

インタラクティブな ICT教材の実装に DSLで立ち向かう

chibivue-land/japanese-companies-using-vuejs



日本で Vue.js を使っている企業一覧

27

Contributors

1

Issue

61

Stars

26

Forks



[README](#)

日本で Vue.js を使っている企業一覧

日本で Vue.js を使っている企業の一覧です！

[CONTRIBUTING.md](#) を参照して、企業情報を追加してください！PR お待ちしています！

企業一覧 (Pull Request 順)

企業名	寄稿者	寄稿日	補足
株式会社メイツ	@ubugeeeeii	2025-02-09	Web プラットフォームのアプリケーションを主軸とした EdTech 企業です！ほぼ全てのプロダクトで Vue.js (Nuxt) を使っています！エンジニア募集中！
株式会社ブレイド	@kazupon	2025-02-09	ブレイドの主要サービス「KARTE」は、Webサイトやアプリ上でのユーザー行動をリアルタイムに解析し、エンドユーザーに最適な体験を提供するための「CX（顧客体験）プラットフォーム」です。2014年からVue.jsを採用し、KARTEの主要な機能の多くはVue.jsで実装されています。エンジニア絶賛採用中！
株式会社 Schoo	@yamanoku	2025-02-10	「世の中から卒業をなくす」をミッションに、インターネットでの学びや教育を起点とした社会変革を進めています。主な事業は社会人向けオンライン学習サービス、法人向けオンライン研修、地方創生。
合同会社 Steg	@kspace-trk	2025-02-10	合同会社Stegは、スタートアップや企業様向けに、SaaS開発や業務システム、Webサービスの設計・開発を手掛けています。ほぼ全ての開発にNuxtを使用し、エンジニアのみならずデザイナーやマーケターもNuxtを活用し、スピーディーで柔軟な開発を実現しています。
合同会社世路庵	@448jp	2025-02-10	東京・中野にあるウェブ制作会社です。ビジネスとクリエイティブの両立を強みとし、創業17年以上の経験と1,000件以上の実績を持っています。Vue.js/Nuxtだけでなく、React/Next.jsな

About

日本で Vue.js を使っている企業一覧

- [Readme](#)
 - [Activity](#)
 - [Custom properties](#)
 - [61 stars](#)
 - [4 watching](#)
 - [26 forks](#)
- [Report repository](#)

Releases

No releases published

Contributors [27](#)



[+ 13 contributors](#)

株式会社メイツ



ナイトウ コウスケ

株式会社メイツ(2025.02~)
フロントエンドエンジニア

Composition API 生まれ script setup 育ち
 Vue.js 3.2~

今日話すこと

株式会社メイツ

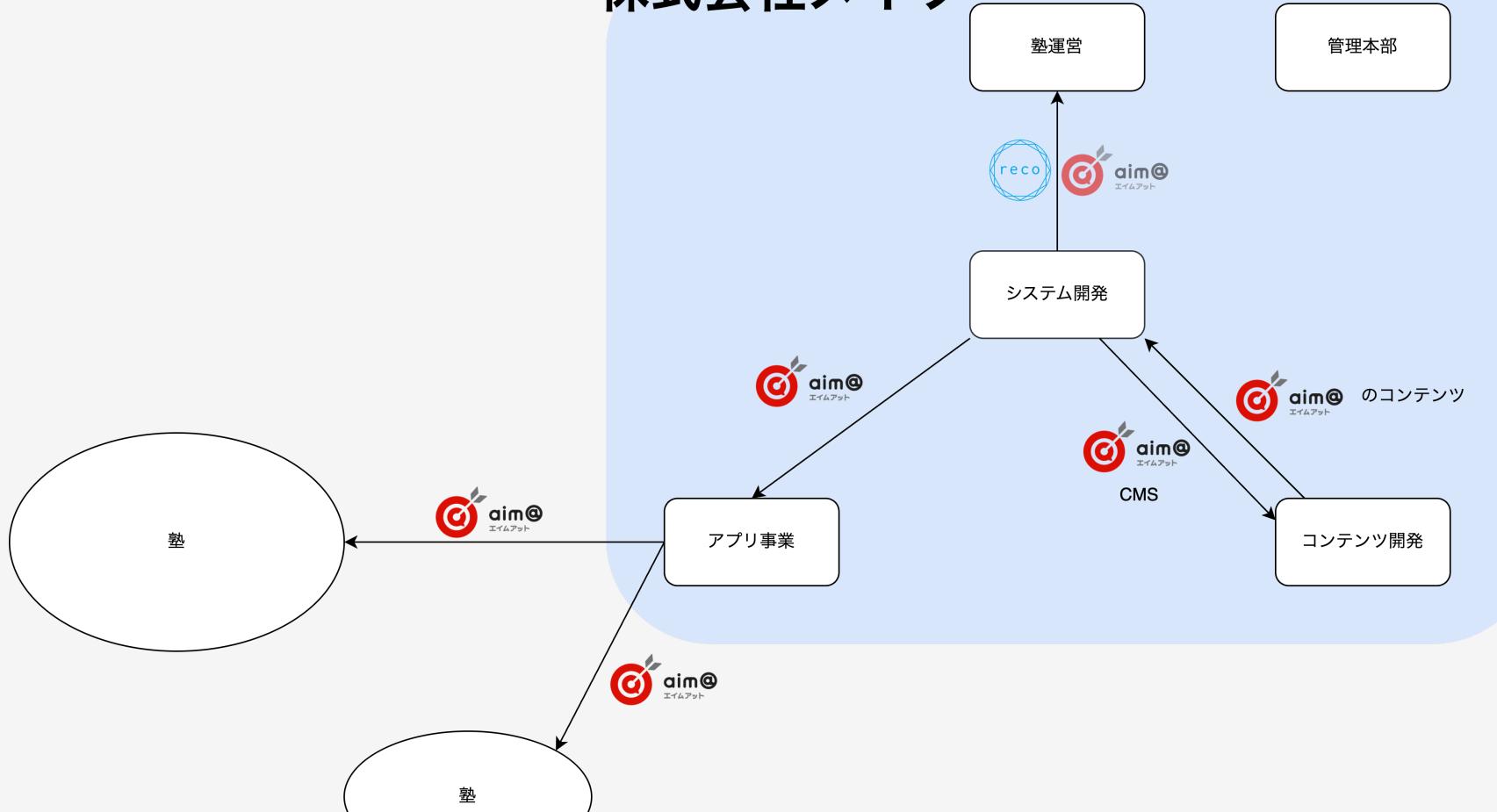
ICT 教材開発の課題

aim@ contents definition language

株式会社マイツ



株式会社メイツ





Web アプリ

豊富な問題/解答形式

マルチデバイス・PWA 対応

時間管理、動画・音声ストリーミング、etc...



学習教材のドメイン知識は複雑

(一般的な話)

多様な学習形式と多様な問題形式

テスト

一問一答

ドリル

模試

講義動画

インプットのための教材

択一

複数選択

穴埋め

長文記述

正誤

書き順

並び替え

数式

画像問題

長文読解

グラフ

リスニング

スピーキング

作文添削



aim@
エイムアット

で扱う問題形式

ChoiceFormat

単一選択/複数選択

複数重複選択

選択肢の解除機能

選択肢のシャッフル

選択肢に数式

解答欄の結合

WriteFormat

様々なキーボード
textarea (作文)
解答欄の伸び縮み
解答欄のクリア

KanjiHiraganaHandWritingFormat

手書き
書き順
消しゴム
undo

全問題形式共通

マルバツ

模範解答

採点

etc...

中2 ▶ 単元別演習
連立方程式・基礎

問題数 1/5 問目 正解 0 問 再挑戦 0 問

(1)次の連立方程式を解きなさい。

$$\begin{cases} x - (y + 3) = 2 \\ \frac{3}{2}x + \frac{y}{3} = -9 \end{cases}$$

$x = \boxed{2 \cancel{\times}}$, $y = \boxed{6}$

$x - y = 5 \cdots \textcircled{1}$

連立方程式の下の式 $\frac{3}{2}x + \frac{1}{3}y = -9$ の両辺に6をかけて係数を整数にすると,
 $9x + 2y = -54 \cdots \textcircled{2}$

$\textcircled{1} \times 2 + \textcircled{2}$ より,
 $11x = -44$ これを解いて, $x = -4$ となる。
 $x = -4$ を $\textcircled{1}$ (または $\textcircled{2}$)に代入して,
 $-4 - y = 5$ これを解いて, $y = -9$ となる。
したがって、答えは $x = -4$, $y = -9$ である。

AIに詳しい解説を聞く

^ つぎへ

実際のコンテンツ開発とその課題

CSV + 問題画像時代の課題

開発者的にも

- CSV のバリデーションがめんどい
- コンテンツ作成時のプレビューが大変
- 画像データなのでレスポンシブ対応が大変 (画像の拡大縮小くらい)
- PDF は扱いたくない、、、



aim@
エイムアット

(CSV 時代)の課題

プレビューが大変

バリデーションが大変

画像などのパスを CSV に書いてた

DSL で立ち向かう

何でどうやってコンテンツを作る？(ベース)

HTML ? 

Markdown ? 

Latex ? 

独自フォーマット ?

LaTeX^{LATEX}

○表現力

数式や図表の表現力が高い

△Webとの親和性

HTMLに変換できるけど...

△馴染みない開発者も少なくない

開発者都合だけど...

○ Web での表現力

✗ 手書きが面倒

タグを全部書くのが大変

数式も MathML で書く... ?

Markdown



- 比較的書きやすい/読みやすい
- 拡張が容易

既存のプロセッサやプラグインが利用できる(markdown-it など)

HTML を書きたい！

(手書きは大変)

HTML を簡潔に書きたい！！

基本的には Markdown で

Latex などは局所的に

A C D L

Aim@ Contents Definition Language

Aim@ Contents Definition Language

Markdown 拡張した DSL

(Domain Specific Language)

基本機能

Katex

ルビ、画像、動画

sanitize

aim@ ドメイン固有の特徴

塾固有のレイアウトや専用のタグ

問題ごとのメタ情報

1 問の問題文、ヒント、解説、メタ情報を 1
つのファイルで記述

ACDL のランタイム

ユーザー入力

自作ソフトウェアキーボード
入力フィールドの状態管理
解答のバリデーション

メディアの取り扱い

ストリーミング
プレイヤー
認可
cleanup の処理

インタラクション実装 (ランタイム) の課題

これらをうまく解決したい...

▼ と生 DOM の組み合わせ

生 DOM 操作の命令的な記述

▼ のスケジューラー (nextTick())

ランタイムの実装のアプローチ

実装を 2 つの空間に分離する

Infra

Bridge の実装
UI 更新の仕組み

Impl

Bridge を使った実装
ビジネスロジック

Infra のアーキテクチャ

宣言的な UI の状態
(デカい reactive object)

HTML にインタラクティブな機能を注入

ACDL → HTML

```
## 因数分解小テスト

### 1. 因数分解しなさい

(1) $9x^2 - 16 = $ <aim-input />



<h2>因数分解小テスト</h2>
<h3>1. 因数分解しなさい</h3>
<p>
(1)
<span class="katex">
<span class="katex-html">
<span class="base">
<span
    class="strut"
    style="height:0.897438em;vertical-align:-0.08333em;"></span>
<span class="mord">9</span>
<span class="mord">
<span class="mord mathnormal">x</span>
<span class="msupsub">
<span class="vlist-t">
<span class="vlist-r">
<span class="vlist" style="height:0.8141079999999999em;">
<span style="top:-3.063em;margin-right:0.05em;">
<span class="pstrut" style="height:2.7em;"></span>
<span class="sizing reset-size6 size3 mtight">
<span class="mord mtight">2</span>
</span>


```

Bri

宣言
問題

```
const bridge = ref<MarkdownQuestionBridge | null>(null);

interface MarkdownQuestionBridge {
    questionFormatsBridges: QuestionFormatBridge[];
}

type QuestionFormatBridge =
    | SingleFormatBridge
    | SpeakFormatBridge
    | ChoiceFormatBridge
    | WriteFormatBridge
    | SelfScoringFormatBridge;
// ...
```

Bridge

Question Format

キーボード情報

正誤

QuestionField
(InputElement とそのラッパー)

入力された値

class のリスト

Hydration

DOM と Bridge のリンク

Vue の世界の外で DOM 操作

```
hydrate({  
  markdownQuestionBridge,  
  inputElementClassName,  
  // ...  
});
```

Hydration Steps

3. HydratedHandler の生成

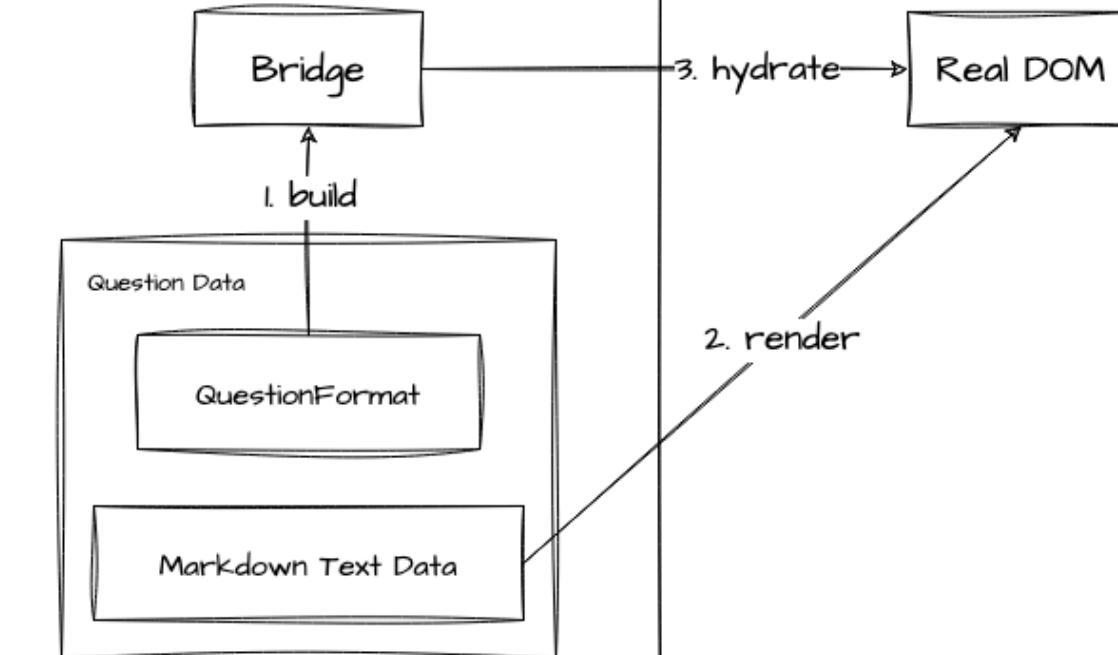
```
function hydrate(ctx: HydrateContext): HydratedHandler {
  // ...
  const update = () => {
    try {
      internalUpdate(ctx, real);
    } catch (e) {
      // error handling
    }
  };
  // HydratedHandler
  return {
    update,
    // ...
  };
}
```

```
// bridge を生  
bridge.value  
  
// DOM を生成  
renderedQues  
  
// bridge を  
nextTick(() =>  
    hydratedHai  
});
```

Bridge を wat

```
const bridge  
watch([bridge],  
    deep: true  
});
```

Vue Composable



});

ここまでで Infra は揃った
Bridge を操作することで UI
が更新される

Bridge を使って UI に関する
ビジネスロジックを実装し
ていく (Impl)

中でも DSL によってレンダリングされた部分からのイベント (input, focus, blur, etc...) は **UserAction** という名前で実装されている

時間切れ→自動採点→正誤が表示

学習フェーズ (回答中、採点後、見直し) の変更→ UI の更新

選択肢選択のビジネスロジック (コンテンツの設定によって挙動が変わる)

すでに選択されている選択肢の選択 (追加されるのか、解除されるのか)

数式入力の不正な入力に対するバリデーション

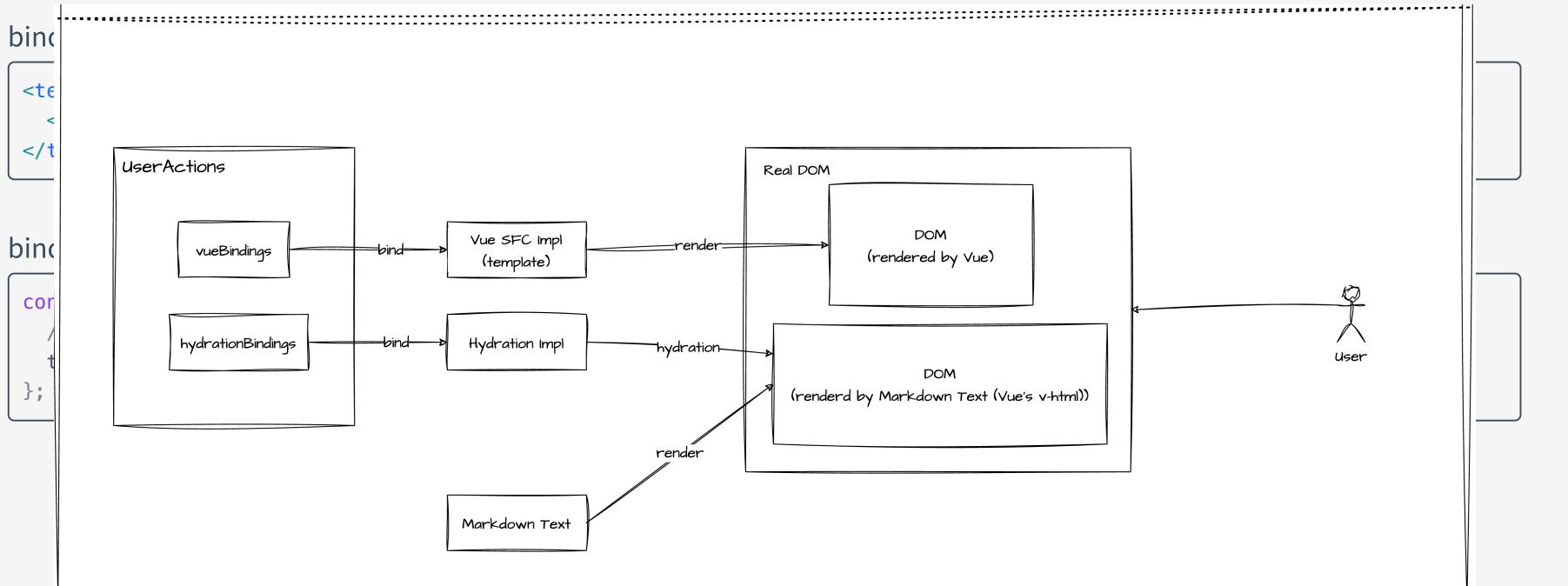
累乗のネストや 2 重根号、半分数の禁止、etc...

左右キーの押下で解答欄移動、etc...

キーボードごとに多様なアクションが実装



UserAction の 2 つのバインディング方法



Infra

Bridge: UI の状態管理

Render: ACDL → HTML

Hydration: DOM 連携

Impl

ビジネスロジック
ユーザーアクション
UI 更新

DOM 操作の安全性と Vue の宣言性を両立

DSLでインタラクティブ教材に立ち向かう

DSL でコンテンツを開発することで、学習体験の UX を改善した
CSV だと問題画像と解答欄が分離していた

レスポンシブ対応

画像ではなく、

TODO: まとめ TODO: 何をして何ができるようになったか

今後の課題

レガシー CSV コンテンツの負債

膨大なフォーマット、複雑怪奇な実装...

抽象化の推進、リファクタリングが急務

テストと品質保証

要件が複雑すぎて、動作確認が非常に大変

テストの拡充と改善

ACDL 自身の進化

より表現力豊かに、より使いやすく

Time's up!

Creator of aim@ is here today ...lol

(Ask him anything!)