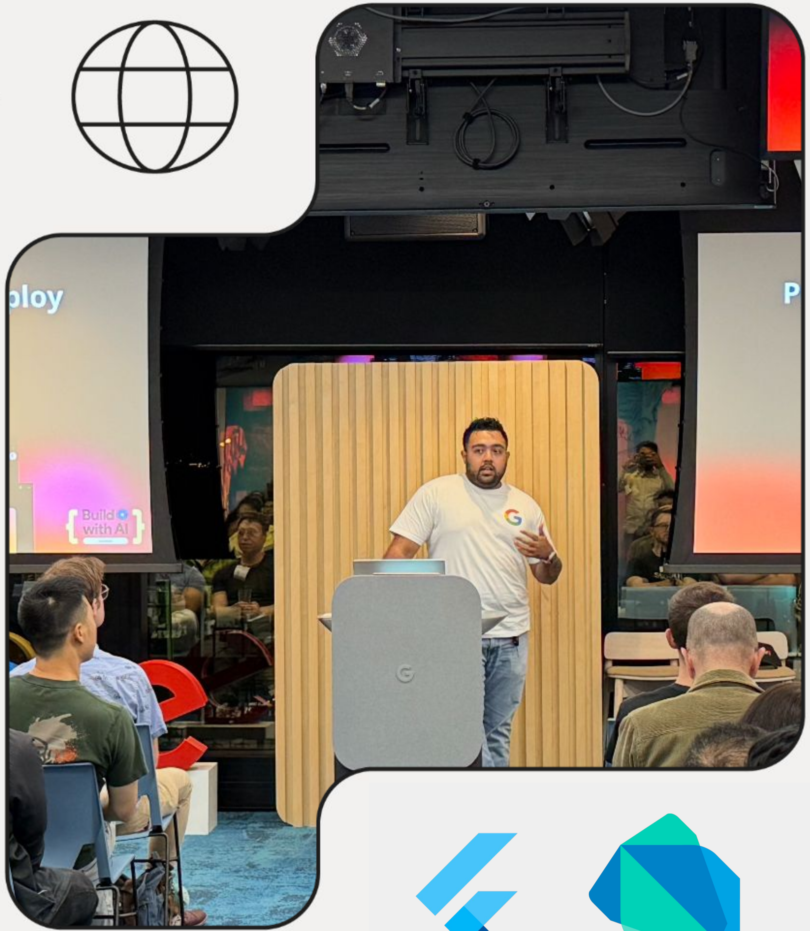
DevFest
Melbourne

# Widgets to Catalog

## Build your Flutter design systems

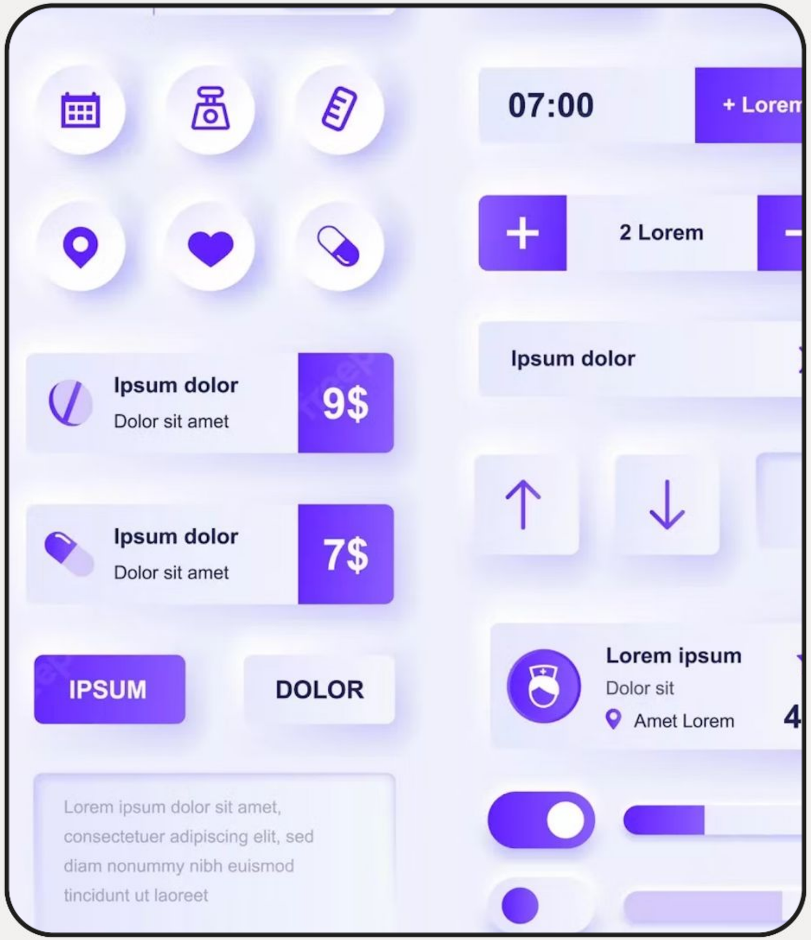


Dilum De Silva
Google Developer Expert
Sr Engineer @ AMP

How do you document and showcase all your custom widgets?

Figma

Confluence



## Challenges in widget development

- Hard to visualize widgets in isolation
- Difficult to test different states/configurations
- No central catalog for design system components
- Designer-developer collaboration gaps

## Widgetbook

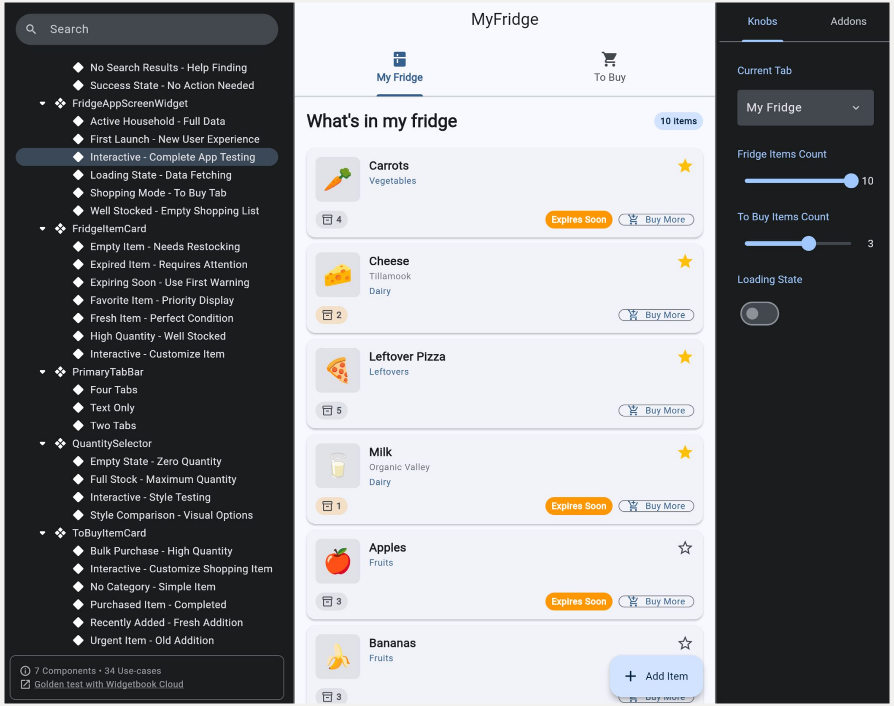
pub.dev/packages/widgetbook

## Flutter’s answer to Storybook

Widgetbook is a Flutter package that provides a **catalog/showcase** environment for your Flutter widgets, allowing you to **develop**, **test**, and **document** UI components in **isolation** from your main application.

Storybook is a hugely popular tool in the web development world (React, Vue, Angular)

# Core Concepts of Widgetbook?


Search

No Search Results - Help Finding
Success State - No Action Needed
FridgeAppScreenWidget
Active Household - Full Data
First Launch - New User Experience
Interactive - Complete App Testing
Loading State - Data Fetching
Shopping Mode - To Buy Tab
Well Stocked - Empty Shopping List
FridgeItemCard
Empty Item - Needs Restocking
Expired Item - Requires Attention
Expiring Soon - Use First Warning
Favorite Item - Priority Display
Fresh Item - Perfect Condition
High Quantity - Well Stocked
Interactive - Customize Item
PrimaryTabBar
Four Tabs
Text Only
Two Tabs
QuantitySelector
Empty State - Zero Quantity
Full Stock - Maximum Quantity
Interactive - Style Testing
Style Comparison - Visual Options
ToBuyItemCard
Bulk Purchase - High Quantity
Interactive - Customize Shopping Item
No Category - Simple Item
Purchased Item - Completed
Recently Added - Fresh Addition
Urgent Item - Old Addition

7 Components • 34 Use-cases
Golden test with Widgetbook Cloud

MyFridge

My Fridge | To Buy

What's in my fridge — 10 items

Carrots — Vegetables — 4 — Expires Soon — Buy More
Cheese — Tillamook — Dairy — 2 — Buy More
Leftover Pizza — Leftovers — 5 — Buy More
Milk — Organic Valley — Dairy — 1 — Expires Soon — Buy More
Apples — Fruits — 3 — Expires Soon — Buy More
Bananas — Fruits — 3

+ Add Item

Knobs | Addons

Current Tab: My Fridge
Fridge Items Count: 10
To Buy Items Count: 3
Loading State

## 01 Use cases (categories)

Different **variations** or **states** of a single widget. Each use case shows one specific example - like a button in its default, disabled, or loading state.

You might also hear them called '**stories**' if you're familiar with Storybook

02

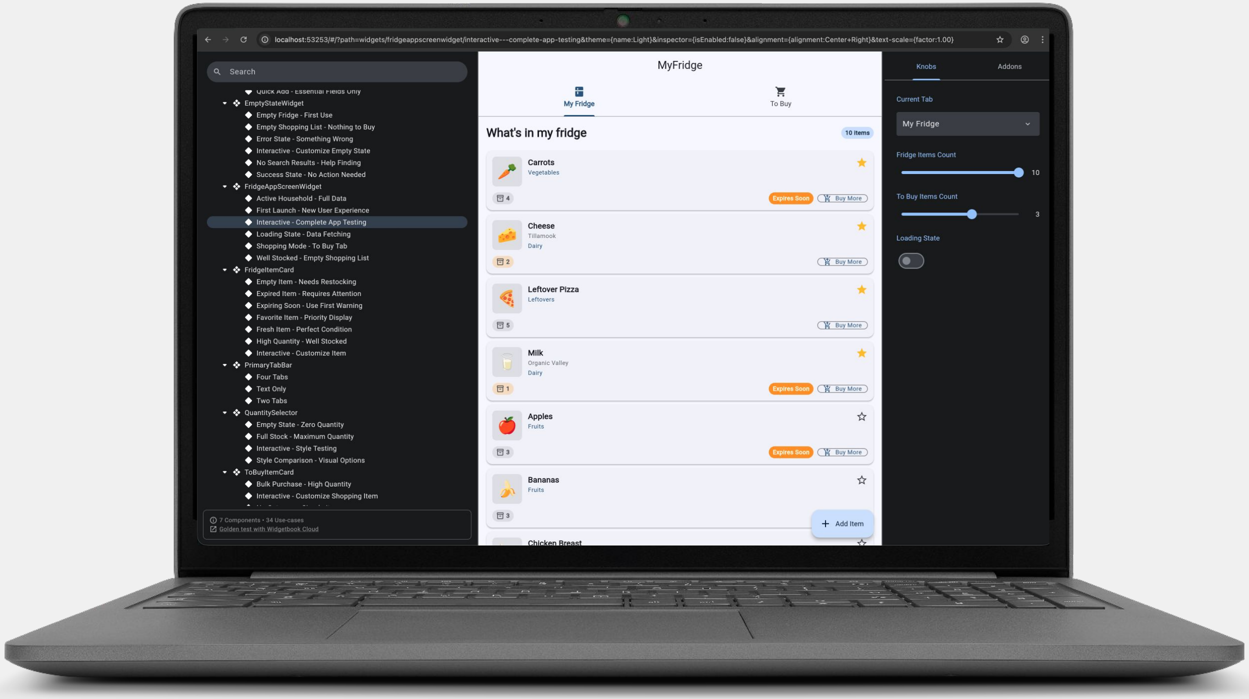
## Knobs (interactive controls)

Interactive controls that let you **modify** widget **properties** in **real-time without changing code**. Toggle booleans, adjust numbers, change text, or select options - all while the app is running.

## 03 Add-ons (themes, devices)

Global features that apply to all your widgets at once. Switch between light/dark **themes**, preview on different **device frames** (iPhone, Android tablet), test different **languages**, or adjust **text scaling**.

# Widgetbook Demo



```
localhost:53253/#/?path=widgets/fridgeappscreenwidget/interactive---complete-app-testing&theme={name:Light}&inspector={isEnabled:false}&alignment={alignment:Center+Right}&text-scale={factor:1.00}
```

- Quick Add - Essential Fields Only
- EmptyStateWidget
  - Empty Fridge - First Use
  - Empty Shopping List - Nothing to Buy
  - Error State - Something Wrong
  - Interactive - Customize Empty State
  - No Search Results - Help Finding
  - Success State - No Action Needed
- FridgeAppScreenWidget
  - Active Household - Full Data
  - First Launch - New User Experience
  - Interactive - Complete App Testing
  - Loading State - Data Fetching
  - Shopping Mode - To Buy Tab
  - Well Stocked - Empty Shopping List
- FridgeItemCard
  - Empty Item - Needs Restocking
  - Expired Item - Requires Attention
  - Expiring Soon - Use First Warning
  - Favorite Item - Priority Display
  - Fresh Item - Perfect Condition
  - High Quantity - Well Stocked
  - Interactive - Customize Item
- PrimaryTabBar
  - Four Tabs
  - Text Only
  - Two Tabs
- QuantitySelector
  - Empty State - Zero Quantity
  - Full Stock - Maximum Quantity
  - Interactive - Style Testing
  - Style Comparison - Visual Options
- ToBuyItemCard
  - Bulk Purchase - High Quantity
  - Interactive - Customize Shopping Item

7 Components • 34 Use-cases
Golden test with Widgetbook Cloud

MyFridge

My Fridge | To Buy

What's in my fridge — 10 items

| Item | Details | Quantity | Tags |
|---|---|---|---|
| Carrots | Vegetables | 4 | Expires Soon, Buy More |
| Cheese | Tillamook, Dairy | 2 | Buy More |
| Leftover Pizza | Leftovers | 5 | Buy More |
| Milk | Organic Valley, Dairy | 1 | Expires Soon, Buy More |
| Apples | Fruits | 3 | Expires Soon, Buy More |
| Bananas | Fruits | 3 | |
| Chicken Breast | | | |

Add Item

Knobs | Addons

Current Tab: My Fridge
Fridge Items Count: 10
To Buy Items Count: 3
Loading State: off

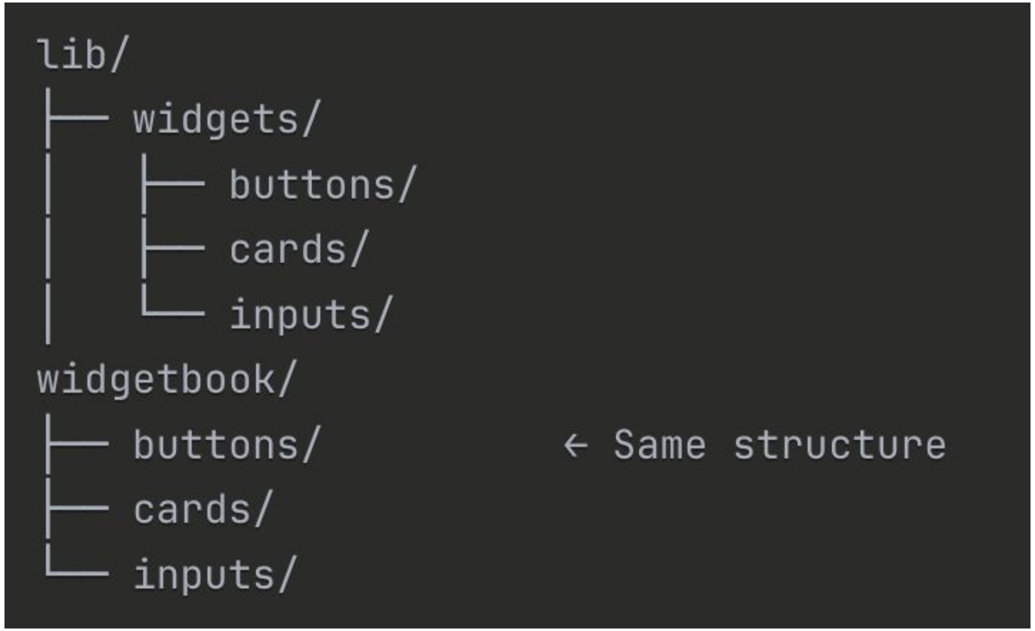
## Key benefits:

- Isolated widget development
- Documentation as code
- Multiple device previews
- Different themes/localizations testing
- Reusable widgets become a practice

## Best Practices

I will adopt best practices
I will adopt best practices
I will adopt best practices
I will adopt best practices
I will adopt best practices
I will adopt best practices
I will adopt best practices

## Mirror your app's component structure



```
lib/
├── widgets/
│   ├── buttons/
│   ├── cards/
│   └── inputs/
widgetbook/
├── buttons/          ← Same structure
├── cards/
└── inputs/
```

# Keep Widgetbook separate

```
# ✅ Good - separate directory
project/
├── lib/                # Production code
└── widgetbook/         # Widgetbook catalog

# ❌ Bad - mixed together
lib/
├── widgets/
│   ├── button.dart
│   └── button.widgetbook.dart  # Don't do this
```

**Ask: Would a designer or developer need to see this variation?**

**Cover** all meaningful states, **Don't** overdo it.

## Use knobs wisely - don't overcomplicate

```
// ❌ Too many knobs = overwhelming
builder: (context) => MyButton(
  text: context.knobs.string(label: 'Text', initialValue: 'Click'),
  fontSize: context.knobs.double.slider(label: 'Font Size', ...),
  paddingTop: context.knobs.double.slider(label: 'Padding Top', ...),
  paddingBottom: context.knobs.double.slider(label: 'Padding Bottom', ...),
  borderRadius: context.knobs.double.slider(label: 'Border Radius', ...),
  // ... 10 more knobs
),

// ✅ Focused on key properties
builder: (context) => MyButton(
  text: context.knobs.string(label: 'Button Text', initialValue: 'Click Me'),
  enabled: context.knobs.boolean(label: 'Enabled', initialValue: true),
  size: context.knobs.list(label: 'Size', options: ['small', 'medium', 'large']),
),
```

Team **Collaboration**, Make it **accessible.**

# Thank You!

Slides and code can be found at

dilumdesilva.dev