PEITHO TOUR

Peithoツアー

コンセプト

- **書くことから届けることまで、ツールはひとつ** —— 執筆・プレビュー・発表・PDF出力・公開まですべて `peitho` で完結。途中でツールを切り替えない。
- **内容は自分の手で、デザインはAIに任せる** —— 書くのはMarkdownだけ。デザインは平文のHTML/CSSでLLMが生成しやすく、人間はgitのdiffとレビューで舵を取れる。

3つの設計柱

- **内容とデザインは分離する** —— 内容はMarkdown、デザインはレイアウトHTMLとテーマCSS。混ぜない。
- **バージョン管理できるデザイン** —— レイアウトもテーマも平文のHTML/CSS。diffが取れて、レビューできる。
- **型付きスロット契約** —— 内容がレイアウトに合わないとき、こっそり捨てられることはなく、行番号付きのビルドエラーで止まる。

書く、プレビューする、発表する

- **書く** —— 内容は `deck.md` に Markdown で書く。デザインはレイアウトHTMLとテーマCSSに閉じ込め、Markdownには一切漏らさない。
- **プレビューする** —— `peitho preview` が保存のたびにビルドし、ブラウザをリロードする。
- **発表する** —— `peitho present` が外部ディスプレイにスライドを、手元マシンに発表者ビューを配置する。

インストール

PeithoはHomebrewで配布しています。レイアウトも、ベーステーマも、発表者シェルも全部バイナリに埋め込まれているので、Node.jsもnpmもランタイム依存はありません。`deck.md`が1枚あれば、どのディレクトリでも始められます。

```
brew install mizzy/tap/peitho
```

スライドはただのMarkdown

スライドは `---` で区切ります。最も浅い見出しがタイトル、フェンス付きコードブロックがコード、それ以外は本文 —— それぞれが対応するスロットに落ちていきます。スロット自体はレイアウトHTMLが宣言していて、これは「デザインする」パートで詳しく見ます。

```
# 最初のスライド

本文の段落。

- 箇条書きも
- 使えます

---

# 次のスライド
```

フロントマターでデッキ全体を設定

デッキ先頭のYAMLでは7つのキーを受け付けます: `time`、`aspect_ratio`、`resolution`、`layouts`、`css`、`syntaxes`、`fonts`。省略した項目はデッキ隣接の同名ディレクトリ、次に組み込みのデフォルトへとフォールバックします。存在しないパスや未知のキーは行番号付きのビルドエラーになります。

```
---
time: 23m             # 発表予定時間
aspect_ratio: 16:9    # 16:9 か 4:3
layouts: ./layouts    # レイアウトHTML(ファイルかディレクトリ)
css: ./css            # テーマCSS(ファイルかディレクトリ)
syntaxes: ./syntaxes # 追加のsublime-syntax定義
fonts: ./fonts        # 同梱するwebフォント
---
```

スライド単位の設定はJSONコメント

スライドの先頭にある `<!-- {...} -->` がそのスライドのページ設定です。`layout` は名前でレイアウトを指定、`key` はスライドにCSSで狙える安定した識別子を付け、`section` と `time` はアジェンダのセクションを宣言します。

```
<!-- {"key":"arch-1","layout":"code-demo","section":"詳細","time":"5m"} -->
# アーキテクチャ
```

発表者ノートはHTMLコメント

JSONでないHTMLコメントは、そのスライドの発表者ノートになります。複数書くと連結されます。ノートは発表者ビューにだけ乗り、公開する `dist/` には決して漏れません。

```
# 締めのスライド

<!-- ここで一呼吸置いて質問を受ける。 -->
<!-- デモが死んでいたら録画に切り替える。 -->
```

画像がメインのスライド

画像1枚だけの段落を書くと、Peithoは `accepts="image"` スロットを持つレイアウトへスライドを振り分けます。画像はデッキ相対のローカルパスで、拡張子は `png`、`jpg`、`jpeg`、`gif`、`webp` に限られます。リモートURLや絶対パス、親ディレクトリ脱出はビルドエラーです。

```
![アーキテクチャ図](img/arch.png)
```

セクションと時間配分

ページ設定コメントで `section` を宣言すると、そのスライドから次の宣言までがひとつのアジェンダセクションになり、`time` はそのセクションの持ち時間です。合計がフロントマターの `time` と合わなければ、行番号付きのビルドエラーになります。発表中は、発表者ビューが計画と実測を並べて見せてくれます。

```
---
time: 15m
---

<!-- {"section":"準備","time":"3m"} -->
# 準備

---

<!-- {"section":"本題","time":"12m"} -->
# 本題
```

レイアウト自体がスキーマ

デザインの単位はレイアウト —— 平文のHTMLファイルです。`<slot>` 要素がスロットの名前、受け付ける型、個数を宣言し、Peithoはビルド時にそれを契約として読みます。`layouts/` ディレクトリをデッキの隣に置けば自動で拾われます。

```
<section class="peitho-slide">
  <h1><slot name="title" accepts="inline" arity="1"></slot></h1>
  <slot name="body" accepts="blocks" arity="0..*"></slot>
  <slot name="code" accepts="code" arity="0..1"></slot>
</section>
```

レイアウトはどう選ばれるか

- **明示指定が最強** —— ページ設定に `{"layout":"cover"}` があればそれを使う。未知の名前は候補一覧つきビルドエラー。
- **1個しかなければ、いつもそれ** —— 契約違反は通常どおりエラーになります。
- **それ以外は型駆動ディスパッチ** —— スライドの中身の形と各レイアウトのスロット契約を突き合わせる。一致がちょうど1件のときだけ通り、0件や複数件はビルドエラーです。

合わないときはビルドが止まる

スロット過不足、型の食い違い、存在しないキーを狙ったCSS、合計がずれたセクション時間 ——どれも行番号付きのビルドエラーで止まります。こっそり捨てられることはありません。

```
error: slide 2 ('code-slide'), line 7: slot 'code' got 2 item(s),
       but layout 'title-body-code' allows 0..1
  = help: use a layout with more code capacity or remove one code block
```

特定のスライドだけを狙う

`key` を宣言したスライドはHTMLで `data-slide-key` 属性を持つので、CSSでそのスライドだけを狙えます。タイトルを直してもCSSは壊れず、存在しないkeyを狙ったセレクタはビルドエラーで捕まえられます。

```
<!-- {"key":"arch-1"} -->
# アーキテクチャ
```

```
[data-slide-key="arch-1"] .slot-code {
  grid-column: 2 / 3;
  width: 60%;
}
```

編集ループは peitho preview

`peitho preview` はデッキとレイアウトとCSSを監視します。保存のたびにリビルドしブラウザをリロードしますが、表示中のスライドとオーバービュー状態はリロード後も維持されるので、いま調整しているスライドが視界から消えることはありません。

One contract, end to end

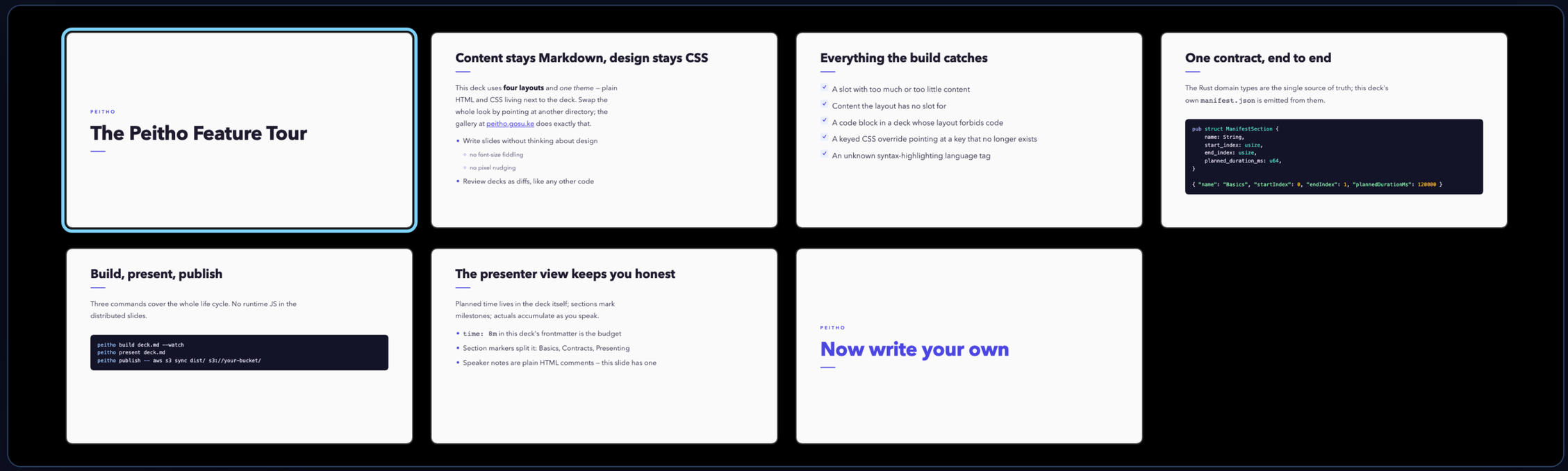
The Rust domain types are the single source of truth; this deck's own `manifest.json` is emitted from them.

```
pub struct ManifestSection {
    name: String,
    start_index: usize,
    end_index: usize,
    planned_duration_ms: u64,
}

{ "name": "Basics", "startIndex": 0, "endIndex": 1, "plannedDurationMs": 120000 }
```

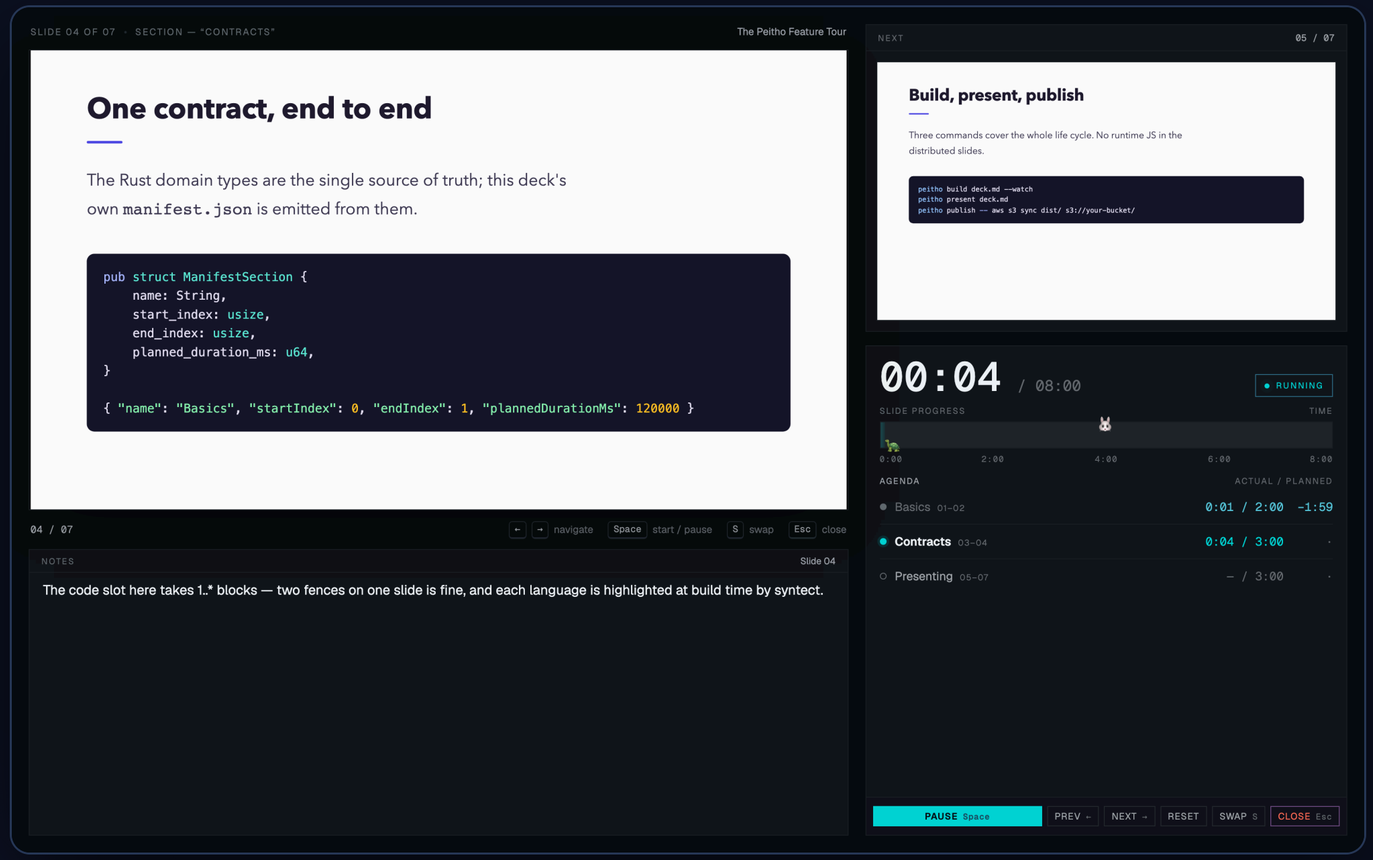
o でオーバービュー

o 、Enter、Escで単一スライドとタイル表示を切り替えられます。タイル表示中は矢印キーでグリッド移動、クリックかEnterでそのスライドが開きます。全体像の確認とジャンプの両方に使えます。



peitho present で本番

`peitho present` は外部ディスプレイにフルスクリーンでスライドを、手元マシンには発表者ビューを配置します。現在スライド、次スライド、発表者ノート、タイマー、セクション別の計画対実測がひとつの画面に載ります。スペースでタイマー開始、Escで全部閉じます。

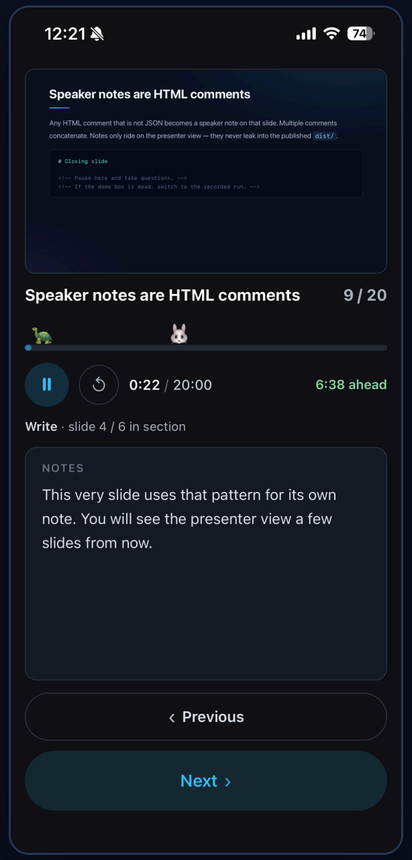


手元のスマホをリモコンに

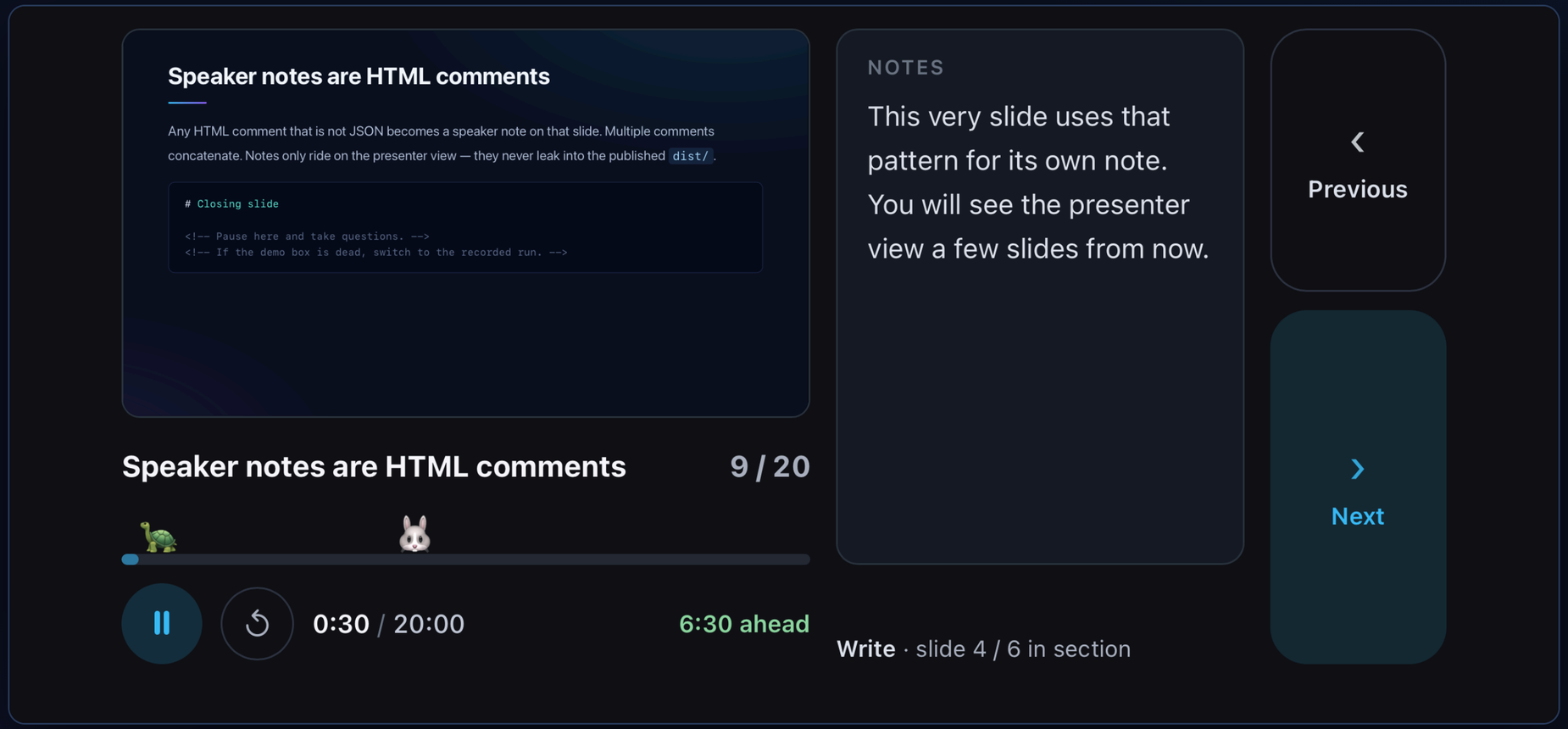
`peitho present --host` は `/remote` をLAN(TailscaleのようなVPNがあれば優先)で公開し、ターミナルにQRコードを表示、ポートは `6173` で固定するのでURLは毎回同じです。SafariでQRを一度読み、共有シートから「ホーム画面に追加」をタップすると、次からはSafariのアドレスバーなしでリモコンが立ち上がります。

```
peitho present --host
```

縦持ち: プレビュー上、ノートとボタン下



横持ち: プレビュー左、ノート中央、ボタン右レール



PDFに書き出す、どこにでも公開する

PDF出力はコマンド1つです。公開は `peitho publish` が `dist/` を検品したあと、`--` の右に置いたあなたのデプロイコマンドへ処理を渡します。Peithoはデプロイを再発明しません。

```
peitho export pdf -o deck.pdf
peitho publish -- aws s3 sync dist/ s3://your-bucket/
```

PEITHO TOUR

まずはMarkdownで1枚から