Page Structure Redesign Proposal

Overview

Modernize the demo application into a responsive, educational resource that showcases both D3 visualization techniques and the Finally Tagless pattern in PureScript.

Current State Analysis

What Works

- Clear categorization (Simple Demos, Interpreters, Spago App)
- Code snippets are extracted and displayed
- Direct examples similar to D3's own documentation

Pain Points

- Not responsive to different screen sizes
- Code and visualization compete for attention
- No easy way to compare PureScript with D3 JavaScript
- Hard to demonstrate the interpreter pattern's power
- Limited discoverability of examples

Proposed Structure: Three-Tier Approach

Tier 1: Landing/Gallery View (NEW)

Purpose: Quick browsing and discovery

```
┌──────────────────────────────────────────────────────────┐
│  PureScript Tagless D3                                   │
│  Type-safe, composable data visualization               │
├──────────────────────────────────────────────────────────┤
│  [Getting Started] [Examples] [Patterns] [GitHub]        │
├──────────────────────────────────────────────────────────┤
│                                                          │
│   ┌──────────┐  ┌──────────┐  ┌──────────┐             │
│   │ Line     │  │ Bar      │  │ Scatter  │  Simple     │
│   │ Chart    │  │ Chart    │  │ Plot     │  Charts     │
│   │ [thumb]  │  │ [thumb]  │  │ [thumb]  │             │
│   └──────────┘  └──────────┘  └──────────┘             │
│                                                          │
│   ┌──────────┐  ┌──────────┐                           │
│   │ Chord    │  │ Bubble   │           Advanced        │
│   │ Diagram  │  │ Chart    │           Layouts         │
│   │ [thumb]  │  │ [thumb]  │                           │
│   └──────────┘  └──────────┘                           │
│
```

```
│                                                          │
│                                                          │
│   ┌──────────┐ ┌──────────┐                              │
│   │ MetaTree │ │ String   │           Alternative  │
│   │ (visual) │ │ (code)   │           Interpreters │
│   │ [thumb]  │ │ [thumb]  │                              │
│   └──────────┘ └──────────┘                              │
│                                                          │
│                                                          │
│   ┌──────────────────────────┐                           │
│   │ Spago Dependency         │        Applications  │
│   │ Explorer                 │                           │
│   │ [larger thumb]           │                           │
│   └──────────────────────────┘                           │
└──────────────────────────────────────────────────────────┘
```

Features:

- Animated thumbnail previews (SVG snapshots)
- Hover to see mini-preview
- Tags/filters: "basic", "interactive", "hierarchical", "network"
- Search functionality
- Visual grouping by complexity/category

Tier 2: Example Detail View (ENHANCED)

Purpose: Deep dive into a specific example

Desktop Layout: Split-Pane

```
┌──────────────────────────────────────────────────────────┐
│ ← Back to Gallery    Line Chart                  [D3 JS] [⚙]│
├──────────────────────┬───────────────────────────────────┤
│                      │                                   │
│  Code Panel          │    Visualization Panel            │
│  ┌──────────────────┐│    ┌──────────────────────────────┐│
│  │ [PureScript] [JS]││    │                              │
│  ├──────────────────┤│    │    [Live Visualization]      │
│  │                  │    │                              │
│  │ draw :: forall m │    │                              │
│  │  ...             │    └──────────────────────────────│
│  │                  │    ┌──────────────────────────────│
│  │                  │    │ Controls (if interactive)    │
│  │                  │    │ [Play] [Reset] [Speed: ▮▮▮▮▮] │
│  │                  │    └──────────────────────────────│
│  │                  │                                   │
│  │                  │    Explanation                    │
│  │                  │    This chart demonstrates...     │
│  │                  │    - Linear scales                │
│  │                  │    - Axis generation              │
│  [Copy] [Download]  │    - Path generation              │
└──────────────────────┴───────────────────────────────────┘
```

Mobile Layout: Tabbed

```
┌──────────────────────────┐
│ Line Chart         [≡]   │
├──────────────────────────┤
│ [Viz] [Code] [About]     │
├──────────────────────────┤
│                          │
│  [Visualization]         │
│                          │
│  [Controls]              │
│                          │
├──────────────────────────┤
│ ↕ Resize                 │
└──────────────────────────┘
```

Tier 3: Interactive Playground (FUTURE)

Purpose: Learn by modifying

Live editing environment where users can:

- Modify parameters and see results
- Switch between interpreters interactively
- Fork examples to experiment

Key Features by Section

1. Navigation & Discovery

Top Navigation Bar

```
Logo | Examples ▾ | Patterns ▾ | Docs | GitHub | Search 🔍
```

Examples Dropdown:

- By Category (Charts, Graphs, Hierarchies, Interactions)
- By Complexity (Beginner, Intermediate, Advanced)
- By D3 Module (Scales, Axes, Shapes, Layouts)

Patterns Dropdown:

- Finally Tagless Explained
- Multiple Interpreters
- Type-Safe Attributes
- Composable Visualizations

Breadcrumb Trail

```
Home > Examples > Simple Charts > Line Chart
```

2. Code Presentation

Multi-Language Toggle

```
┌──────────────────────────────┐
│ [PureScript] [JavaScript]    │  ← Tab switcher
├──────────────────────────────┤
│ draw :: forall m.            │
│   Bind m =>                  │
│   MonadEffect m =>           │
│   ...                        │
└──────────────────────────────┘
```

Show equivalent D3 JavaScript side-by-side or via toggle to help:

- PureScript developers learn D3 patterns
- D3 developers learn PureScript translation

Syntax Highlighting

- Use Prism.js or Highlight.js
- PureScript-specific color scheme
- Highlight differences when comparing with JS

Code Sections with Collapsible Regions

```
-- Setup (click to expand) ▼
...

-- Scales ▼
xScale <- liftEffect $ createLinearScale { domain: [minX, maxX], ... }
yScale <- liftEffect $ createLinearScale { domain: [minY, maxY], ... }

-- Axes ▼
...

-- Drawing ▼
...
```

Line Numbers & Annotations

```
draw :: forall m.                          │ The type signature shows
  Bind m =>                                │ we're in a monadic context
  MonadEffect m =>                         │ with side effects
  SelectionM D3Selection_ m =>             │ and D3 selection capabilities
  Array DataPoint -> Selector -> m Unit
```

3. Visualization Panel

Interactive Features

- **Zoom controls**: +/- buttons, fit-to-screen
- **Export**: PNG, SVG download
- **Share**: Generate URL with current state
- **Responsive preview**: See how it looks at different sizes

Data Controls (where applicable)

```
┌──────────────────────────────┐
│ Data Points: [▮▮▮▮▮▮▮▮▮▮] 50  │
│ Noise:       [▮▮▮▮▮▮▮▮▮▮] 0.2 │
│ [Regenerate]  [Reset]        │
└──────────────────────────────┘
```

4. Educational Content

Inline Explanations

Each example includes:

1. **What**: Brief description (1-2 sentences)
2. **Why**: When to use this visualization
3. **How**: Key concepts demonstrated
4. **Learn More**: Links to D3 docs, PureScript concepts

Pattern Callouts

```
┌──────────────────────────────────────┐
│ 💡 Pattern Spotlight                 │
├──────────────────────────────────────┤
│ This example demonstrates the Finally │
│ Tagless pattern by separating the    │
│ visualization DSL from its interpreter.│
│                                      │
│ [Learn more about Finally Tagless]   │
└──────────────────────────────────────┘
```

Difficulty Indicators

- 🟢 Beginner: Basic shapes and scales
- 🟡 Intermediate: Layouts and interactions
- 🔴 Advanced: Complex simulations, custom interpreters

Special Feature: Interpreter Switcher

The Killer Feature - Demonstrate the power of Finally Tagless

```
┌──────────────────────────────────────────────────────┐
│ Interpreter: [D3 Renderer ▾]                         │
│              • D3 Renderer (default)                 │
│              • MetaTree Visualizer                   │
│              • String/Code Generator                 │
└──────────────────────────────────────────────────────┘
```

When D3 Renderer selected:

```
[Normal visualization appears]
```

When MetaTree Visualizer selected:

```
[Shows the DSL tree structure of the visualization itself]

appendTo
├── Svg
│   ├── viewBox
│   ├── width
│   └── height
└── Group
    ├── transform
    ├── Path
    │   ├── d
    │   ├── fill
    │   └── strokeColor
    └── ...
```

When String Generator selected:

```
[Shows generated code or documentation]

This visualization creates:
- 1 SVG element (800x400)
```

```
  - 1 Group container with margin offset
  - 2 Axis groups (x and y)
  - 1 Path element with 50 data points
```

This visually demonstrates that the **same code** produces **different outputs** based on the interpreter - a core concept of the library.

Responsive Design Breakpoints

Desktop (> 1024px)

- Split pane: Code left, Viz right (60/40 or adjustable)
- Full navigation visible
- Sidebar for related examples

Tablet (768px - 1024px)

- Tabbed interface: Code / Visualization / About
- Simplified navigation
- Stacked layout option

Mobile (< 768px)

- Visualization first, then code (scroll down)
- Hamburger menu for navigation
- Single column layout
- Touch-optimized controls

Enhanced Category Structure

Option A: Keep Current + Add Gallery View

```
Landing/Gallery (NEW)
├── Simple Charts (existing examples)
├── Advanced Layouts (new category)
├── Interactive Patterns (new category)
├── Alternative Interpreters (existing)
└── Applications (Spago example)
```

Option B: Reorganize by Learning Path

```
Getting Started
├── Three Little Circles
└── Bar Chart
```

```
Scales & Axes
├── Line Chart
├── Scatter Plot
└── Multi-axis Examples

Layouts
├── Hierarchical (Trees, Bubble)
├── Network (Force, Chord)
└── Specialized

Interactions
├── Drag & Drop
├── Zoom & Pan
└── Transitions

Advanced Patterns
├── Multiple Interpreters
├── Custom Generators
└── Data Joins (GUP)

Real Applications
└── Spago Dependency Explorer
```

Option C: By D3 Module (matches D3 docs)

```
Scales (d3-scale)
Axes (d3-axis)
Shapes (d3-shape)
├── Line Chart
├── Bar Chart
└── Scatter Plot

Layouts (d3-hierarchy, d3-chord, etc.)
├── Trees
├── Chord Diagram
└── Bubble Chart

Interactions (d3-drag, d3-zoom)
Force Simulation (d3-force)

Meta (PureScript-specific)
├── Alternative Interpreters
└── Pattern Examples
```

Recommendation: Start with Option A (keep familiar structure, add gallery), consider Option B for v2.

Technical Implementation Approach

Phase 1: Foundation (Week 1)

1. **Update Halogen routing** for new structure
2. **Create gallery component** with card-based layout
3. **Implement responsive breakpoints** with CSS Grid/Flexbox
4. **Add split-pane component** (resizable)

Phase 2: Enhanced Example View (Week 2)

1. **Code syntax highlighting** integration
2. **Tab component** for Code/Viz/About
3. **Collapsible sections** in code view
4. **Export functionality** (PNG/SVG)

Phase 3: Educational Features (Week 3)

1. **D3 JavaScript comparison** toggle
2. **Inline annotations** system
3. **Pattern callouts** component
4. **Related examples** sidebar

Phase 4: Interpreter Switcher (Week 4)

1. **Interpreter selector** UI component
2. **Wire up MetaTree** interpreter visualization
3. **Wire up String** interpreter output
4. **Add explanation** of interpreter pattern

Phase 5: Polish (Week 5)

1. **Mobile optimization**
2. **Performance tuning** (lazy loading, code splitting)
3. **Accessibility** (ARIA labels, keyboard navigation)
4. **Analytics** (optional - track popular examples)

Design System

Color Palette

```
Primary:   #4a90e2 (D3 blue)
Secondary: #50c878 (PureScript green)
Accent:    #f39c12 (highlight orange)
Dark:      #2c3e50 (code background)
Light:     #ecf0f1 (page background)
```

Typography

```
Headers:   Inter, system-ui
Code:      Fira Code, Monaco, monospace
Body:      -apple-system, system-ui
```

Components

- **Card**: Rounded corners, subtle shadow, hover effect
- **Button**: Solid or outline variants, consistent spacing
- **Code Block**: Dark theme, line numbers, copy button
- **Tabs**: Underline active state
- **Split Pane**: Draggable divider with min/max constraints

Example Metadata Structure

Enhance each example with structured metadata:

```
type ExampleMetadata = {
  id :: String
, title :: String
, description :: String
, difficulty :: Difficulty
, category :: Array Category
, tags :: Array String
, d3Modules :: Array String          -- e.g., ["d3-scale", "d3-axis"]
, purescript Concepts :: Array String -- e.g., ["Monad", "Finally
Tagless"]
, interactive :: Boolean
, hasComparison :: Boolean           -- Has D3 JS equivalent
, interpreters :: Array Interpreter  -- Which interpreters work with this
, relatedExamples :: Array String    -- IDs of related examples
, learnMore :: Array Link            -- External resources
}

data Difficulty = Beginner | Intermediate | Advanced

data Category
  = BasicChart
  | AdvancedLayout
  | Interactive
  | Interpreter
  | Application

data Interpreter = D3Interpreter | MetaTreeInterpreter | StringInterpreter
```

This metadata drives:

- Gallery filtering/sorting
- Related example suggestions

- Difficulty-based learning paths
- Search functionality

Out-of-the-Box Ideas

1. "Code Journey" Mode

Progressive reveal of the code, step-by-step:

```
Step 1: Setup SVG and dimensions
[Shows just that code, highlights in viz]

Step 2: Create scales
[Reveals scale code, shows scale mapping]

Step 3: Add axes
[Reveals axis code, highlights axes in viz]
...
```

2. "Diff Mode"

Show what changed between two similar examples:

```
Line Chart → Area Chart
+ fill opacity
+ area generator
- stroke color
```

3. "Playground Challenges"

Interactive exercises:

```
Challenge: Make the bars green
[ ] Modify the fill attribute
[ ] Change bar width to 20px
[ ] Add a hover effect

[Check Answer]
```

4. "Export Learning Path"

Let users save a custom learning sequence:

```
Your Learning Path:
☑ Bar Chart
☑ Line Chart
☐ Scatter Plot
☐ Force Directed Graph

[Continue] [Reset] [Share Path]
```

5. "Performance Metrics"

Show render times, element counts:

```
Performance:
• Initial render: 45ms
• Elements created: 127
• Data points: 50
• Memory: 2.3MB
```

6. "Embed Mode"

Generate embeddable code for examples:

```
<iframe src="https://purescript-d3.example/embed/line-chart">
```

7. "Compare Interpreters" Split View

Show all three interpreters simultaneously:

```
┌──────────────┬──────────────┬──────────────┐
│ D3 Renderer  │   MetaTree   │    String    │
│ [viz]        │ [tree viz]   │ [code out]   │
└──────────────┴──────────────┴──────────────┘
```

Accessibility Considerations

1. **Keyboard Navigation**
    - Tab through examples
    - Arrow keys for split pane
    - Escape to close modals
2. **Screen Readers**
    - Alt text for visualization thumbnails

    - ARIA labels for interactive elements
    - Descriptive text for SVG visualizations

3. **Color Contrast**
    - WCAG AA compliance
    - High contrast mode option
    - Colorblind-friendly palettes
4. **Reduced Motion**
    - Respect `prefers-reduced-motion`
    - Optional: disable animations

Success Metrics

How to know if the redesign works:

1. **Engagement**: Time spent on examples, pages per session
2. **Learning**: Can users find and understand examples?
3. **Conversion**: GitHub stars, npm downloads
4. **Feedback**: User surveys, issue reports
5. **Technical**: Page load time, mobile usability score

Phased Rollout

MVP (Minimum Viable Product)

- Gallery view with example cards
- Split-pane detail view (desktop)
- Tabbed view (mobile)
- Syntax highlighting
- Basic responsive design

V1.0

- All examples migrated
- Interpreter switcher working
- D3 JavaScript comparison
- Search and filtering
- Export functionality

V1.1

- Interactive playground
- Code journey mode
- Learning paths
- Advanced filtering

V2.0

- User accounts (save progress)
- Community examples
- Live editing
- Embedding support

References & Inspiration

Excellent example documentation sites to draw from:

1. **Observable** (observablehq.com) - Interactive notebooks
2. **React Three Fiber** (docs.pmnd.rs) - Clean, minimal, example-focused
3. **Stripe Docs** - Split code/content view
4. **Three.js Examples** - Gallery with live previews
5. **Storybook** - Component explorer pattern
6. **MDN Web Docs** - Tabbed examples, clear explanations
7. **TailwindUI** - Copy-paste focused, great search
8. **D3 Graph Gallery** - Visual browsing, filter by type

Conclusion

The proposed redesign transforms the demo application from a simple example viewer into an **interactive learning environment** that:

1. **Teaches D3 concepts** through clear, annotated examples
2. **Demonstrates PureScript patterns** (Finally Tagless, type safety)
3. **Showcases unique features** (multiple interpreters)
4. **Works on all devices** (responsive, mobile-first)
5. **Encourages exploration** (gallery view, related examples, search)

The key differentiator is the **interpreter switcher** - no other D3 library can show the same code producing visualization, meta-visualization, and documentation simultaneously. This makes the abstract concept of "separation of concerns" concrete and visual.

Proposal created: 2025-10-16

Next Steps: Review, prioritize features, create wireframes/mockups, begin Phase 1 implementation.