

SLIDEV Docs - 日本語



Sliddev

Table of contents

• Ja - README	4
• Ja - TRANSLATIONS	5
• Builtin - Components	8
• Builtin - Layouts	10
• Custom - Config katex	14
• Custom - Config mermaid	15
• Custom - Config monaco	16
• Custom - Config shortcuts	18
• Custom - Config vite	19
• Custom - Config vue	20
• Custom - Config windicss	21
• Custom - Directory structure	22
• Custom - Fonts	26
• Custom - Global layers	28
• Custom - Highlighters	30
• Custom - Index	32
• Custom - Vue context	34
• Guide - Animations	36
• Guide - Drawing	40
• Guide - Editors	42
• Guide - Exporting	45
• Guide - Faq	46
• Guide - Hosting	49
• Guide - Index	52
• Guide - Install	55
• Guide - Navigation	57
• Guide - Presenter mode	59
• Guide - Recording	60
• Guide - Syntax	61
• Guide - Why	70
• Resources - Covers	73
• Resources - Learning	74
• Ja - Showcases	75
• Themes - Gallery	76

- Themes - Use _____ 77
- Themes - Write a /theme _____ 78

sli.dev

Slidevの日本語ドキュメントです

翻訳一覧

	Repo	Site	Maintainers
English	docs	sli.dev	@antfu
简体中文	docs-cn	cn.sli.dev	@QC-L @Ivocin
Français	docs-fr	fr.sli.dev	@ArthurDanjou
Español	docs-es	es.sli.dev	@owlnai
Русский	docs-ru	ru.sli.dev	@xesjkeee
Việt Nam	docs-vn	vn.sli.dev	@bongudth
Deutsch	docs-de	de.sli.dev	@fabiankachlock
Português (BR)	docs-br	br.sli.dev	@luisfelipesdn12
Ελληνικά	docs-el	el.sli.dev	@GeopJr
日本語	docs-ja	ja.sli.dev	@IkumaTadokoro

ローカルでサーバーを立ち上げる

```
npm i -g pnpm
pnpm i
pnpm run dev
```

コマンドを実行したら `http://localhost:3000` にアクセスしてください。

または [Vite extension for VS Code](#) をインストールすると、サイドバイサイドで編集できます。

翻訳に貢献する

[TRANSLATIONS.md](#) をご一読ください

[Go to TOC](#)

Help on Translating

First of all, thank you for being interested in contributing to translations!

You can find the repositories for each existing translation in [README.md](#). To help improve them, simply sending a Pull Request to their repo.

If the language you want to contribute isn't on the list, join [our Discord server](#), and find the `#translations` channel to see if someone is already working on the language you want, consider joining them and translate together. If not, you can start a new translation project with the following steps.

In case it's already been translated but you're wondering how to maintain it, skip to the end. ## Some tips before you get started

- It is recommended that you use your IDE of choice (e.g VSCode) paired with a development server running, so you can see your translation changes in real-time.
- You can mark these checkmarks as the translation progresses or use your own workflow. The translations don't need to be made in any particular order.
- Translations don't need to be literal, but they should convey the same message. In case you're not sure how to translate something, you can either leave it as it is or use online tools like WordReference or Linguee to aid you.
- Most translations will simply consist in editing Markdown files. Certain areas are buried under Vue components, which will be listed below. You can also use your IDE to find the string to translate.

Getting started

- Fork the main docs repo: [slidevjs/docs](#)
- Translate README.md, you can take one of the already translated repositories as an example.
- Share your repo's link to the `#translations` channel telling people you are working on it and find collaborators.

Translating Markdown files

- `showcases.md` - A gallery showcase of Slidev presentations.
- `index.md` - Mainpage content, note that some of it is buried under Vue components listed further below.

.vitepress/

- `config.js` - Sitemap
- `/theme/components/WorkingInProgress.vue` - WIP notice shown in mainpage
- `/theme/components/demo/Demo.vue` - Animated demo shown in mainpage
- `/theme/components/Environment.vue` - Describes the environment of a setting.

builtin/

- `components.md` - Use [Vue components](#) inside Slidev
- `layouts.md` - Use Vue layouts inside Slidev

custom/

- `config-katex.md` - Configuring Katex
- `config-mermaid.md` - Configuring Mermaid
- `config-monaco.md` - Configuring Monaco
- `config-shortcuts.md` - Configuring Shortcuts
- `config-vite.md` - Configuring Vite
- `config-vue.md` - Configuring Vue
- `config-windicss.md` - Configuring Windicss
- `directory-structure.md` - Configuring the directory structure
- `fonts.md` - Configuring fonts
- `global-layers.md` - Configuring the global layers
- `highlighters.md` - Configuring code highlighters
- `index.md` - Customizations index page
- `vue-context.md` - The Vue global context

guide/

- `animations.md` - Animations and transitions
- `editors.md` - Editor integrations
- `exporting.md` - Exporting your slides
- `faq.md` - Frequent Answered Questions
- `index.md` - Getting started with Slidev
- `navigation.md` - Navigation across slides
- `presenter-mode.md` - Toggling presenter mode
- `recording.md` - Recording your presentation
- `syntax.md` - Markdown syntax
- `why.md` - *Why Slidev?*

resources/

- `covers.md` - Curated covers for Slidev

themes/

- `gallery.md` - Theme gallery
- `use.md` - How to use Slidev themes
- `write-a-theme.md` - Write your own theme

Publishing your translations

- When you finish the translation (at least 90%), [@antfu](#) in the Discord and we will invite you to the org and make the translation official.
- Once the transferring is done, we will set up the subdomain, auto-deployment, and a daily sync-up bot to keep the translation up-to-date with the latest English docs.
- The site is live, and we will send a shout-out tweet on [our Twitter account](#).

Maintaining the translations up-to-date

- `docschina-bot` will periodically submit merge requests from the `slidev/docs` repository. Switch to the branch created in the pull request, make any changes necessary and merge it. [example](#).
- Sometimes it will occur that a merge request is made and you haven't merged the previous one. The latest PR always checks your main branch against the English one; so you can just close the previous PR(s), move your work to the latest one and merge it.

[Working-in-progress translation list](#)

Thanks again!

[Go to TOC](#)

コンポーネント

ビルトインコンポーネント

このセクションのドキュメントはまだ作成中です。完成するまでは[ソースコード](#)を直接参照してください。

TOC

目次を挿入します。

タイトルとタイトルレベルは、各スライドの最初のタイトル要素から自動的に取得されます。

フロントマターの構文を使用することで、スライドに対するこの自動取得の動作をオーバーライドすることができます：

```
---  
title: Amazing slide title  
level: 2  
---
```

またスライドが目次に全く表示されないようにしたい場合には、次のようにします：

```
---  
hideInToc: true  
---
```

使い方

`<Toc />`

パラメータ：

- `columns (string | number, デフォルト: 1)`: 表示する列数
- `maxDepth (string | number, デフォルト: Infinity)`: 表示する目次の階層の最大値
- `minDepth (string | number, デフォルト: 1)`: 表示する目次の階層の最小値
- `mode ('all' | 'onlyCurrentTree' | 'onlySiblings', デフォルト: 'all')`:
 - `'all'` : すべての項目を表示する
 - `'onlyCurrentTree'` : 現在のページが含まれるツリーのアイテムのみを表示する（アクティブな要素、アクティブな要素の親と子）
 - `'onlySiblings'` : 現在のページが含まれるツリーと直接の兄弟関係にある要素のみを表示する

カスタムコンポーネント

プロジェクトルートに `components/` ディレクトリを作成し、カスタムVueコンポーネントを配置するだけで、Markdown ファイル内で同じ名前を使用してコンポーネントを使用することができます！

詳細は[コンポーネント](#)

テーマが提供するコンポーネント

テーマはコンポーネントを提供することもできます。提供されているものについては、それぞれのテーマのドキュメントを参照してください。

詳細は[ディレクトリ構造](#)のセクションを参照してください。

[Go to TOC](#)

レイアウト

ビルトインレイアウト

テーマはレイアウトの動作を上書きすることがあるため、使い方やパラメータ、サンプルを正しく知るには、各テーマのドキュメントを参照するのがよいでしょう。

center

コンテンツを画面中央に表示します。

cover

プレゼンテーションの表紙を表示するために使用します。プレゼンテーションのタイトルや、コンテキストを含めることができます。

default

最も基本的なレイアウトで、あらゆる種類のコンテンツを表示することができます。

end

プレゼンテーションの最後のページです。

fact

事実やデータを画面上で大きく目立たせて見せるために使用します。

full

画面のすべてのスペースを使って、コンテンツを表示します。

image-left

画面の左側に画像を表示し、右側にコンテンツを配置します。

使い方

```
---  
layout: image-left  
  
# the image source  
image: ./path/to/the/image  
  
# a custom class name to the content  
class: my-cool-content-on-the-right  
---
```

image-right

画面右側に画像を表示し、左側にコンテンツを配置します。

使い方

```
---  
layout: image-right  
  
# the image source  
image: ./path/to/the/image  
  
# a custom class name to the content  
class: my-cool-content-on-the-left  
---
```

image

画像をページのメインコンテンツとして表示します。

使い方

```
---  
layout: image  
  
# the image source  
image: ./path/to/the/image  
---
```

iframe-left

画面の左側にWebページを表示し、右側にコンテンツを配置します。

使い方

```
---  
layout: iframe-left  
  
# the web page source  
url: https://github.com/slidesjs/slides  
  
# a custom class name to the content  
class: my-cool-content-on-the-right  
---
```

iframe-right

画面の右側にWebページを表示し、左側にコンテンツを配置します。

使い方

```
---  
layout: iframe-right  
  
# the web page source  
url: https://github.com/slidesjs/slides
```

```
---  
# a custom class name to the content  
class: my-cool-content-on-the-left  
---
```

iframe

Webページをメインコンテンツとして表示します。

使い方

```
---  
layout: iframe  
  
# the web page source  
url: https://github.com/slidevjs/slidev  
---
```

intro

プレゼンテーションの始まりに使用します。一般的にはプレゼンテーションのタイトル、簡潔な説明、著者などです。

none

スタイルなしのレイアウトです。

quote

引用文を目立つように表示します。

section

新しいプレゼンテーションのセクションの開始を示すために使用します。

statement

断言/宣言をメインページのコンテンツとして表示します。

two-cols

ページのコンテンツを2列に分割します。

使い方

```
---  
layout: two-cols  
---  
  
# Left  
  
これは左側に表示されます。  
::right::
```

Right

これは右側に表示されます。

カスタムレイアウト

プロジェクトルートの配下に `layouts/` というディレクトリを作成し、そこにカスタムしたVueのレイアウトコンポーネントを配置します。

詳細は [レイアウト](#)

テーマが提供するレイアウト

テーマはレイアウトを提供したり、既存のレイアウトを上書きすることができます。テーマが提供するレイアウトについては、各テーマのドキュメントを参照してください。

[Go to TOC](#)

KaTeXの設定

以下の内容で `./setup/katex.ts` を作成します：

```
import { defineKatexSetup } from '@slidev/types'

export default defineKatexSetup(() => {
  return {
    /* ... */
  }
})
```

この設定により、[KaTeX Options](#)のカスタム設定を使用することができます。詳細については、型の定義とドキュメントを参照してください。

Mermaidの設定

以下の内容で `./setup/mermaid.ts` を作成します：

```
import { defineMermaidSetup } from '@slidev/types'

export default defineMermaidSetup(() => {
  return {
    theme: 'forest',
  }
})
```

セットアップにより、**Mermaid**のカスタムデフォルト設定を使用することができます。詳細については、型の定義とドキュメントを参照してください。

Monacoの設定

以下の内容で `./setup/monaco.ts` を作成します。

```
import { defineMonacoSetup } from '@slidev/types'

export default defineMonacoSetup(async (monaco) => {
  // 設定するためには`monaco`を使用します
})
```

詳細は[configuring Monaco](#)を参照してください。

使い方

スライドでMonacoを使用するには、コードスニペットに `{monaco}` を追加するだけです：

```
//``js
const count = ref(1)
const plusOne = computed(() => count.value + 1)

console.log(plusOne.value) // 2

plusOne.value++ // error
//``
```

これを以下のように変更します

```
//``js {monaco}
const count = ref(1)
const plusOne = computed(() => count.value + 1)

console.log(plusOne.value) // 2

plusOne.value++ // error
//``
```

エクスポート

デフォルトでは、Monacoは `開発者` モードのみで動作します。エクスポートされたSPAでMonacoを使用したい場合は、フロントマターで設定してください：

```
---
monaco: true # デフォルト "dev"
---
```

型の自動インストール

MonacoでTypeScriptを使用する場合、依存関係のある型は自動的にクライアントサイドにインストールされます。

```
//``ts {monaco}
import { ref } from 'vue'
import { useMouse } from '@vueuse/core'

const counter = ref(0)
//``
```

上記の例では、`vue`と`@vueuse/core`は`dependencies` / `devDependencies`としてローカルにインストールされています。あとはSlidevが自動的にエディタに対応した型を作成します。

テーマの設定

テーマはライトテーマ/ダークテーマをベースにSlidevで制御されています。テーマをカスタマイズしたい場合は、テーマのIDを`setup`関数に指定します：

```
// ./setup/monaco.ts
import { defineMonacoSetup } from '@slidev/types'

export default defineMonacoSetup(() => {
  return {
    theme: {
      dark: 'vs-dark',
      light: 'vs',
    },
  }
})
```

カスタムテーマを使用する場合には次のようにします：

```
import { defineMonacoSetup } from '@slidev/types'

// change to your themes
import dark from 'theme-vitesse/themes/vitesse-dark.json'
import light from 'theme-vitesse/themes/vitesse-light.json'

export default defineMonacoSetup((monaco) => {
  monaco.editor.defineTheme('vitesse-light', light as any)
  monaco.editor.defineTheme('vitesse-dark', dark as any)

  return {
    theme: {
      light: 'vitesse-light',
      dark: 'vitesse-dark',
    },
  }
})
```

Slidev用のテーマを作成する場合は、`setup`関数内で動的`import()`を使用すると、より良いtree-shakingとcode-splittingの結果を得ることができます。

ショートカットの設定

v0.20から使用可能です

以下の内容で `./setup/shortcuts.ts` を作成します：

```
import { defineShortcutsSetup, NavOperations } from '@slidev/types'

export default defineShortcutsSetup((nav: NavOperations) => {
  return [
    {
      key: 'enter',
      fn: () => nav.next(),
      autoRepeat: true,
    },
    {
      key: 'backspace',
      fn: () => nav.prev(),
      autoRepeat: true,
    },
  ],
})
```

セットアップによって、[ナビゲーション](#)

設定用の関数は、いくつかのナビゲーションメソッドを持つオブジェクトを受け取り、いくつかのショートカット設定を含む配列を返します。詳細については型定義を参照してください。

キーが押されたときのイベントについての詳細は、[useMagicKeys | VueUse](#)を参照してください。

[Go to TOC](#)

Viteの設定

SlidevはViteを搭載しています。つまり、Viteの素晴らしいプラグインシステムを利用して、スライドをさらにカスタマイズすることができます。

`vite.config.ts`があれば、それが採用されます。

Slidevには以下のプラグインがあらかじめ設定されています：

- `@vitejs/plugin-vue`
- `unplugin-vue-components`
- `unplugin-icons`
- `vite-plugin-md`
- `vite-plugin-windicss`
- `vite-plugin-remote-assets`

詳細は[pre-configurations here](#)を参照してください。

内部プラグインの設定

v0.21から使用可能です

上記のビルトインプラグインのリストを設定するために、以下の内容で `vite.config.ts` を作成します。Slidevはこれらのプラグインに対して、いくつかあらかじめ設定されたオプションを保持していることに注意してください。この使い方はそれらの設定を上書きし、アプリケーションが潜在的に壊れる原因になる可能性があります。これは**高度な機能**として扱い、何を行っているのかを確認してから次に進んでください。

```
import { defineConfig } from 'vite'

export default defineConfig({
  slidev: {
    vue: {
      /* vue options */
    },
    markdown: {
      /* markdown-it options */
      markdownItSetup(md) {
        /* custom markdown-it plugins */
        md.use(/* ... */)
      },
    },
    /* options for other plugins */
  },
})
```

その他のオプションについては[type declarations](#)を参照してください。

[Go to TOC](#)

Vueの設定

SlidevはクライアントサイドでアプリケーションをレンダリングするためにVue 3を使用しています。カスタムプラグインや設定を追加することで、アプリケーションを拡張することができます。

以下の内容で `./setup/main.ts` を作成します：

```
import { defineAppSetup } from '@slidev/types'

export default defineAppSetup(({ app, router }) => {
  // Vue App
  app.use(YourPlugin)
})
```

これはSlidevアプリのメインエントランスとしても使用することができ、アプリ起動前にいくつかの初期化を行うことができます。

詳細： [アプリケーションAPI](#)

[Go to TOC](#)

Windi CSSの設定

Markdownは当然ですが、埋め込まれたHTMLマークアップをサポートしています。したがって、好きなようにコンテンツをスタイルすることができます。いくつかの利便性を提供するために、[Windi CSS](#)を内蔵し、クラスユーティリティを使用して、直接マークアップにスタイルを設定することができます。

例：

```
<div class="grid pt-4 gap-4 grids-cols-[100px,1fr]>
  ### Name
  - Item 1
  - Item 2
</div>
```

[Windi CSS v3.0](#)のAttributify Modeはデフォルトで有効になっています。

設定

Windi CSSを設定するために、以下の内容で `setup/windicss.ts` を作成し、ビルトインの設定を拡張します。

```
// setup/windicss.ts

import { defineWindiSetup } from '@slidev/types'

// ビルトインのWindi CSSの設定を拡張する
export default defineWindiSetup(() => ({
  shortcuts: {
    // デフォルトの背景をカスタマイズします
    'bg-main': 'bg-white text-[#181818] dark:(bg-[#121212] text-[#ddd])',
  },
  theme: {
    extend: {
      // フォントはここで置き換えることができますが、`index.html`のWebフォントのリンクの更新を
      // 忘れないようにしてください
      fontFamily: {
        sans: 'ui-sans-serif,system-ui,-apple-system,BlinkMacSystemFont,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji"',
        mono: '"Fira Code", monospace',
      },
    },
  },
}))
```

詳細は[Windi CSS configurations](#)を参照してください。

[Go to TOC](#)

ディレクトリ構造

Slidevは設定面を最小化し、機能拡張を柔軟かつ直感的に行うために、いくつかのディレクトリ構造の規約を採用しています。

基本的な構造は以下の通りです：

```
your-slidev/
  └── components/      # カスタムコンポーネント
  └── layouts/         # カスタムレイアウト
  └── public/          # 静的アセット
  └── setup/           # カスタムセットアップ/フック
  └── styles/          # カスタムスタイル
  └── index.html        # index.htmlへのインジェクション
  └── slides.md         # メインスライド
  └── vite.config.ts    # 拡張されたviteの設定
```

すべてオプションです。

コンポーネント

規約： `./components/*.{vue,js,ts,jsx,tsx,md}`

このディレクトリ内のコンポーネントは、ファイル名と同じコンポーネント名で、スライドのMarkdownで直接使用することができます。

例：

```
your-slidev/
  └── ...
  └── components/
      └── MyComponent.vue
      └── HelloWorld.ts
```

```
<!-- slides.md -->

# My Slide

<MyComponent :count="4"/>

<!-- どちらの名前も使えます -->

<hello-world foo="bar">
  Slot
</hello-world>
```

この機能は `vite-plugin-components` によって提供されています。詳細はこちらを参照してください。

またSlidevはいくつかの[ビルトインコンポーネント](#)を提供していますので、それを利用することもできます。

レイアウト

規約： `./layouts/*.{vue,js,ts,jsx,tsx}`

```
your-slidev/
  └── ...
    └── layouts/
      ├── cover.vue
      └── my-cool-theme.vue
```

レイアウトには任意のファイル名を使用することができます。そしてファイル名を使用して、YAMLヘッダでレイアウトを参照します。

```
---  
layout: my-cool-theme  
---
```

作成したレイアウトがビルトインのレイアウトやテーマと同じ名前の場合、カスタムレイアウトがビルトイン/テーマレイアウトより優先されます。優先順位は `ローカル > テーマ > ビルトイン` の順です。

レイアウトコンポーネントでは、スライドのコンテンツに対して `<slot />` を使用します。例：

```
<!-- default.vue -->
<template>
  <div class="slidev-layout default">
    <slot />
  </div>
</template>
```

静的アセット

規約： `./public/*`

このディレクトリに配置されているアセットは、開発中はルートパス `/` で提供され、そのままdistディレクトリのルートにコピーされます。詳細は[Vite's public directory](#)を参照してください。

スタイル

規約： `./style.css | ./styles/index.{css,js,ts}`

この規約に従って配置されたファイルは、Appのルートに挿入されます。複数のCSSをインポートする必要がある場合は、以下のような構造を作成し、インポートの順序を自分で管理することができます。

```
your-slidev/
  └── ...
    └── styles/
      ├── index.ts
      ├── base.css
      ├── code.css
      └── layouts.css
```

```
// styles/index.ts

import './base.css'
import './code.css'
import './layouts.css'
```

スタイルはWindi CSSとPostCSSで処理されるため、CSSのネストやat-directivesをそのまま使用することができます。
例：

```
.slidev-layout {
  @apply px-14 py-10 text-[1.1rem];

  h1, h2, h3, h4, p, div {
    @apply select-none;
  }

  pre, code {
    @apply select-text;
  }

  a {
    color: theme('colors.primary');
  }
}
```

[シンタックスについて詳しく学ぶ](#)

index.html

規約：`index.html`

`index.html` はメインの `index.html` にmetaタグやscriptを挿入する機能を提供します。

例えば、次のようなカスタム `index.html` の場合：

```
<!-- ./index.html -->
<head>
  <link rel="preconnect" href="https://fonts.gstatic.com">
  <link href="https://fonts.googleapis.com/css2?family=Fira+Code:wght@400;600&family=Nunito+Sans:wght@200;400;600&display=swap" rel="stylesheet">
</head>

<body>
  <script src="./your-scripts"></script>
</body>
```

最終的にホストされる `index.html` は次のようになります。

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="icon" type="image/png" href="https://cdn.jsdelivr.net/gh/slidevjs/slidev/assets/favicon.png">
  <!-- 挿入されたhead -->
```

```
<link rel="preconnect" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css2?
family=Fira+Code:wght@400;600&family=Nunito+Sans:wght@200;400;600&display=swap"
rel="stylesheet">
</head>
<body>
  <div id="app"></div>
  <script type="module" src="__ENTRY__"></script>
  <!-- 挿入されたbody -->
  <script src="./your-scripts"></script>
</body>
</html>
```

グローバルレイヤー

規約 : [global-top.vue](#) | [global-bottom.vue](#)

詳細 : [グローバルレイヤー](#)

[Go to TOC](#)

フォント

v0.20から使用可能です

HTMLやCSSを使ってスライドのフォントやスタイルを自由にカスタマイズすることができます、Slidevはそれらを楽に扱うための便利な方法も提供しています。

フロントマターで、以下のように設定します。

```
---  
  fonts:  
    # basically the text  
    sans: 'Robot'  
    # use with `font-serif` css class from windicss  
    serif: 'Robot Slab'  
    # for code blocks, inline code, etc.  
    mono: 'Fira Code'  
---
```

フォントは [Google Fonts](#)から自動的にインポートされます。つまり、Google Fontsで利用可能なフォントを直接利用することができます。

ローカルフォント

デフォルトでは、Slidevは `fonts` で指定されたすべてのフォントをGoogle Fontsのものとして認識します。ローカルのフォントを使用したい場合は、`fonts.local` を指定して、自動インポートを無効にします。

```
---  
  fonts:  
    # CSSにおけるfont-familyの指定のように、` `を使用することでフォールバックのためのフォントを複数指定できます。  
    sans: 'Helvetica Neue, Robot'  
    # 'Helvetica Neue'をローカルフォントとして指定します。  
    local: 'Helvetica Neue'  
---
```

太さと斜体

デフォルトでは、Slidevは各フォントに対して、`200`、`400`、`600` の3つの太さをインポートします。次のように指定することができます：

```
---  
  fonts:  
    sans: 'Robot'  
    # デフォルト  
    weights: '200,400,600'  
    # デフォルトで`false`になっている斜体フォントをインポートします。  
    italic: false  
---
```

この設定はすべてのWebフォントに適用されます。各フォントの太さをより細かく制御するには、[HTML](#)

フォントのフォールバック

ほとんどのケースでは、"特別なフォント"を指定するだけで、Slidevがフォントフォールバックを追加してくれます。例：

```
---  
fonts:  
  sans: 'Robot'  
  serif: 'Robot Slab'  
  mono: 'Fira Code'  
---
```

これは次のようにになります

```
.font-sans {  
  font-family: "Robot", ui-sans-serif, system-ui, -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, "Noto Sans", sans-serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color Emoji";  
}  
.font-serif {  
  font-family: "Robot Slab", ui-serif, Georgia, Cambria, "Times New Roman", Times, serif;  
}  
.font-mono {  
  font-family: "Fira Code", ui-monospace, SFMono-Regular, Menlo, Monaco, Consolas, "Liberation Mono", "Courier New", monospace;  
}
```

フォントフォールバックを無効化するには、次のように設定します。

```
---  
fonts:  
  mono: 'Fira Code, monospace'  
  fallback: false  
---
```

プロバイダー

- オプション：`google` | `none`
- デフォルト：`google`

現時点ではGoogle Fontsのみをサポートしていますが、将来的にはほかのプロバイダーも追加する予定です。`none` を指定すると、自動インポート機能を完全に無効にし、すべてのフォントをローカルに扱えるようになります。

```
---  
fonts:  
  provider: 'none'  
---
```

[Go to TOC](#)

グローバルレイヤー

v0.17から使用可能です

グローバルレイヤーを使用すると、スライド間で永続的なカスタムコンポーネントを持つことができます。これはフッター、スライドをまたぐアニメーション、グローバルエフェクトなどに便利です。

Slidevはこのために2つのレイヤーを提供しています。プロジェクトのルートに `global-top.vue` や `global-bottom.vue` を作成すると、自動的にピックアップされます。

レイヤーの関係性：

- グローバルトップ (`global-top.vue`)
- スライド
- グローバルボトム (`global-bottom.vue`)

例

```
<!-- global-top.vue -->
<template>
  <footer class="absolute bottom-0 left-0 right-0 p-2">Your Name</footer>
</template>
```

`Your Name` という文字はすべてのスライドに表示されます。

条件付きで有効にするには、[Vueグローバルコンテキスト](#)を使用して適用します。

```
<!-- Page 4からフッターを隠します-->
<template>
  <footer
    v-if="$slidev.nav.currentPage !== 4"
    class="absolute bottom-0 left-0 right-0 p-2">
    Your Name
  </footer>
</template>
```

```
<!-- "cover"レイアウトからフッターを隠します -->
<template>
  <footer
    v-if="$slidev.nav.currentLayout !== 'cover'"
    class="absolute bottom-0 left-0 right-0 p-2">
    Your Name
  </footer>
</template>
```

```
<!-- ページ用フッターの一例 -->
<template>
  <footer>
```

```
v-if="$slidev.nav.currentLayout !== 'cover'"  
  class="absolute bottom-0 left-0 right-0 p-2"  
>    {{ $slidev.nav.currentPage }} / {{ $slidev.nav.total }}  
  </footer>  
</template>
```

シンタックスハイライト

Slidevには2種類のシンタックスハイライトが付属しており、好みに合わせて選択できます：

- [Prism](#)
- [Shiki](#)

Prismは最も人気のあるシンタックスハイライトの1つです。ハイライトはトークンクラスにコードを追加することで行われ、CSSによって色付けされます。[公式テーマ](#)を確認したり、`prism-theme-vars`を使用して簡単にテーマを作成/カスタマイズすることができます。

一方で**Shiki**はTextMateの文法に対応したシンタックスハイライトです。色付きのトークンを生成するため、追加のCSSは必要ありません。充実した文法サポートにより、生成された色はVS Codeで見ると同じように非常に正確です。またShikiには[多くのビルトインテーマ](#)が付属しています。Shikiの欠点は、ハイライトを行うためにTextMateのテーマ（VS Codeのテーマと互換性あり）も必要で、カスタマイズが少し難しくなることです。

Slidevのテーマは通常PrismとShikiの双方をサポートしていますが、使用するテーマによっては、どちらか一方しかサポートしていない場合があります。

選択肢がある場合、基本的にトレードオフです：

- **Prism**はカスタマイズが容易です。
- **Shiki**はより正確にハイライトできます。

デフォルトでは、SlidevはPrismを使用します。フロントマターを修正することで、この設定を変更できます：

```
---  
highlighter: shiki  
---
```

Prismの設定

Prismを設定する場合には、テーマのCSSをインポートするか、`prism-theme-vars`を使用してテーマを設定することで、ライトモードとダークモードの両方のテーマを設定することができます。詳しくはドキュメントを参照してください。

Shikiの設定

以下の内容で `./setup/shiki.ts` ファイルを作成します。

```
/* ./setup/shiki.ts */
import { defineShikiSetup } from '@slidev/types'

export default defineShikiSetup(() => {
  return {
    theme: {
      dark: 'min-dark',
      light: 'min-light',
    },
  }
})
```

利用可能なテーマの名前については、[Shikiのドキュメント](#)を参照してください。

自身のテーマを使用したい場合は次のようにします：

```
/* ./setup/shiki.ts */

import { defineShikiSetup } from '@slidev/types'

export default defineShikiSetup(async({ loadTheme }) => {
  return {
    theme: {
      dark: await loadTheme('path/to/theme.json'),
      light: await loadTheme('path/to/theme.json'),
    },
  }
})
```

カスタマイズ

Slidevは、スタイル링からツールの設定まで、フルカスタマイズが可能です。Slidevの配下にあるツール（[Vite](#)、[Windicss](#)、[Monaco](#)など）を設定することができます。

フロントマターの設定

最初のスライドのフロントマターで、Slidevの設定をすることができます。以下に各オプションのデフォルト値を示しています。

```
---  
# テーマのIDもしくはパッケージ名  
theme: 'default'  
# スライドのタイトル 指定されていない場合、最初のヘッダーから自動的に推測されます。  
title: ''  
# webページのタイトルテンプレート `%-s`は各ページのタイトルで置き換えられます。  
titleTemplate: '%s - Slidev'  
  
# SPAビルドにおけるPDFダウンロードを有効化します。カスタムURLを使用することも可能です。  
download: true  
# シンタックスハイライト 'prism'か'shiki'が選択可能です。  
highlighter: 'prism'  
# コードブロックに行番号を表示します。  
lineNumbers: false  
# Monacoエディタを有効化します。デフォルトでは開発環境のみ有効です。  
monaco: 'dev'  
  
# スライドのカラースキーマを変更します。'auto'、'light'または'dark'を指定可能です。  
colorSchema: 'auto'  
# vue-routerのためのrouterModeを指定します。"history"または"hash"が指定可能です。  
routerMode: 'history'  
# スライドのアスペクト比を指定します。  
aspectRatio: '16/9'  
# canvasの実際の横幅を指定します。単位はpxです。  
canvasWidth: 980  
  
# faviconにはローカルファイルのパスか、URLを使用できます。  
favicon: 'https://cdn.jsdelivr.net/gh/slidevjs/slidev/assets/favicon.png'  
# フォントはGoogle Fontsから自動的にimportされます。  
# 詳細: https://sli.dev/custom/fonts  
fonts:  
  sans: 'Roboto'  
  serif: 'Roboto Slab'  
  mono: 'Fira Code'  
  
# デフォルトのフロントマターはすべてのスライドに適用されます。  
defaults:  
  layout: 'default'  
  # ...  
  
# スライドの情報をMarkdownで記述することができます。  
info: |  
  ## Slidev  
  My first [Slidev](http://sli.dev/) presentations!  
---
```

より詳しいオプションは[type definitions](#)を参照してください。

ディレクトリ構造

Slidevはディレクトリ構造の規約を利用して、設定を最小化し、機能の拡張を柔軟かつ直感的に行えるようにしています。

[ディレクトリ構造](#)のセクションを参照してください。

ツールを設定する

- シンタックスハイライト
- [Vueの設定](#)
- [Viteの設定](#)
- [Windi CSSの設定](#)
- [Monacoの設定](#)
- [KaTeXの設定](#)
- [Mermaidの設定](#)

[Go to TOC](#)

Vueグローバルコンテキスト

Slidevは高度な条件やナビゲーションのコントロールのために、[グローバルVueコンテキスト](#) `$slidev` を注入しています。

使い方

MarkdownやVueテンプレートのどこでも、["Mustache"タグ](#)を使ってアクセスできます。

```
<!-- slides.md -->
# Page 1
Current page is: {{ $slidev.nav.currentPage }}
```

```
<!-- Foo.vue -->
<template>
  <div>Title: {{ $slidev.configs.title }}</div>
  <button @click="$slidev.nav.next">Next Page</button>
</template>
```

プロパティ

`$slidev.nav`

スライドナビゲーションのプロパティとコントロールを保持するリアクティブオブジェクトです。例：

```
$slidev.nav.next() // go next step
$slidev.nav.nextSlide() // go next slide (skip v-clicks)
$slidev.nav.go(10) // go slide #10
```

```
$slidev.nav.currentPage // current slide number
$slidev.nav.currentLayout // current layout id
$slidev.nav.clicks // current clicks count
```

利用できるプロパティの詳細については、[nav.ts](#)のエクスポートを参照してください。

`$slidev.configs`

`slides.md` のフロントマターの設定 例：

```
---
title: My First Slidev!
---
```

```
{{ $slidev.configs.title }} // 'My First Slidev!'
```

\$slidev.themeConfigs

テーマの設定をパースしたものを保持するリアクティブオブジェクトです。

```
---  
title: My First Slidev!  
themeConfig:  
  primary: #213435  
---
```

```
{} $slidev.themeConfigs.primary }} // '#213435'
```

アニメーション

クリックアニメーション

v-click

要素に対して"クリックアニメーション"を適用するには、`v-click` ディレクティブか `<v-click>` コンポーネントを使用することができます。

```
# Hello

<! -- コンポーネントの使い方: "次へ"を押すまで、ここから下の内容は表示されません -->
<v-click>

Hello World

</v-click>

<! -- ディレクティブの使い方: 2回目の"次へ"を押すまで、ここから下の内容は表示されません -->
<div v-click class="text-xl p-2">

Hey!

</div>
```

v-after

`v-after` は `v-click` に似ていますが、直前の `v-click` がトリガーされたときに要素を可視化します。

```
<div v-click>Hello</div>
<div v-after>World</div>
```

"次へ"ボタンを押した時に、`Hello` と `World` の両方が一緒に表示されます。

v-click-hide

`v-click` 同じですが、要素を表示するのではなく、クリックした後に要素を非表示にします。

```
<div v-click-hide>Hello</div>
```

v-clicks

`v-clicks` はコンポーネントとしてのみ提供されています。これは `v-click` ディレクティブをそのすべての子要素に適用するためのショートハンドです。特にリストを扱う場合に便利です。

```
<v-clicks>
  - Item 1
  - Item 2
  - Item 3
```

- Item 4

```
</v-clicks>
```

"次へ"をクリックするたびに、項目が表示されるようになります。

カスタムクリックカウント

デフォルトでは、Slidevは次のスライドに進む前に必要なステップ数をカウントします。 `clicks` というフロントマターオプションを記述することで、この設定をオーバーライドできます。

```
---  
# このスライドでは、次のスライドに進むまでに10回クリックする  
clicks: 10  
---
```

並び替え

ディレクティブにクリックインデックスを渡すことで、公開する順番をカスタマイズすることができます。

```
<div v-click>1</div>  
<div v-click>2</div>  
<div v-click>3</div>
```

```
<!-- 順番が逆転する -->  
<div v-click="3">1</div>  
<div v-click="2">2</div>  
<div v-click="1">3</div>
```

```
---  
clicks: 3  
---  
  
<!-- 3回クリックした後に見えるようになる -->  
<v-clicks at="3">  
  <div>Hi</div>  
</v-clicks>
```

要素のトランジション

要素に `v-click` ディレクティブを適用すると、`slidev-vclick-target` というクラス名が付与されます。要素が非表示になった場合、クラス名 `slidev-vclick-hidden` が付与されます。例：

```
<div class="slidev-vclick-target slidev-vclick-hidden">テキスト</div>
```

クリックすると、以下のようになります

```
<div class="slidev-vclick-target">テキスト</div>
```

デフォルトでは、これらのクラスにはわずかな透明度のトランジションが適用されます。

```
// デフォルト  
.slidev-vclick-target {  
  transition: opacity 100ms ease;
```

```

}
.slidev-vclick-hidden {
  opacity: 0;
  pointer-events: none;
}

```

トランジション効果をカスタマイズするために、カスタムスタイルシートでそれらをオーバーライドすることができます。

例えば、スケールアップのトランジションは次のようにして実現することができます：

```

// styles.css

.slidev-vclick-target {
  transition: all 500ms ease;
}

.slidev-vclick-hidden {
  transform: scale(0);
}

```

特定のスライドもしくはレイアウトにのみアニメーションを適用する場合

```

.slidev-page-7,
.slidev-layout.my-custom-layout {
  .slidev-vclick-target {
    transition: all 500ms ease;
  }

  .slidev-vclick-hidden {
    transform: scale(0);
  }
}

```

詳細は[スタイルのカスタマイズ](#)

モーション

Slidevは[@vueuse/motion](#)を内蔵しています。任意の要素にモーションを適用するために `v-motion` ディレクティブを使用することができます。例：

```

<div
  v-motion
  :initial="{ x: -80 }"
  :enter="{ x: 0 }">
  Slidev
</div>

```

`Slidev` というテキストは初期化時に `-80px` から元の位置へ移動します。

注: Slidevはパフォーマンスのために次のスライドをプリロードします、つまり、ページに遷移する前にアニメーションが始まる可能性があります。正しく動作させるために、特定のスライドに対してプリロードを無効にすることができます。

```
---  
preload: false  
---
```

もしくは要素のライフサイクルを `v-if` で制御することで、きめ細やかな制御を行うこともできます。

```
<div  
  v-if="$slidev.nav.currentPage === 7"  
  v-motion  
  :initial="{ x: -80 }"  
  :enter="{ x: 0 }">  
  Slidev  
</div>
```

詳細：[デモ](#) | [@vueuse/motion](#) | [v-motion](#) | [Presets](#)

ページのトランジション

現在のバージョンでは、スライドのビルトインサポートはまだ提供されていません。次のメジャー・バージョンでサポートする予定です。それまでは、カスタムスタイルやライブラリを使ってスライドを作成することができます。

[Go to TOC](#)

描画と注釈

v0.23から使用可能です

描画や注釈を行うためのdrauuを内蔵しており、プレゼンテーションをさらに充実させることができます。

ツールバーのアイコンをクリックして、描画を開始します。 プrezenterモードでも使用可能です。作成した描画や注釈は、全インスタンスでリアルタイムに同期されます。

スタイラスペンとともに使用する

タブレットでスタイラスペンを使用する場合（例えば、iPadとApple Pencil）、Slidevは入力をスマートに検出することができます。描画モードをオンにすることなく、指やマウスでナビゲーションをコントロールしながら、ペンで直接スライドに描画することができます。

描画を保存する

以下のフロントマターの設定により、描画した内容はSVGとして`.slidev/drawings`ディレクトリ配下に保存され、エクスポートしたPDFやホスティングしたサイト内に表示させることができます。

```
---  
drawings:  
  persist: true  
---
```

描画を非表示にする

全体に対して：

```
---  
drawings:  
  enabled: false  
---
```

開発環境でのみ：

```
---  
drawings:  
  enabled: dev  
---
```

プレゼンテーションモードでのみ：

```
---  
drawings:  
  presenterOnly: true  
---
```

描画を同期させる

デフォルトでは、Slidevはすべてのインスタンスで描画を同期します。もし他の誰かとスライドを共有している場合は、同期を無効にした方がいいかもしれません：

```
---  
drawings:  
  syncAll: false  
---
```

この設定により、プレゼンターのインスタンスからの描画のみ、他のインスタンスと同期することができるようになります。

エディタサポート

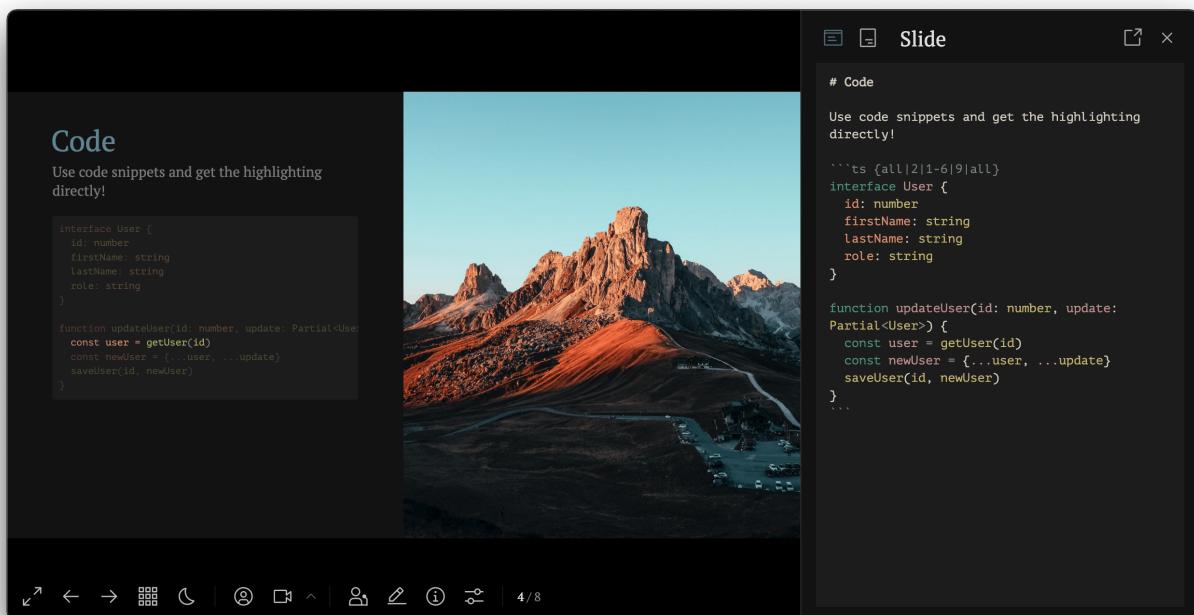
Slidevはソースの入力にMarkdownを使用しているので、あなたが好きなエディタで書くことができます。

スライドを高度に管理したい場合、以下のエディタインテグレーションを使用することができます。

統合エディタ

SlidevにはCodeMirrorというエディタが統合されており、ファイルの変更を即座に再読み込みして保存してくれます。

起動するためには ボタンを押してください。



VS Code拡張機能



Slidev for VS Code

VS Code Marketplace v0.4.1

downloads 26k

VS Code拡張機能はスライドをよりよく整理し、その概要を素早く把握するための機能を提供します。

機能

- スライドをサイドパネルで確認
- 次へ / 戻るボタン
- スライドの順序入れ替え
- スライドブロックのための折り畳み機能
- MarkdownをHTMLに変換

The screenshot shows a code editor window titled "slides.md — demo". The left sidebar contains a tree view of files under "SLIDEV: SLIDES", including "Composable Vue", "Anthony Fu", "Sponsors", "Composable Vue", "VueUse", "Composition API", "Ref", "Ref Auto Unwrapping", "unref - Oppsite of Ref", "Patterns & Tips", "What's Composable Func...", "Think as "Connections"" (highlighted), "One Thing at a Time", "Passing Refs as Argumen...", "MaybeRef <MarkerTips/>", "Make it Flexible <MarkerP...", "'useTitle' <Marker class=...>", "'Reuse' Ref <MarkerCore...>", "'ref' / 'unref' <MarkerTi...>", "Object of Refs <MarkerPa...>", "Async to "Sync" <Marker...>", "'useFetch' <Marker clas...>", and "Side-effects Self Cleanup...". The main editor area displays the following code:

```
168 # Ref Auto Unwrapping <MarkerCore />
170
    Get rid of `value` for most of the time.
171
        <div class="grid grid-cols-2 gap-x-4">
172            <v-clicks :every='2'>
173                - `watch` accepts ref as the watch target, and returns the
174                    unwrapped value in the callback
175
176                ````ts
177
178        const counter = ref(0)
179
180        watch(counter, count => {
181            console.log(count) // same as `counter.value`
182        })
183
184        - Ref is auto unwrapped in the template
185
186        ````html
187
188        <template>
189            <button @click="counter += 1">
```

[Go to TOC](#)

エクスポート

PDF

PDFもしくはPNGへのエクスポートはレンダリングのためにPlaywrightに依存しています。したがって、この機能を使用するためには `playwright-chromium` をインストールする必要があります。CI環境でエクスポートを行う場合は、[the playwright CI guide](#)が参考になります。

`playwright-chromium` のインストール

```
$ npm i -D playwright-chromium
```

次のコマンドを使用してスライドをPDFにエクスポートします。

```
$ slidev export
```

しばらくすると、スライドが `./slides-exports.pdf` に出力されます。

クリックステップをエクスポートする

v0.21から使用可能です

デフォルトでは、Slidevはクリックアニメーションを無効にして、スライド単位で1ページをエクスポートします。複数のステップがあるスライドを複数のページにエクスポートしたい場合は、`--with-clicks` オプションを指定します。

```
$ slidev export --with-clicks
```

PNGs

`--format png` オプションを指定した場合、Slidevは各スライドをPDFの代わりにPNG画像として出力します。

```
$ slidev export --format png
```

シングルページアプリケーション (SPA)

[静的ホスティング](#)を参照してください。

[Go to TOC](#)

FAQ

グリッド

SlidevはWebをベースにしているので、グリッドレイアウトを自由に適用することができます。[CSS Grids](#)、[flexboxes](#)、あるいは[Masonry](#)で制御することができます。

また[Windi CSS](#)が内蔵されているので、参考までにそれを使った簡単な方法を紹介します：

```
<div class="grid grid-cols-2 gap-4">
<div>

最初のカラム

</div>
<div>

2番目のカラム

</div>
</div>
```

さらに、各カラムのサイズをカスタマイズすることも可能です。：

```
<div class="grid grid-cols-[200px,1fr,10%] gap-4">
<div>

最初のカラム (200px)

</div>
<div>

2番目のカラム (自動調整)

</div>
<div>

3番目のカラム (親のコンテナに対して10%)

</div>
</div>
```

詳細は[Windi CSS Grids](#)を参照してください。

ポジショニング

スライドは固定サイズ（デフォルトは 980x552px）で定義され、ユーザーのスクリーンに合わせて拡大・縮小されます。画面に合わせて拡大・縮小されるため、absolute positionを使用しても安全です。

例：

```
<div class="absolute left-30px bottom-30px">
  これは左下寄せのフッターです
</div>
```

キャンバスの実際の大きさを変更するには、最初のフロントマターで `canvasWidth` オプションを指定します。

```
---  
  canvasWidth: 800  
---
```

フォントサイズ

スライドのフォントサイズが小さすぎると感じる場合、いくつかの方法で調整することができます：

ローカルスタイルをオーバーライドする

インライン `<style>` タグで、各スライドごとのスタイルをオーバーライドすることができます。

```
# Page 1
<style>
h1 {
  font-size: 10em;
}
</style>
---

# Page 2
ここには適用されません
```

詳細：[埋め込みスタイル](#)

グローバルスタイルをオーバーライドする

カスタムグローバルスタイルを定義するには、次のように `./style.css` を作成します。

```
/* style.css */
h1 {
  font-size: 10em !important;
}
```

詳細：[グローバルスタイル](#)

キャンバスの拡大・縮小

キャンバスの実寸を変更すると、すべてのコンテンツ（テキスト、画像、コンポーネントなど）とスライドが拡大・縮小されます。

```
---  
# default: 980  
# キャンバスが小さくなれば、視覚的なサイズは大きくなります。  
canvasWidth: 800  
---
```

Transformの使用

CSSのtransformプロパティの薄いラッパーである、ビルトインコンポーネント `<Transform />` を提供しています。

```
<Transform :scale="1.4">  
- Item 1  
- Item 2  
</Transform>
```

[Go to TOC](#)

静的ホスティング

シングルページアプリケーション(SPA)を構築する

スライドをセルフホスティング可能なSPAとして構築することができます：

```
$ slidev build
```

生成されたアプリケーションは `dist/` の配下に配置され、[GitHub Pages](#)、[Netlify](#)、[Vercel](#)など、好きな場所にホストすることができます。これでリンク1つで世界中のひととスライドを共有することができます。

ベースパス

サブルート下にスライドをデプロイするには、`--base` オプションを指定する必要があります。例：

```
$ slidev build --base /talks/my-cool-talk/
```

詳細は[Viteのドキュメント](#)を参照してください。

ダウンロード可能なPDFを提供する

以下の設定により、SPAの閲覧者向けにダウンロード可能なPDFを提供することができます：

```
---  
download: true  
---
```

Slidevはビルトと一緒にPDFファイルを生成し、SPAにダウンロードボタンが表示されます。

またPDFに対してカスタムURLを指定することもできます。その場合、レンダリング処理はスキップされます。

```
---  
download: 'https://myside.com/my-talk.pdf'  
---
```

サンプル

以下はSPAとしてエクスポートされた例です：

- [Starter Template](#)
- [Composable Vue by Anthony Fu](#)

詳しくは[ショーケース](#)を参照してください。

ホスティング

`npm init slidev@lastest` を使って、サービスをそのままホスティングするために必要な設定ファイルが含まれたプロジェクトの雛形を生成することを推奨します。

Netlify

- [Netlify](#)

プロジェクトルートに以下の内容で `netlify.toml` を作成します。

```
[build.environment]
  NODE_VERSION = "14"

[build]
  publish = "dist"
  command = "npm run build"

[[redirects]]
  from = "/"
  to = "/index.html"
  status = 200
```

Netlifyのダッシュボードを開き、リポジトリを指定して新しいサイトを作成してください。

Vercel

- [Vercel](#)

プロジェクトルートに以下の内容で `vercel.json` を作成します。

```
{
  "rewrites": [
    { "source": "/(.*)", "destination": "/index.html" }
  ]
}
```

Vercelのダッシュボードを開き、リポジトリを指定して新しいサイトを作成してください。

GitHub Pages

- [GitHub Pages](#)

GitHub Actions を使用して GitHub Pages にスライドをデプロイするために、以下の内容で `.github/workflows/deploy.yml` を作成してください。

```
name: Deploy pages
on: push
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - uses: actions/setup-node@v2
        with:
          node-version: '14'
      - name: Install dependencies
        run: npm install
      - name: Build
        run: npm run build
      - name: Deploy pages
```

```
uses: crazy-max/ghaction-github-pages@v2
with:
  build_dir: dist
env:
  GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

はじめに

概要

Slidev (slide + dev, `/slайдв/`) はWebベースのスライド作成およびプレゼンテーションツールです。開発者がMarkdownでコンテンツを書くことに集中しつつ、HTMLとVueコンポーネントを用いて、プレゼンテーションにインタラクティブなデモを埋め込んだピクセルパーフェクトなレイアウトとデザインを提供できるようにも設計されています。

機能豊富なマークダウンファイルを使用して、ライブコーディング、PDFエクスポート、プレゼンテーションのレコードティングのような、多くのビルトインインテグレーションとともに、瞬時に再読み込みが可能な美しいスライドを生成します。webで動くので、Slidevを使ってどんなことでもできます - 可能性は無限大です。

プロジェクトの論理的根拠については [なぜSlidev?](#) のセクションで詳しく説明しています。

機能

- 📝 **Markdownベース** - お気に入りのエディタとワークフローを使用
- 💻 **デベロッパフレンドリー** - ビルトインのシンタックスハイライト、ライブコーディングなど
- 🎨 **豊富なテーマ** - テーマはnpmパッケージで共有・利用が可能
- 🌈 **スタイリッシュ** - Windi CSS オンデマンドユーティリティ、使いやすい埋め込まれたスタイルシート
- 🧙‍♂️ **インタラクティブ** - Vueコンポーネントをシームレスに埋め込み
- 🖨 **プレゼンターモード** - 別のウィンドウ、スマートフォンでさえもスライドを操作
- 🖌 **描画** - スライドに描画し、注釈をつける
- TeX **LaTeX** - LaTeX数式のビルトインサポート
- 📈 **図形** - 説明と合わせて図形を作成
- 🌟 **アイコン** - どんなアイコンセットからでも、直接アイコンにアクセス
- 💻 **エディタ** - 統合されたエディタとVS Code拡張機能
- 🎥 **レコード** - ビルトインのレコードリングとカメラビュー
- 🖨 **ポータブル** - PDF、PNG、またはホスト可能なSPAにエクスポート
- ⚡ **高速** - Vite によって提供されたインスタントリロード
- 🛠 **自由に開発可能** - Viteプラグイン、Vue components、どんなnpmパッケージも使用可能

技術スタック

これらのツールや技術を組み合わせることで、Slidevは実現されています。

- Vite - 非常に高速なフロントエンドツール
- Vue 3をベースにしたMarkdown - 必要に応じてHTMLとVueコンポーネントを使いつつ、コンテンツに集中できます
- Windi CSS - オンデマンドなユーティリティファーストのCSSフレームワーク、スライドを自在にスタイルリング
- Prism、Shiki、Monaco Editor - ファーストクラスのコードスニペットサポートとライブコーディング機能
- RecordRTC - ビルトインのレコードリングとカメラビュー
- VueUseファミリー - `@vueuse/core`、`@vueuse/head`、`@vueuse/motion` など
- Iconify - アイコンセットコレクション
- Drauu - 描画と注釈のサポート
- KaTeX - LaTeX数式のレンダリング

- [Mermaid](#) - テキストによる図解

はじめてのプレゼンテーションを作成する

オンラインで試す

sli.dev/new



[Open in StackBlitz](#)

ローカルで作成する

NPMで作成：

```
$ npm init slidev
```

Yarnで作成：

```
$ yarn create slidev
```

プロンプトに従って、今すぐスライドを作り始めましょう！ Markdownシンタックスの詳細は、 [シンタックスガイド](#) を参照してください。

コマンドラインインターフェース

Slidevがインストールされたプロジェクトでは、 npmスクリプトで `slidev` コマンドを使用することができます。

```
{
  "scripts": {
    "dev": "slidev", // start dev server
    "build": "slidev build", // build for production SPA
    "export": "slidev export" // export slides to pdf
  }
}
```

あるいは `npx` で使用することができます。

```
$ npx slidev
```

その他のオプションについては、 `slidev --help` を実行してください。

Markdownシンタックス

Slidevはプロジェクトルートの配下にある `slides.md` を読み取り、スライドに変換します。 変更を加えると、 スライドのコンテンツに即時に反映されます。 例：

```
# Slidev
```

```
Hello World
```

```
---  
# Page 2  
  
Directly use code blocks for highlighting  
  
//```ts  
console.log('Hello, World!')  
//```  
  
---  
# Page 3
```

SlidevのMarkdownシンタックスについては [シンタックスガイド](#) を参照してください。

[Go to TOC](#)

インストール

スターターテンプレート

SlidevはNode.js >=14.0で動作します

まずは公式スターターテンプレートを利用してみてください。

NPMで使用する：

```
$ npm init slidev@latest
```

Yarnで使用する：

```
$ yarn create slidev
```

表示されるプロンプトに従って操作をすると、スライドショーが http://localhost:3030/ で自動的に立ち上がります。

スターターテンプレートには基本的な設定やSlidevの使い方を説明した簡単なデモも収録されています。

マニュアルインストール

Slidevを手動でインストールしたい、または既存のプロジェクトに統合したい場合は、次のようにします：

```
$ npm install @slidev/cli @slidev/theme-default
```

```
$ touch slides.md
```

```
$ npx slidev
```

pnpmを使用している場合、Slidevを正しく動作させるためにshamefully-hoistオプションを有効にする必要があることに注意してください。

```
$ echo 'shamefully-hoist=true' >> .npmrc
```

グローバルインストール

v0.14から使用可能です

以下のコマンドで、Slidevをグローバルにインストール可能です。

```
$ npm i -g @slidev/cli
```

毎回プロジェクトを作成することなく、どこでも `slidev` コマンドを使用できるようになります。

```
$ slidev
```

このコマンドはローカルの `@slidev/cli` が `node_modules` にあれば、それを実行します。

Docker上にインストールする

コンテナでプレゼンテーションを迅速に実行する必要がある場合、あらかじめ組み込まれている `docker` イメージ（メンテナー：[stig124](#)）を使うか、あるいは自分でビルドします。

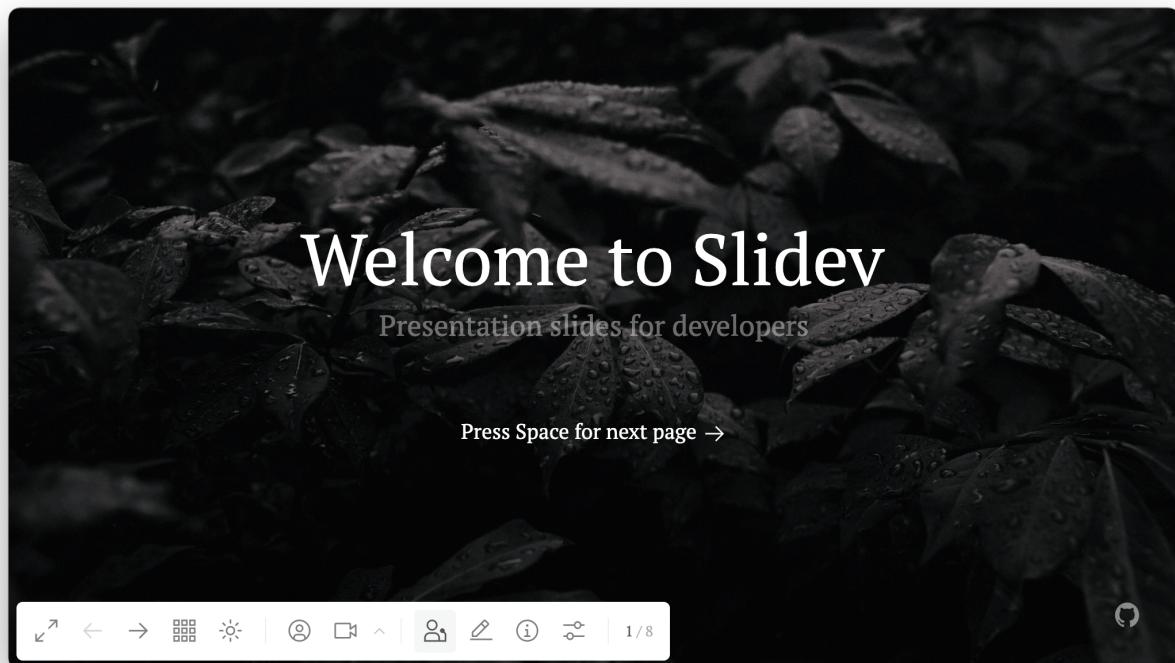
詳しくは [slidevjs/container repo](#) を参照してください。

[Go to TOC](#)

ナビゲーション

ナビゲーションバー

Slidevのページの左下にマウスを移動すると、ナビゲーションバーが表示されます。

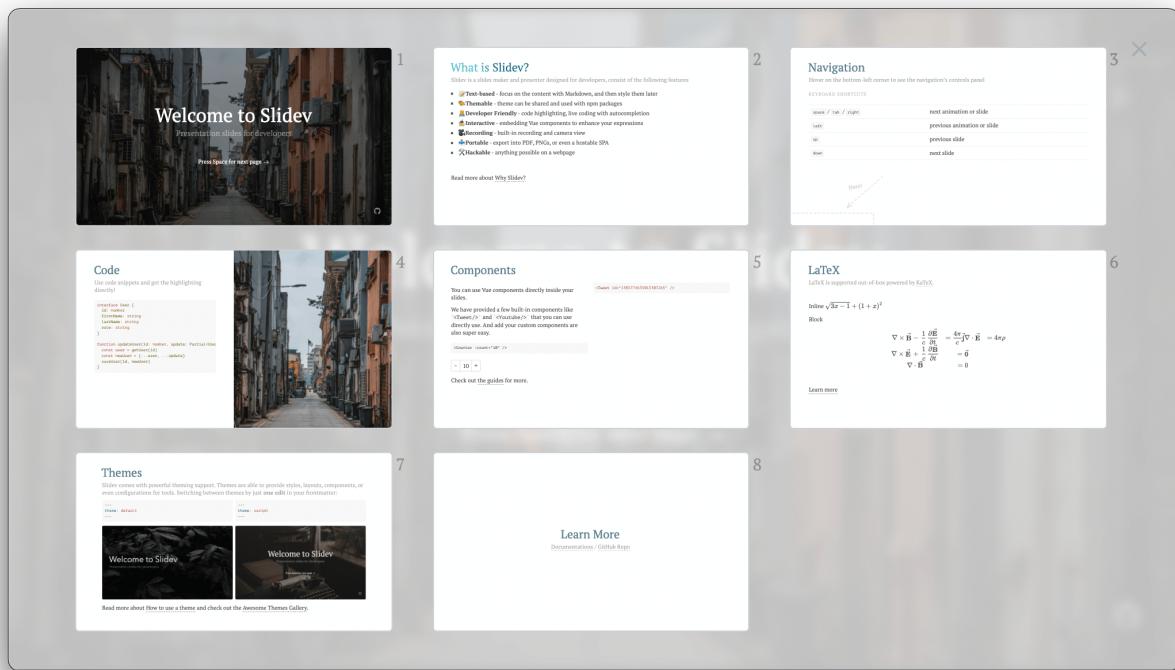


ショートカット	ボタン	説明
f		フルスクリーンに切り替え
right / space		次のアニメーションもしくはスライドへ
left		前のアニメーションもしくはスライドへ
up	-	前のスライドへ
down	-	次のスライドへ
o		スライドオーバービューに切り替え
d		ダークモードに切り替え
-		カメラビュー
-		レコーディング
-		プレゼンター モード

ショートカット ボタン	説明
-	統合エディタ
-	スライドをダウンロード (SPAビルト)
-	スライドの情報を表示
-	設定メニューを表示
g	- goto...を表示

スライドオーバービュー

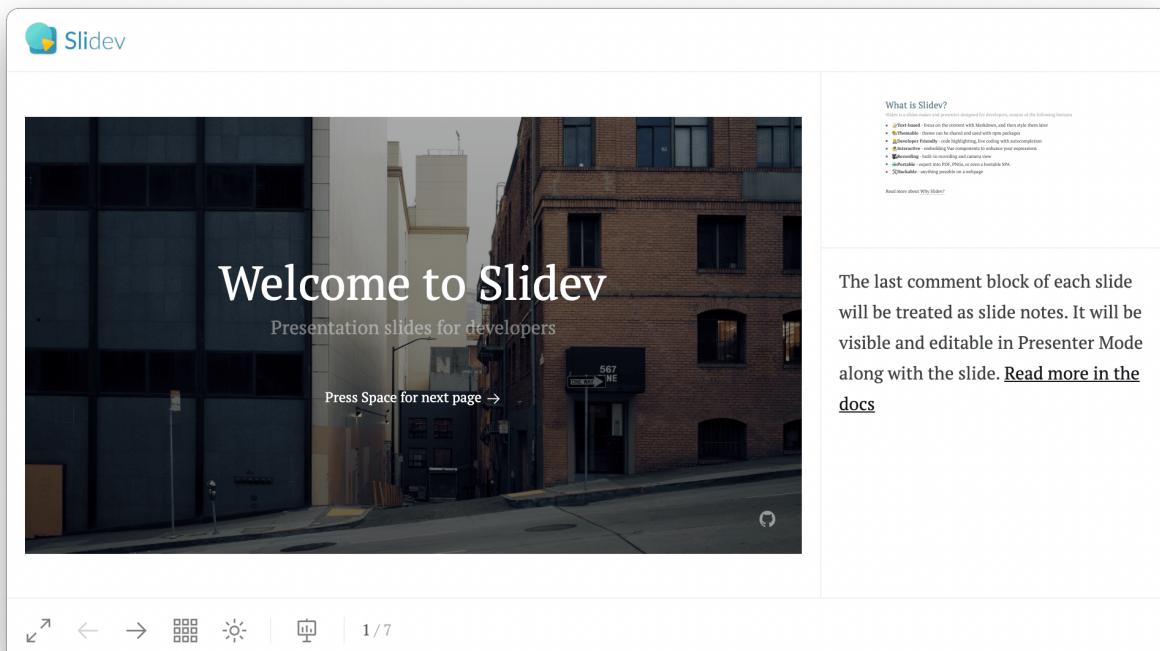
oを押すか、ナビゲーションバーのボタンをクリックすることで、スライドを俯瞰することができ、スライド間を簡単にジャンプできます。



[Go to TOC](#)

プレゼンター モード

ナビゲーションパネルの ボタンをクリックするか、 <http://localhost:3030/presenter> に手動でアクセスすることで、 プrezenter モードに入ります。プレゼンター モードに入ると、他のページインスタンスも自動的にプレゼンターと同期するようになります。



[Go to TOC](#)

レコーディング

Slidevにはレコーディング機能とカメラビューが内蔵されています。これらを使って一箇所で簡単にプレゼンテーションをレコーディングすることができます。

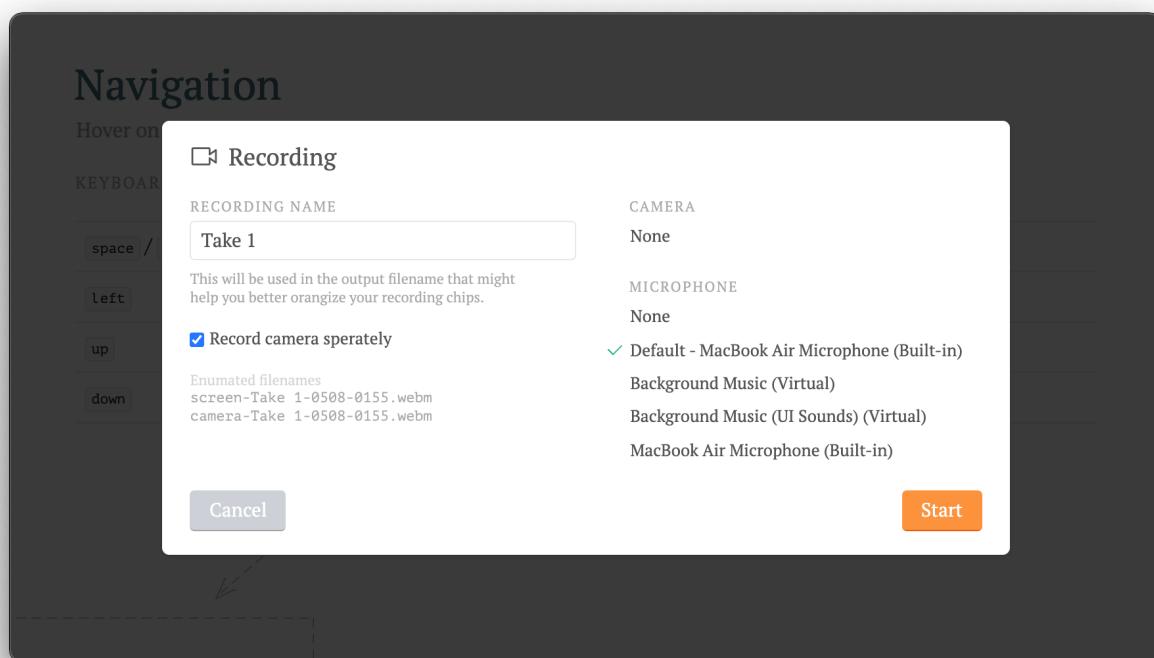
カメラビュー

カメラビューをプレゼンテーションで有効にするには、ナビゲーションパネルの ボタンをクリックしてください。ドラッグで移動、右下のハンドラでサイズ変更も可能です。サイズと位置は `localStorage` に保存され、リフレッシュされても維持されるため、心配する必要はありません。

レコーディング

ナビゲーションパネルの ボタンをクリックすると、ダイアログが表示されます。ここでは、スライドにカメラを埋め込んで録画するか、2つの動画ファイルに分離して録画するかを選択することができます。

この機能は[RecordRTC](#)によって提供され、[WebRTC API](#)を使用しています。



[Go to TOC](#)

Markdownシンタックス

スライドは **1つのマークダウンファイル** (デフォルト : `./slides.md`) の中に記述されます。

Markdownの機能は通常通り使用することができ、インラインHTMLとVueコンポーネントが追加でサポートされています。Windi CSSを使ったスタイリングもサポートされています。スライドを区切るには改行で囲まれた `---` を使用してください。

```
# Slides
```

Hello, World!

```
---
```

```
# Page 2
```

コードブロックを直接使用してハイライト表示する

```
//```ts
console.log('Hello, World!')
//```
```

```
---
```

```
# Page 3
```

Windi CSSとVueコンポーネントを直接使用して、スライドをスタイリングし、リッチにすることができます。

```
<div class="p-3">
  <Tweet id="20" />
</div>
```

フロントマター & レイアウト

スライドのセパレータを **フロントマターブロック** に変換して、各スライドのレイアウトやその他のメタデータを指定します。各フロントマターはトリプルダッシュで始まり、トリプルダッシュで終わります。その間にあるテキストはYAML形式のデータオブジェクトになります。例：

```
---
```

```
layout: cover
```

```
---
```

```
# Slides
```

これはカバーページです。

```
---
```

```
layout: center
background: './images/background-1.png'
class: 'text-white'
```

```
---
```

```
# Page 2
```

これはレイアウト`center`とバックグラウンドイメージが指定されたページです。

Page 3

これはいかなる追加のメタデータもない、デフォルトのページです。

詳細は[カスタマイズ](#)を参照してください。

コードブロック

Slidevを開発した大きな理由の1つは、自分のコードをスライド上で正しく見せる必要があるためです。SlidevではMarkdownフレーバーなコードブロックを使って、意図した通りにコードをハイライトすることができます。

```
//``ts
console.log('Hello, World!')
//``
```

SlidevはシンタックスハイライターとしてPrismとShikiをサポートしています。詳細は[シンタックスハイライト](#)を参照してください。

行のハイライト

特定の行をハイライトするためには、ブラケット`{}`の中に単純に行番号を追加するだけです。行番号のカウントは1から始まります。

```
//``ts {2,3}
function add(
  a: Ref<number> | number,
  b: Ref<number> | number
) {
  return computed(() => unref(a) + unref(b))
}
//``
```

ハイライトする行を複数ステップに分けて変更するには、`|`を使用して行番号を分割してください。例：

```
//``ts {2-3|5|all}
function add(
  a: Ref<number> | number,
  b: Ref<number> | number
) {
  return computed(() => unref(a) + unref(b))
}
//``
```

このサンプルでは、はじめに`a: Ref<number> | number`と`b: Ref<number> | number`を、次にクリックがされた後に`return computed(() => unref(a) + unref(b))`を、最後にコードブロック全体をハイライトします。詳細は[アニメーションガイド](#)を参照してください。

Monacoエディタ

プレゼンテーション中になんらかの変更を加えたいときは、言語名の後ろに`{monaco}`を追加するだけで、ブロックが完全なMonacoエディタに切り替わります！

```
//``ts {monaco}
console.log('HelloWorld')
//``
```

詳細は [Monacoの設定](#) を参照してください。

埋め込みスタイル

Markdownで直接 `<style>` タグを使用すると、[現在のスライド](#)のスタイルをオーバーライドすることができます。

```
# このページはRed

<style>
h1 {
  color: red
}
</style>

---

# 次のスライドには適用されない
```

Markdown内の `<style>` タグは常に [scoped](#) です。グローバルにスタイルをオーバーライドする場合は、[スタイル Windi CSS](#)を搭載しているため、ネストしたCSSや[directives](#) (e.g. `@apply`)を直接利用することができます。

```
# Slidev

> Hello `world`

<style>
blockquote {
  code {
    @apply text-teal-500 dark:text-teal-400;
  }
}
</style>
```

静的アセット

Markdownで書くのと同じように、リモートまたはローカルのURLを指定して画像を使用することができます。

リモートのアセットについては、ビルトインの `vite-plugin-remote-assets` が初回実行時にディスクにキャッシュするため、あとで大きなサイズの画像を読み込み場合でもすぐに読み込むことができます。

![リモートの画像]([https://sli.dev/favicon.png\)](https://sli.dev/favicon.png)

ローカルのアセットについては、`public` フォルダ

![ローカルの画像](pic.png)

カスタムサイズやスタイルを適用したい場合は、`` タグに変換することもできます。

``

ノート

各スライドにメモを取ることもできます。メモは [プレゼンターモード](#) に表示され、プレゼンテーションの際に参照することができます。

Markdownでは、各スライドの最後のコメントブロックはノートとして扱われます。

```
---
layout: cover
---

# Page 1

これはカバーページです。

<!-- これはノートです -->

---

# Page 2

<!-- これは、スライドの内容より前にあるため、ノートではありません。 -->

2ページ目

<!--
これもまたノートです
-->
```

アイコン

Slidevを使用すると、Markdownの中でほとんどすべての人気のあるオープンソースのアイコンセットに直接アクセスすることができます。`vite-plugin-icons` と [Iconify](#)によって提供されています。

命名は [Iconify](#) の `{collection-name}-{icon-name}` の形式に従います。例：

- `<mdi-account-circle />` - from [Material Design Icons](#)
- `<carbon-badge />` - from [Carbon](#)
- `<uim-rocket />` - from [Unicons Monochrome](#)
- `<twemoji-cat-with-tears-of-joy />` - from [Twemoji](#)
- `<logos-vue />` - from [SVG Logos](#)
- その他にも。..

利用可能なすべてのアイコンは [Icônes](#) で閲覧・検索できます。

アイコンのスタイル

他のHTML同様にアイコンをスタイルすることができます。例：

```
<uim-rocket />
<uim-rocket class="text-3xl text-red-400 mx-2" />
<uim-rocket class="text-3xl text-orange-400 animate-ping" />
```

スロット

v0.18から使用可能です

レイアウトによっては、[名前付きスロット](#)を使用して、複数のコントリビューションポイントを提供できます。

例えば、`two-cols` レイアウトでは、左 (`default` スロット) と右 (`right` スロット) の2つのカラムを並べることができます。

```
---  
layout: two-cols  
---  
  
<template v-slot:default>  
# Left  
  
これは左側に表示されます。  
  
</template>  
<template v-slot:right>  
# Right  
  
これは右側に表示されます。  
</template>
```

左

これは左側に表示されます。

右

これは右側に表示されます。

またスロット名のショートハンドシンタックスシュガー `::name::` も用意されています。次の例は前に示した例と全く同じように動作します。

```
---  
layout: two-cols  
---  
  
# Left  
  
これは左側に表示されます。  
  
::right::  
# Right  
  
これは右側に表示されます。
```

またデフォルトのスロットを明示的に指定し、カスタムオーダーで提供することも可能です。

```
---
layout: two-cols
---

::right::
# Right
これは右側に表示されます。

::default::
# Left
これは左側に表示されます。
```

設定

必要な設定はすべてMarkdownファイルで定義することができます。例：

```
---
theme: serif
layout: cover
background: 'https://source.unsplash.com/1600x900/?nature,water'
---

# Sliderv
これはカバーページです。
```

詳細は[フロントマターの設定](#)

LaTeX

Slidevは[KaTeX](#)によってLaTeXをサポートしています。

インライン

インラインで表示する場合は、LaTeXの両側を1つの\$で囲います。

```
$\sqrt{3x-1} + (1+x)^2$
```

ブロック

ブロックで表示するには、2つの(\$\$)を使います。このモードではより大きな記号を使用し、結果を中央に配置します。

```
$$
\begin{array}{c}
\nabla \times \vec{\mathbf{B}} - \nabla \frac{1}{c} \cdot \frac{\partial \vec{\mathbf{E}}}{\partial t} \\
= \frac{4\pi}{c} \vec{\mathbf{J}} \cdot \vec{\mathbf{E}} & \nabla \cdot \vec{\mathbf{E}} = 4\pi \rho
\end{array}
```

```
\nabla \times \vec{\mathbf{E}}\,, +\,, \frac{1}{c}\,, \frac{\partial \vec{\mathbf{E}}}{\partial t} \& = \vec{\mathbf{B}} \\ 
\nabla \cdot \vec{\mathbf{B}} \& = 0 \\
\end{array}
$$
```

詳細： [デモ](#) | [KaTeX](#) | [markdown-it-katex](#)

図形

[Mermaid](#)を利用して、Markdownのテキスト記述から図 / グラフを作成することも可能です。

`mermaid` として指定されたコードブロックは図形に変換されます。例：

```
//```mermaid
sequenceDiagram
    Alice->John: Hello John, how are you?
    Note over Alice,John: A typical interaction
//````
```

さらにオプションオブジェクトを渡すことで、スケーリングやテーマを指定することができます。オブジェクトのシンタックスはJavaScriptのオブジェクトリテラルで、文字列には引用符 (') を、キーの間には (,) を追加する必要があります。

```
//```mermaid {theme: 'neutral', scale: 0.8}
graph TD
    B[Text] --> C{Decision}
    C -->|One| D[Result 1]
    C -->|Two| E[Result 2]
//````
```

詳細： [デモ](#) | [Mermaid](#)

マルチプレンター

v0.15から使用可能です

`slides.md` を複数のファイルに分割して、好きなように整理することができます。

`slides.md` :

```
# Page 1
これは通常のページです。
---
src: ./subpage2.md
---
<!-- このページは'./subpage2.md'から読み込まれます -->
INLINE CONTENTSは無視されます
```

```
subpage2.md :
```

```
# Page 2
```

```
このページは別のファイルのものです
```

フロントマターのマージ

フロントマターはメインのエントリーと外部のMarkdownページの両方から指定することができます。もし同じキーがある場合は、**メインエントリに記載されている内容がより優先度が高くなります**。例：

```
slides.md :
```

```
---
src: ./cover.md
background: https://sli.dev/bar.png
class: text-center
---
```

```
cover.md :
```

```
---
layout: cover
background: https://sli.dev/foo.png
---

# カバー

カバーページ
```

以下のページと同じように評価されます：

```
---
layout: cover
background: https://sli.dev/bar.png
class: text-center
---

# カバー

カバーページ
```

ページの再利用

マルチエントリーにより、ページの再利用が容易になります。例：

```
---
src: ./cover.md
---
```

```
---
src: ./intro.md
---
```

```
---
src: ./content.md
```

```
---  
---  
# reuse  
src: ./content.md  
---
```

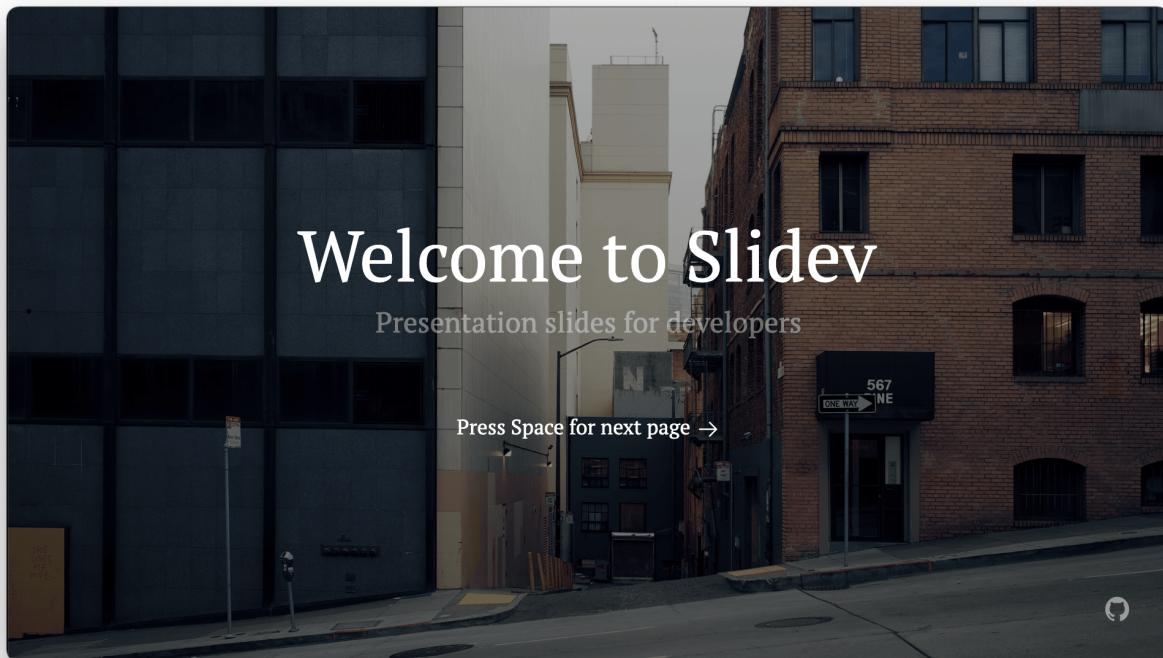
なぜSlidev?

Microsoft PowerPointやApple Keynoteなど、機能豊富で汎用的なWYSIWYGのスライド作成ツールがたくさんあります。これらは非常に直感的で簡単に学ぶことができる一方で、アニメーションやチャート、その他の多くの機能を使った素晴らしいスライドを作成するのに非常によく機能します。では、なぜわざわざSlidevを作ったのでしょうか？

Slidevの目的は、開発者が既に慣れ親しんでいるツールや技術を使って、プレゼンテーションをさらに面白く、表現力豊かに、そして魅力的にするための柔軟性と対話性を提供することです。

WYSIWYGエディタで作業していると、スタイリングオプションに気を取られがちです。Slidevはコンテンツとビジュアルを分離することでそれを改善します。これによって、一度に1つのことに集中できるようになり、同時にコミュニティのテーマを再利用できるようになります。Slidevは他のスライドデッキビルダーに完全に取って代わろうとはしていません。それよりも、開発者コミュニティに応えることに重点をおいています。

Slidev



ここでは、いくつかのSlidevのクールな機能を紹介します：

Markdownベース

Slidevは拡張されたMarkdown形式を使用して、スライドを単一のプレーンテキストファイルに保存し、整理します。これにより、コンテンツの作成に集中することができます。またコンテンツとスタイルが分離されているので、異なるテーマに楽に切り替えることができます。

詳しくは[Markdownシンタックス](#)を参照してください。

豊富なテーマ

Slidevのテーマは、npmパッケージを使用して共有とインストールができます。そして1行設定するだけでテーマを適用することができます。

[テーマギャラリー](#)や[テーマを作成する](#)をチェックしてみてください。

デベロッパーフレンドリー

Slidevは開発者のためにコードスニペットのファーストクラスのサポートを提供します。[Prism](#)と[Shiki](#)の両方をサポートし、ピクセルパーフェクトなシンタックスハイライトを実現しつつ、いつでもコードを修正することができます。[Monaco editor](#)を内蔵し、オートコンプリート、タイプホバーリング、TypeScriptの型チェックサポートにより、プレゼンテーションでのライブコーディングやデモも可能になります。

詳しくは[シンタックスハイライト](#)と[Monacoの設定](#)を参照してください。

高速

Slidevは[Vite](#)、[Vue 3](#)、そして[Windi CSS](#)を利用しておおり、もっとも素晴らしいオーサリング体験を提供しています。あなたが行ったすべての変更は、**即時に**あなたのスライドに反映されます。

詳しくは[技術スタック](#)

インタラクティブ & エクスプレッシブ

Markdownファイルの中に直接Vueのカスタムコンポーネントを記述することができます。また、プレゼンテーションの中でそれらとやりとりすることで、より面白く、より直感的にアイデアを表現することができます。

レコーディングサポート

Slidevはビルトインのレコーディング機能とカメラービューを提供します。カメラービューを含めたプレゼンテーションを共有したり、画面とカメラで別々に録画・保存することも可能です。すべてSlidevだけで完結しており、追加のツールは必要ありません。

詳しくは[レコーディング](#)を参照してください。

ポータブル

コマンド1つでスライドをPDF、PNG、あるいはホスティング可能なSPAとしてエクスポートでき、どこへでも共有することができます。

詳しくは[エクスポート](#)を参照してください。

自由に開発可能

Web技術を使用していることにより、WebアプリでできることはSlidevでも実現可能です。例えば、WebGL、APIリクエスト、iframe、あるいはライブシェアリングなどが利用可能です。あなたの想像力次第でなんでもできます！

試してみる

百聞は一件にしかずということで、実際にSlidevを使ってみましょう。コマンドを実行：

```
$ npm init slidev
```

またはプレビュー：

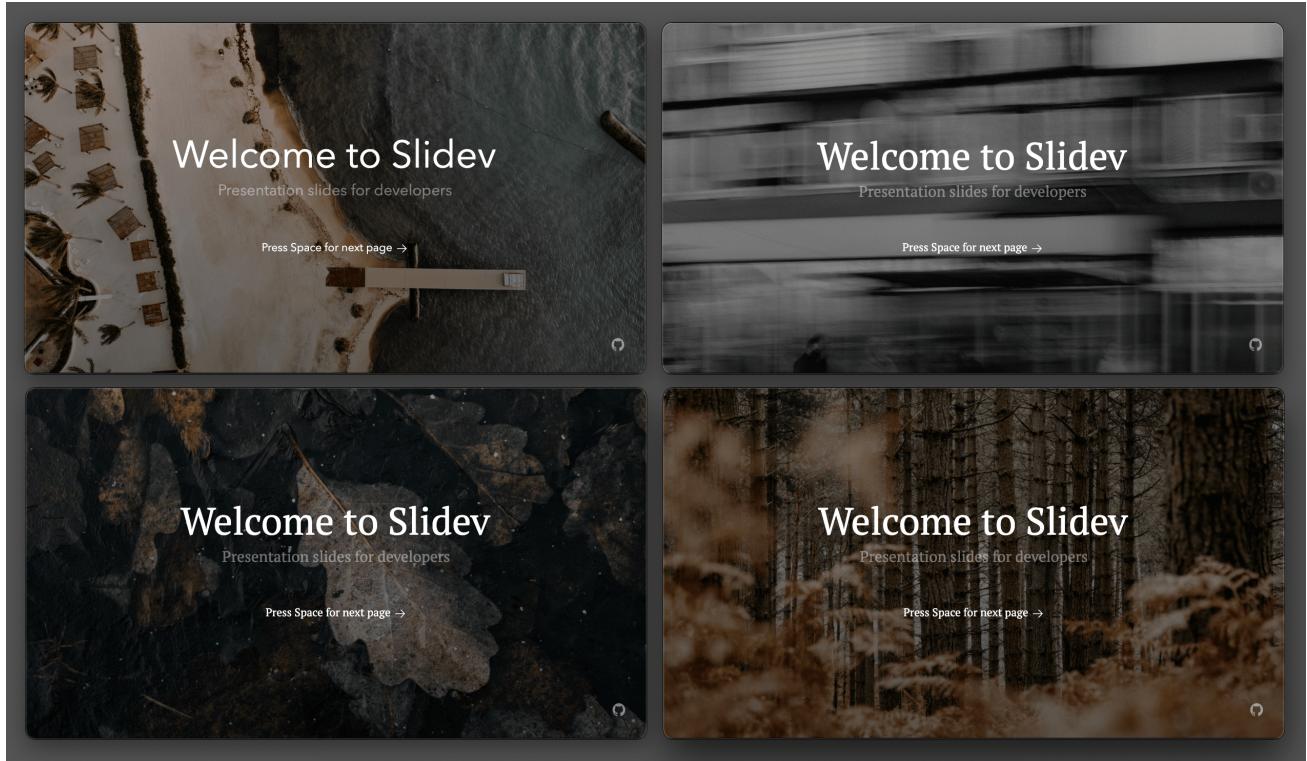
Slidev First Preview Demo



[Go to TOC](#)

キュレーションカバー

スターターテンプレートのデモとして、いくつかのカバー画像をキュレーションしました。



```
---  
# キュレーションコレクションからランダムに画像が表示されます  
background: https://source.unsplash.com/collection/94734566/1920x1080  
---
```

もし気に入ったものがあれば、[Unsplash collection](#)をチェックして作者を探してみてください。

[Go to TOC](#)

学習リソース

英語

動画

Slidev - one of the best presentation software and it is free!



記事

- [Tips To Turn R Markdown Into Slidev Presentation](#) by Hiroaki Yutani

中文

- [Slidev : 一个用Markdown写slides的神器](#) by 梦里风林
- [神器！这款开源项目可以让你使用 Markdown 来做 PPT！](#) by [Github掘金计划](#)
- [【用 markdown 写 Slide!】神器 Slidev 的安装及 bug 解决](#) by HaloHoohoo

日本語

- [開発者のためのスライド作成ツール Slidev がすごい](#) by ryo_kawamata
- [Markdownでオシャレなスライドを作るSli.dev](#) by Nobuko YAMADA

[Go to TOC](#)

ショーケース

Slidevを使ったトーク / プрезентーションです。

[Go to TOC](#)

テーマギャラリー

Slidevで利用できる素晴らしいテーマはこちらでご覧いただけます。

詳細は[テーマを使用する](#)または[テーマを作成する](#)を参照してください。テーマをコミュニティで共有しましょう！

公式テーマ

コミュニティテーマ

コミュニティで作成されたキュレーションテーマを紹介します。

その他のテーマ

NPMで利用可能な[テーマ](#)をすべて検索できます。

[Go to TOC](#)

テーマを使用する

Slidevのテーマを変更するのは驚くほど簡単です。フロントマターに `theme:` フィールドを追加するだけです。

```
---  
theme: serif  
---
```

サーバーを起動すると、テーマの自動インストールを促すメッセージが表示されます。

? The theme "@slidev/theme-serif" was not found in your project, do you want to install it now? > (Y/n)

あるいは以下の方法で手動でテーマをインストールします

```
$ npm install @slidev/theme-serif
```

以上、新しいテーマをお楽しみください。各テーマの使い方の詳細については、テーマのREADMEを参照してください。

あなたのテーマを共有したいですか？[テーマを作成する](#)を参照してください。

テーマの取り出し

現在のテーマを完全に制御したい場合は、ローカルファイルシステム上に取り出しして好きなように変更することができます。次のコマンドを実行します。

```
$ slidev theme eject
```

現在使用しているテーマを `./theme` に出力し、フロントマターを次のように変更します。

```
---  
theme: ./theme  
---
```

また、これは既存のテーマをベースにテーマを作成するのに参考になります。その際は、元のテーマと作者について言及することを忘れないでください :)

ローカルテーマ

前のセクションでおわかりのように、プロジェクトにローカルテーマを指定することができます。`theme`フィールドに相対パスを指定します。

```
---  
theme: ./path/to/theme  
---
```

詳細は[テーマを作成する](#)を参照してください。

[Go to TOC](#)

テーマを作成する

最初のテーマを作成するために、ジェネレータを作成することを推奨します。

```
$ npm init slidev-theme
```

作成されたテーマを修正し、試すことができます。例として [公式テーマ](#)を参照することもできます。

できること

テーマでは次のことができます：

- グローバルスタイル
- デフォルトの設定を指定する（フォント、カラースキーマ、ハイライト、など）
- カスタムレイアウトを指定する、もしくは既存のレイアウトを上書きする
- カスタムコンポーネントを指定する、もしくは既存のコンポーネントを上書きする
- Windi CSSの設定を拡張する
- MonacoやPrismのようなツールの設定をする

規約

テーマはnpmレジストリに公開され、以下の規約に従います。

- パッケージ名は `slidev-theme-` から始めます。例：`slidev-theme-awesome`
- `package.json` の `keywords` フィールドに、`slidev-theme` と `slidev` を追加します。

セットアップ

テーマのテスト用プレイグラウンドをセットアップするには、以下のようなフロントマターで `example.md` を作成し、現在のディレクトリをテーマとして使用することをSlidevに伝えます。

```
---  
theme: ./  
---
```

オプションで、いくつかのスクリプトを `package.json` に追加することもできます。

```
// package.json
{
  "scripts": {
    "dev": "slidev example.md",
    "build": "slidev build example.md",
    "export": "slidev export example.md",
    "screenshot": "slidev export example.md --format png"
  }
}
```

テーマを公開するには `npm publish` を実行すればOKです。ビルドプロセスは必要ありません（つまり、`.vue` と `.ts` ファイルを直接公開することができ、Slidevはスマートなのでそれらを読み込むことができます）。

テーマのコントリビューションポイントはローカルカスタマイズと同じ規約に従います。[命名規約についてのドキュメント](#)を参照してください。

デフォルトの設定

v0.19から使用可能です

テーマでは `package.json` を介して、デフォルトの設定

```
// package.json
{
  "slidev": {
    "default": {
      "aspectRatio": "16/9",
      "canvasWidth": 980,
      "fonts": {
        "sans": "Robot",
        "mono": "Fira Code"
      }
    }
  }
}
```

フォントは[Google Fonts](#)から自動でインポートされます。

詳細は[フォントとフロントマターの設定](#)

メタデータ

カラースキーマ

デフォルトでは、Slidevはライトモードとダークモードの両方をサポートするテーマを想定しています。もしデザインされたカラースキーマだけでテーマを表示したいなら、`package.json` で明示的に指定する必要があります。

```
// package.json
{
  "name": "slidev-theme-my-cool-theme",
  "keywords": [
    "slidev-theme",
    "slidev"
  ],
  "slidev": {
    "colorSchema": "light" // or "dark" or "both"
  }
}
```

テーマのスタイルを作成する際にダークモードにアクセスするには、ダークモード特有の設定を `dark` クラスで囲みます：

```
/* 全体に適用されるCSS */
html:not(.dark) {
    /* ライトモードのCSS */
}

html.dark {
    /* ダークモードのCSS */
}
```

Slidevはカラースキーマを切り替えるために、ページの `html` 要素の `dark` クラスを切り替えます。

シンタックスハイライト

シンタックスハイライトの色もテーマで指定することができます。PrismとShikiの両方をサポートしています。詳細は[シンタックスハイライトについてのドキュメント](#)を参照してください。

どちらかだけをサポートすることもできます。サンプルとして、デフォルトテーマを参照してください
`./styles/code.css` / `./setup/shiki.ts`

また `package.json` でサポートしているシンタックスハイライトを指定することを忘れないでください。

```
// package.json
{
    "slidev": {
        "highlighter": "shiki" // or "prism" or "all"
    }
}
```

Slidevのバージョン

テーマが新しく追加されたSlidevの機能に依存している場合は、テーマが正しく動作するのに必要な最小のSlidevのバージョンを指定する必要があります。

```
// package.json
{
    "engines": {
        "slidev": ">=0.19.3"
    }
}
```

ユーザーが指定されたバージョンよりも古いSlidevを使用している場合、例外が発生します。

[Go to TOC](#)

Colophon

This book is created by using the following sources:

- Slidev - 日本語
- GitHub source: [slidevjs/docs-ja](https://github.com/slidesjs/docs-ja)
- Created: 2022-11-27
- Bash v5.2.2
- Vivliostyle, <https://vivliostyle.org/>
- By: @shinokada
- GitHub repo: <https://github.com/shinokada/markdown-docs-as-pdf>