Quarto × reveal.js デモ

ラボ共有資料

2026-02-06

目次

ゴール

- Markdown 1ファイルから reveal.js スライドを生成
- 同じ原稿を PDF にも落とし込んで配布に備える
- 再現できるコマンドを残し、研究室の誰でも追従できるようにする

ディレクトリ構成

```
project-root/
└── demo_quarto/
    ├── slides.qmd   # このファイル（ソース）
    ├── slides.html  # reveal.js 版（`quart
    └── slides.pdf   # PDF 版（`?print-pdf`
```

ワークフロー

1. `quarto check` で必要な依存関係（Pandoc / LaTeX等）を確認
2. `quarto preview slides.qmd` でライブプレビュー
3. `quarto render slides.qmd --to revealjs` でHTML確定
4. `google-chrome --headless --print-to-pdf` で配布用PDFを同梱

reveal.js で押さえるポイント

- セクション（`##`）= スライド
- `###` でサブスライドを重ねる
- `incremental: true` で箇条書きを段階表示
- `embed-resources: true` で単一HTMLにバンドル
- スピーカーノートは `::: notes` ブロック
- `layout: true` スライドで背景やレイアウトを共有可能

PDF 出力の注意

- reveal.js の `?print-pdf` モードを使うと登壇画面と同じ状態をそのままPDF化できる
- Chromeヘッドレス実行時は `--no-sandbox --disable-dev-shm-usage` がCIでも安定
- LaTeX派生のPDFが必要な場合は `format.pdf` を追加して `quarto render --to pdf`

サンプルコードブロック

```
$ quarto render slides.qmd --to revealjs
$ google-chrome --headless --disable-gpu -
    --disable-dev-shm-usage --print-to-pdf
    "file://$(pwd)/slides.html?print-pdf"
```

```python
# デモ用: プレースホルダデータを生成
import numpy as np
np.random.seed(42)
noise = np.random.normal(size=100)
trend = np.linspace(0, 1, 100)
data = trend + 0.2 * noise
print(data[:5])
```

まとめ

- Markdownで台本を集中管理し、出力形式はQuartoに任せる
- reveal.jsを本番、PDFを配布という役割分担で破綻を防げる
- `quarto render` をCIに組み込めば、配布忘れ・崩れを早期検知できる