユーザビリティを犠牲にしては元も子もないので、アクセシビリティを確保した上で独自実装に踏み切るという選択肢はありえる



まとめ

- アクセシビリティはとても身近なもの
- ゼロイチではなく、ちょっとずつ改善できる
- 現状を把握することから始める
- 自動化によって定点観測する

ご清聴ありがとうございました。

株式会社ユーザベース