

Bridging the Gap Between UX Design and Development

Atomic Design & Storybook.js



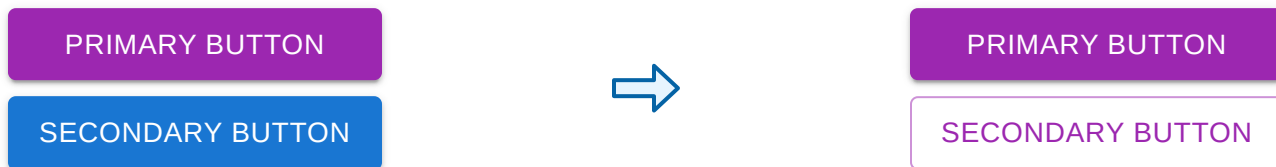
Franz Thoma



13.3.2026

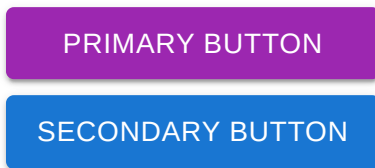


BobKonf 2026, Berlin

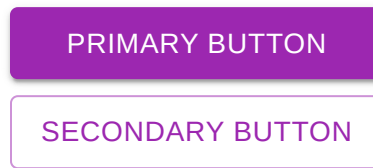


This change took multiple weeks to complete.

Can You Spot the Problem?



```
<Button variant="contained" color="primary">  
  Primary Button  
</Button>  
  
<Button variant="contained" color="secondary">  
  Secondary Button  
</Button>
```



```
<Button variant="contained" color="primary">  
  Primary Button  
</Button>  
  
<Button variant="outlined" color="primary">  
  Secondary Button  
</Button>
```

What we did

```
<Button variant="contained" color="primary">  
  Primary Button  
</Button>  
  
<Button variant="outlined" color="primary">  
  Secondary Button  
</Button>
```

What we should have done

```
<AcmeButton variant="primary">  
  Primary button  
</AcmeButton>  
  
<AcmeButton variant="secondary">  
  Secondary button  
</AcmeButton>
```

The Handover Process



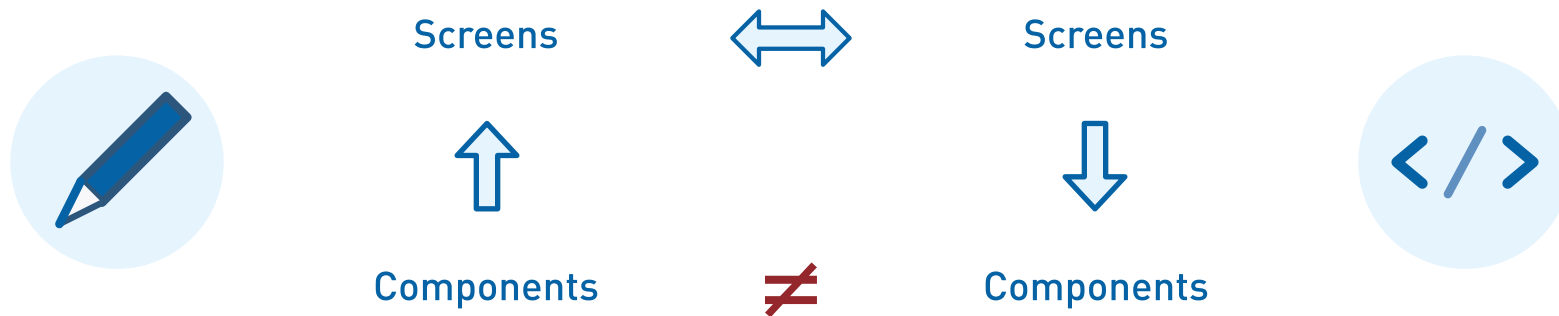
UX-Designers work in Figma



Handover: Full screens in Figma,
review of finished implementation



Developers reproduce designs in
code, and use screens to verify



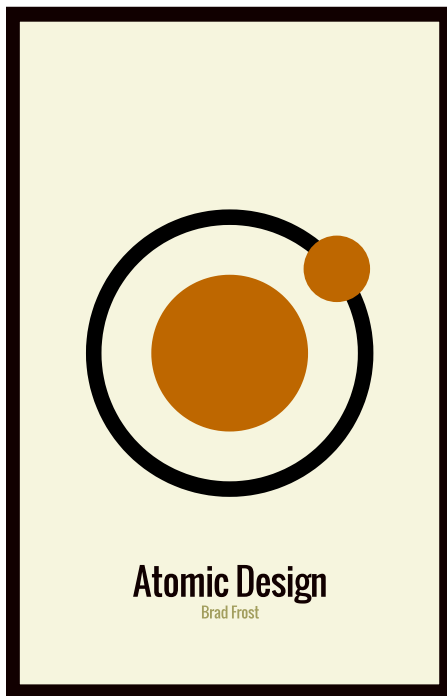
Handover based only on screens leads to information loss.

⇒ How to find a common language?

1

Part 1: Atomic Design

Finding a common language



Atomic Design by Brad Frost

We're tasked with making interfaces for more users in more contexts using more browsers on more devices with more screen sizes and more capabilities than ever before. That's a daunting task indeed. Thankfully, design systems are here to help.

Atomic Design details all that goes into creating and maintaining robust design systems, allowing you to roll out higher quality, more consistent UIs faster than ever before. This book introduces a methodology for thinking of our UIs as thoughtful hierarchies, discusses the qualities of effective pattern libraries, and showcases techniques to transform your team's design and development workflow.

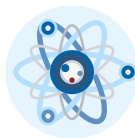
[order the e-book](#)

[read now](#)



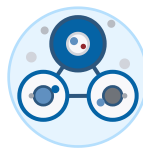
Design Tokens

Colors, typography,
spacing



Atoms

Smallest indivisible
UI elements



Molecules

Groups of atoms
working together



Organisms






Complex compound
components



Templates & Pages

Complete layouts

```
const theme = createTheme({
  palette: {
    primary: { main: '#1976d2' },
    secondary: { main: '#dc004e' },
  },
  typography: {
    fontFamily: 'Inter, sans-serif',
    h1: { fontSize: '2.5rem' },
  },
  spacing: 8, // 8px base unit
});
```

-  **Design Tokens** Colors, typography, spacing
-  **Atoms** Smallest indivisible UI elements
-  **Molecules** Groups of atoms working together
-  **Organisms** Complex compound components
-  **Templates & Pages** Complete layouts

Name

PRIMARY BUTTON



Design Tokens Colors, typography, spacing



Atoms Smallest indivisible UI elements



Molecules Groups of atoms working together



Organisms Complex compound components



Templates & Pages Complete layouts

Email

john@example.com

Please enter your email address

SAVE



Design Tokens Colors, typography, spacing



Atoms Smallest indivisible UI elements



Molecules Groups of atoms working together



Organisms Complex compound components



Templates & Pages Complete layouts

Create Account

Enter your details to get started

Full Name *

Email *

Password *

SIGN UP



Design Tokens Colors, typography, spacing



Atoms Smallest indivisible UI elements



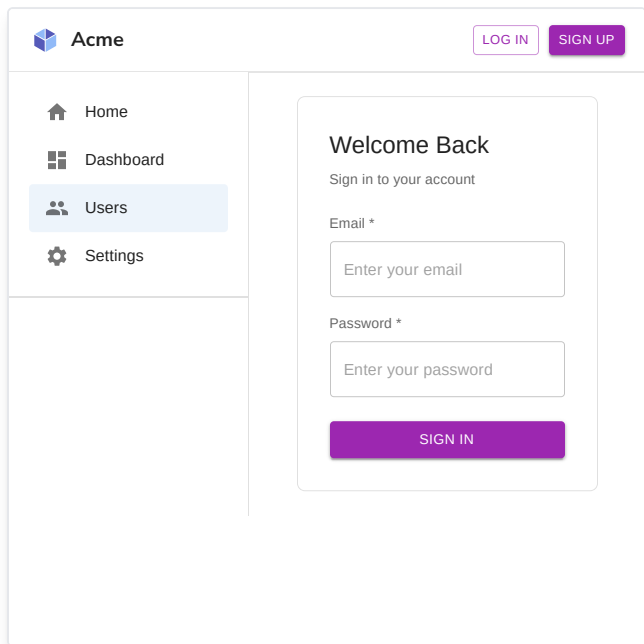
Molecules Groups of atoms working together








Organisms Complex compound components



Templates & Pages Complete layouts



-  **Design Tokens** Colors, typography, spacing
-  **Atoms** Smallest indivisible UI elements
-  **Molecules** Groups of atoms working together
-  **Organisms** Complex compound components
-  **Templates & Pages** Complete layouts

Before: Full-screen handover

"What was that button variant again...?"

- Designers saw: Button variants as building blocks
- Developers saw: Buttons with different styling

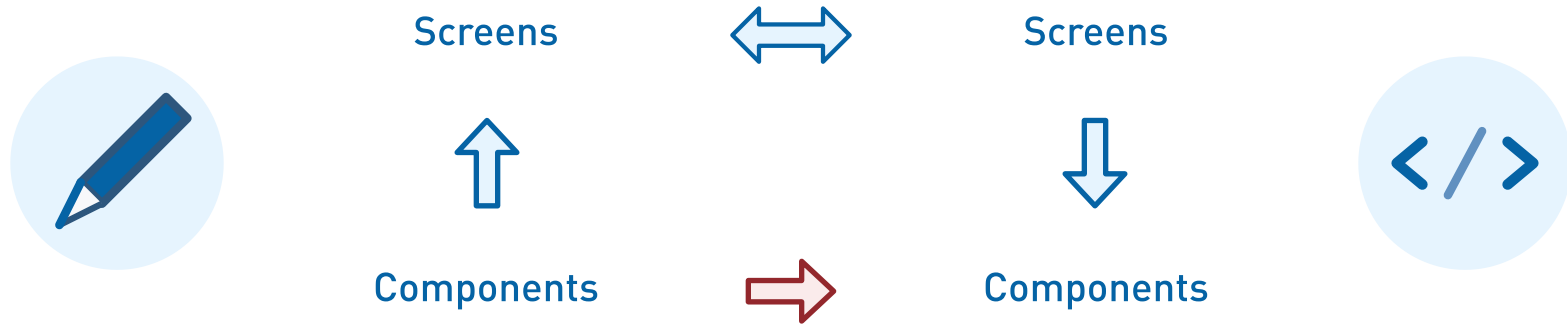
```
<Button variant="contained" color="primary">  
  Primary Button  
</Button>  
  
<Button variant="outlined" color="primary">  
  Secondary Button  
</Button>
```

After: Common design language

Button as an **Atom** — clearly defined!

- Designers and developers have the same hierarchy
- Changes are applied on the same building block level

```
<AcmeButton variant="primary">  
  Primary button  
</AcmeButton>  
  
<AcmeButton variant="secondary">  
  Secondary button  
</AcmeButton>
```



2

Part 2: Storybook.js

Developer-friendly Component Library

— An isolated development environment for UI components

- Visualize every component state
- Interactive playground with controls
- Built for any framework
- Perfect for design systems & collaboration



Storybook



Find components



DESIGN TOKENS

Colors

Typography

ATOMS

Button

Docs

Primary

Secondary

Danger

Old Design



light theme



PRIMARY BUTTON

Controls 4

Actions

Interactions

Visual tests

Accessibility



Name

Control



onClick

-

variant

 primary secondary text

children

Primary Button

disabled

 False True

color

Choose option

Button

TextField

MOLECULES

ButtonGroup

InputField

SplitButton

Docs

Default

Contained Primary

Outlined Secondary

Text

Medium Size

Large Size

Disabled

ORGANISMS



SQUASH AND MERGE

Controls 5

Actions

Interactions

Visual tests

Accessibility

Name

Control

options*

options :

0: "Squash and merge"

1: "Create a merge commit"

2: "Rebase and merge"

]

Edit JSON

onClick

-

variant

 text

>  ButtonGroup


>  InputField

>  SplitButton

∨ ORGANISMS 


>  FormSection

>  Header

>  Sidebar

∨ PAGES 

∨  LoginPage

 Docs

 **Logged Out**

 Logged In

>  Page

    |        light theme  

 **Acme**

LOG IN

SIGN UP

Controls 1

Actions

Interactions

Visual tests

Accessibility  

Name

Control 

onLogin

-

onLogout

-

onCreateAccount

-

user

Set object

Capture all component variants in one place

```
export const Primary: Story = {  
  render: (props) => <Button {... props} />,  
  args: {  
    variant: "primary",  
    children: "Primary Button",  
  },  
};
```

A solid purple rectangular button with the text "PRIMARY BUTTON" in white, uppercase letters.

```
export const Secondary: Story = {  
  render: (props) => <Button {... props} />,  
  args: {  
    variant: "secondary",  
    children: "Secondary Button"  
  },  
};
```

A white rectangular button with a purple border and the text "SECONDARY BUTTON" in purple, uppercase letters.

Explore and customize props interactively

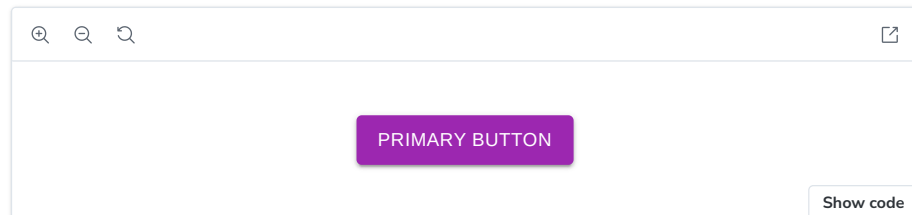
```
export const Primary: Story = {
  render: (props) => <AcmeButton {...props} />,
  args: {
    variant: "primary",
    children: "Primary Button",
    color: undefined,
    disabled: false,
  },
};
```

Button

A flexible button component supporting three variants: primary, secondary, and text.

- primary: Call-to-action button with contained styling (filled). Use at most one primary button per view for the most important action.
- secondary: Outlined button for secondary actions. Can be used multiple times alongside a primary button.
- text: Text-only button with no border or background for low-emphasis actions.

The `color` prop can be used to override the color (e.g. `color={"error"}` for a button warning of a destructive action).



Name	Description	Default	Control	
<code>onClick</code>	function	-	-	

Define your own controls

```
const meta = {
  title: 'Atoms/Button',
  component: AcmeButton,
  argTypes: {
    variant: {
      control: 'inline-radio',
      options: ['primary', 'secondary', 'text'],
    },
    color: {
      control: 'select',
      options: ['primary', 'secondary', 'error', 'info', 'success', 'warning'],
    },
    disabled: {
      control: 'boolean',
    },
  },
} satisfies Meta<typeof AcmeButton>;
```

Actions: Debug Event Handling

```
const meta = {  
  title: 'Molecules/InputField',  
  component: InputField,  
  args: {  
    onChange: fn(),  
    onFocus: fn(),  
    onBlur: fn(),  
  },  
} satisfies Meta<typeof InputField>;
```

The screenshot shows the Storybook interface. On the left is a component catalog with a search bar and a tree view of components. The 'Labeled' component is selected. On the right is the preview area showing a visual representation of the 'Full Name' input field with the text 'John Doe'. Below the preview is a table of actions and props.

Name	Control
onChange	-
onFocus	-
onBlur	-
label*	Full Name
placeholder	John Doe

Rich markdown support in Storybook stories

```
import { Meta, ColorPalette, ColorItem } from '@storybook/addon-docs';
import { createTheme } from '@mui/material/styles';
```

```
<Meta title="Design Tokens/Colors" />
```

```
export const { palette } = createTheme();
```

Color Palette

Our design system uses Material-UI's color palette to ensure visual

Color Reference





```
<ColorPalette>
```

```
  <ColorItem
```

Color Palette

Our design system uses Material-UI's color palette to ensure visual consistency across all components.

Color Reference

Name	Swatches			
Primary Main actions and key interface elements	 <table border="1"> <tr> <td>dark #1565c0</td> <td>main #1976d2</td> <td>light #42a5f5</td> </tr> </table>	dark #1565c0	main #1976d2	light #42a5f5
dark #1565c0	main #1976d2	light #42a5f5		
Secondary Secondary actions and complementary elements	 <table border="1"> <tr> <td>dark #7b1fa2</td> <td>main #9c27b0</td> <td>light #ba68c8</td> </tr> </table>	dark #7b1fa2	main #9c27b0	light #ba68c8
dark #7b1fa2	main #9c27b0	light #ba68c8		
Error Destructive actions and error states	 <table border="1"> <tr> <td>dark #c62828</td> <td>main #d32f2f</td> <td>light #ef5350</td> </tr> </table>	dark #c62828	main #d32f2f	light #ef5350
dark #c62828	main #d32f2f	light #ef5350		
Warning	 <table border="1"> <tr> <td>dark</td> <td>main</td> <td>light</td> </tr> </table>	dark	main	light
dark	main	light		

```
const meta = {
  title: 'Atoms/Button',
  component: AcmeButton,
  tags: ['autodocs']
} satisfies Meta<typeof AcmeButton>;
```

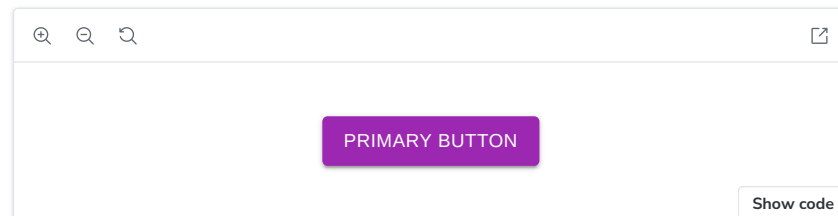
```
/**
 * The primary button is the main call-to-action for
 * a view. It uses the contained style with a solid
 * background color to draw attention. Use sparingly
 * - at most one primary button per section or view
 * to maintain emphasis.
 */
export const Primary: Story = {
  render: (props) => <AcmeButton {...props} />,
};
```

Button

A flexible button component supporting three variants: primary, secondary, and text.

- **primary**: Call-to-action button with contained styling (filled). Use at most one primary button per view for the most important action.
- **secondary**: Outlined button for secondary actions. Can be used multiple times alongside a primary button.
- **text**: Text-only button with no border or background for low-emphasis actions.

The `color` prop can be used to override the color (e.g. `color={"error"}` for a button warning of a destructive action).



Name	Description	Default	Control	
onClick	function	-	-	

- Theming (light/dark mode)
- Accessibility
- Visual testing
- Interaction testing

Discover at

<https://storybook.js.org/addons>

The screenshot displays a web application interface. At the top, there is a navigation bar with several icons: a refresh icon, a zoom in icon, a zoom out icon, a search icon, a grid icon, a photo icon, a document icon, a dashed box icon, a list icon, a close icon, and a 'light theme' toggle. Below the navigation bar, the main content area is divided into two sections. On the left is a sidebar menu for 'Acme' with a blue cube icon. The menu items are: Home (house icon), Dashboard (grid icon), Users (two people icon, highlighted with a light blue background), and Settings (gear icon). On the right is a login form titled 'Welcome Back'. It contains the text 'Sign in to your account', an 'Email *' label, an input field with the placeholder 'Enter your email', a 'Password *' label, and another input field. At the bottom of the page, there is a footer with the text 'Pages/LoginPage/Logged Out' and a small icon.

Join live: How to ship UI with Storybook MCP

Storybook

Docs

Addons

Showcase

Blog

Visual Test [↗]

Enterprise [↗]

89,446

CTRL K

Get Started

Integrations

Add your integration

Integrate your tools with Storybook to connect workflows and unlock advanced features.

React

Component

Svelte

Web-components

Components

Angular

+ 14 more

Categories

Popular

Essential

Code

Data & State

Test

Style

Design

Appearance

Organize

How to install addons

Create an addon

New to Storybook 8

Visual Tests

Catch unexpected visual changes & UI bugs in your stories

1M

Downloads



+ 12

Popular addons

Chromatic

Automate visual testing across browsers. Gather UI feedback. Versioned documentation.

Themes

Storybook Themes addon: Switch between themes from the toolbar

Mock Service Worker

Mock API requests in Storybook with Mock Service Worker.



Add to an existing frontend

Documentation and visual testing for components of a frontend application



Standalone component library


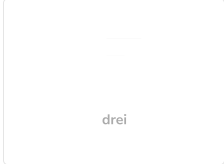
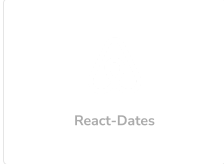







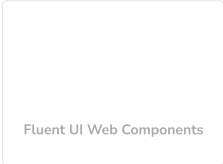
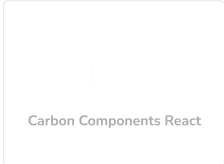
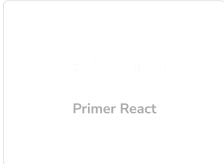
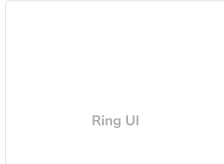






Create a shared component library or design system for your organization

[S Storybook](#)[Docs](#)[Addons](#)[Showcase](#)[Blog](#)[Visual Test [↗]](#)[Enterprise [↗]](#)[🔍 88,935](#)[Get Started](#)[Overview](#)[About](#)[+ Add your project](#)

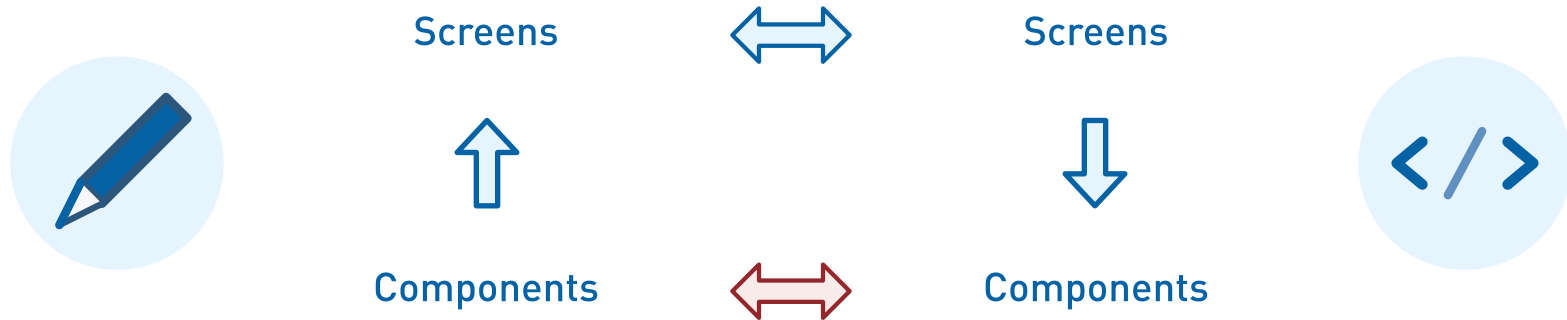
Showcase

Reference Storybook projects from leading UI engineering teams.

[Popular [▼]](#)

 <p>Chakra UI</p>	 <p>drei</p>	 <p>React-Dates</p>	 <p>fundamental-styles</p>	 <p>Grafana UI</p>
 Chakra 55	 pmndrs 69	 AirBnb 12	 SAP 404	 Grafana 87
 <p>Fluent UI Web Components</p>	 <p>Carbon Components React</p>	 <p>Primer React</p>	 <p>Ring UI</p>	 <p>GitLab UI</p>
 Microsoft 38	 Carbon Design System 67	 GitHub 80	 JetBrains 122	 GitLab 100

[Load more projects](#)



⇒ Components developed & reviewed **collaboratively**

Storybook encourages better code organization

- **Component isolation** — Must work independently
- **Prop-driven UI** — Logical presentation layer separation
- **Reusability first** — Components designed as building blocks
- **Explicit contracts** — Props interface = component API

Welcome Back

Sign in to your account

Email *

Password *

SIGN IN

Dependency injection

✗ Coupled to Backend

```
function LoginForm() {
  const [credentials, setCredentials] = useState({});
  const navigate = useNavigate();
  const handleSubmit = async (e) => {
    const res = backendClient.login(credentials);
    localStorage.setItem('token', res.token);
    navigate("/dashboard")
  };

  return (
    <form onSubmit={handleSubmit}>
      <input onChange={...} />
      <input type="password" onChange={...} />
      <button>Login</button>
    </form>
  );
}
```

✓ Prop-Driven Actions

```
function LoginForm({ onLogin }) {
  const [credentials, setCredentials] = useState({});

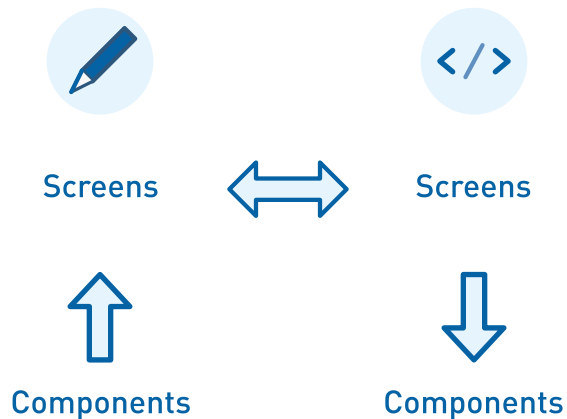
  return (
    <form onSubmit={() => onLogin(credentials)}>
      <input onChange={...} />
      <input type="password" onChange={...} />
      <button>Login</button>
    </form>
  );
}
```

3

Conclusion

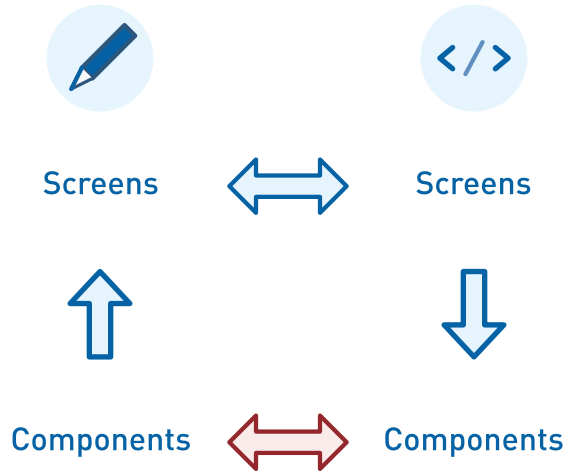
Bringing it all together

Before: Full-Screen handover



- **Designers** create full-screen designs in Figma
- **Developers** implement full-screen designs
- **Designers** review the implementation
- Separate tooling, each side is a black box

After: Collaboration



- **Designers** create Design Tokens, Atoms, ..., Pages
- **Developers** implement these components in Storybook
- Both sides share single source of truth
- Both sides know how the other is working

Beware of Big Up-Front Design

✗ Don't:

- Create design systems without involving developers
- Force a big-bang migration to a new design system
- Create all the components from scratch

✓ Do instead:

- Implement each component as it's designed
- Refactor existing components
- Build on existing component libraries like MaterialUI or ChakraUI

Install Storybook

```
# Automatic setup for your framework  
npm create storybook@latest
```

Start small

- Begin with **atoms**: Buttons, inputs, cards
- Add components you've adapted from libraries
- Create a few stories to understand the workflow
- Add stories for every new or changed component



Build on existing component libraries

Designers often have a Figma component library:

- Design tokens
- Component variants
- Usage guidelines
- Style guides

Use these it as your blueprint:

- Map **Design Tokens** to Theme configuration
- Create **Atoms** from UI primitives
- Build **Molecules** from Figma compositions
- Collaborate on Storybook stories together

Key Takeaways

- **Full-screen handover** loses component hierarchy information.
- **Atomic Design** provides a shared vocabulary for designers and developers.
- **Storybook** enables bi-directional handover on component level.
- **Component isolation** leads to cleaner architecture and allows independent, lightweight testing of components.



Thank you for your attention

Any questions?



Franz Thoma

Principal Consultant
franz.thoma@tngtech.com