Building Design Systems

Goals and Challenges

- With design systems, we want to *enable* product teams to get to consistent results—faster:
  - *One solution* for one component; no duplicates.
  - *Synchronizing* all designers and product teams.
  - *Shared vocabulary* to reduce communication issues.
  - *Easier testing,* debugging and QA on component level.
  - *Faster iterations* with established design patterns.
  - *Future-proof* base for extensions and refinements.

- *One solution* for one component; no duplicates.
- *Synchnizing* all designers and product teams.
- *Shared vocabulary* to reduce communication issues.
- *Easier testing,* debugging and QA on component level.
- *Faster iterations* with established design patterns.
- *Future-proof* base for extensions and refinements.

- The point of a design system is *extensibility*.

  Style guides *increase in value* over time. *(Ideally.)*

- The point of a design system is *extensibility*.
  Style guides *increase in value* over time. *(Ideally.)*

- ...unless they are *difficult to use* or *out-of-date*.
  It happens because they require too much effort.

- The point of a design system is *extensibility*.
  Style guides *increase in value* over time. *(Ideally.)*

- ...unless they are *difficult to use* or *out-of-date*.
  It happens because they require too much effort.

- Let's find a *strategy* for building maintainable and *scalable design systems* for product teams, within a reasonable timeframe.

> In a working design system, the *time savings* at the second use of an existing pattern are *much larger* than the *effort* to introduce the pattern in the first place. The design system won't work if the effort is much bigger, or the pattern library is getting out-of-date very quickly.

— Wolf Brüning, Otto.de

http://www.produktbezogen.de/bauanleitung-pattern-library-1/

- Common challenges with design systems root in *technical or organizational* issues:

  - *Architecting the pattern library.*
    Finding, extracting, grouping and naming modules.

  - *Building the pattern library.*
    A strategy and tooling for building components.

  - *Dealing with maintenance.*
    Keeping all the assets up-to-date (PSD/CSS/PDF).

  - *Team workflow issues.*
    Responsibilities and ownership for updates.

- Common challenges with design systems root in *technical or organizational* issues:

  - *Architecting the pattern library.*
    Finding, extracting, grouping and naming modules.

  - *Building the pattern library.*
    A strategy and tooling for building components.

  - *Dealing with maintenance.*
    Keeping all the assets up-to-date (PSD/CSS/PDF).

  - *Team workflow issues.*
    Responsibilities and ownership for updates.

- Common challenges with design systems root in *technical or organizational* issues:
  - *Architecting the pattern library.*
  - *Building the pattern library.*
  - *Dealing with maintenance.*
  - *Team workflow issues.*
- A common way to organize a design system is by following the *atomic design* approach.

> *Atomic design* doesn't work well. Re-usable components can be used in many *different but similar* ways. It leaves room for interpretation. This opens the door for all kinds of disjointed experiences and makes the system harder to maintain.

— *Karri Saarinen, AirBnB*

http://airbnb.design/co-creating-experiences-with-our-community/

Beyond Atomic Design

- Having a shared understanding of *building blocks* helps, but they need *context* to be used effectively.
  - *Pattern library isn't the end game.* It shines when internal teams use it to extend the product.
  - *Show examples.* The team should know how to apply patterns in appropriate and meaningful ways.
- The context exists on the most concrete levels of atomic design — *applications and features.*

atoms molecules organisms templates pages

APPLICATIONS FEATURES TEMPLATES COMPONENTS BASICS PRINCIPLES

GE's Predix Design System

Applications | Features | Templates | Components | Basics | Principles

Pages | Templates | Organisms | Molecules | Atoms

Atomic Design

Beyond Atomic Design

- Having a shared understanding of *building blocks* helps, but they need *context* to be used effectively.
  - *Pattern library isn't the end game.* It shines when internal teams use it to extend the product.
  - *Show examples.* The team should know how to apply patterns in appropriate and meaningful ways.
- The context exists on the most concrete levels of atomic design— *applications and features.*

- The context exists on the most concrete levels of atomic design— *applications and features.*

    - *Applications* *(accounting, asset management)*

      Case studies documenting applications of the system.

    - *Features* *(analytics, data analysis)*

      Interface screens that allow users to accomplish a task.

- The context exists on the most concrete levels of atomic design— *applications and features.*

  - *Applications* *(accounting, asset management)*
    Case studies documenting applications of the system.

  - *Features* *(analytics, data analysis)*
    Interface screens that allow users to accomplish a task.

  - *Templates* *(dashboard, cards, widgets)*
    Patterns that deal mostly with layout.

  - *Components* *(data tables, range pickers)*
    Patterns that deliver complex interactions.

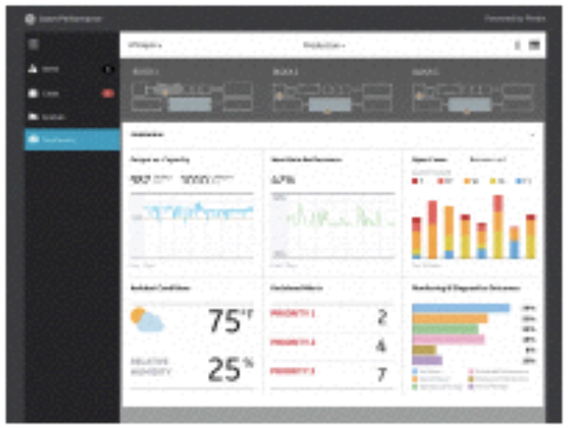
  - *Basics & Principles* *(atoms, animation, truncating text)*
    General, simple interaction patterns and principles.

GE's Predix Design System

Applications | Features | Templates | Components | Basics | Principles

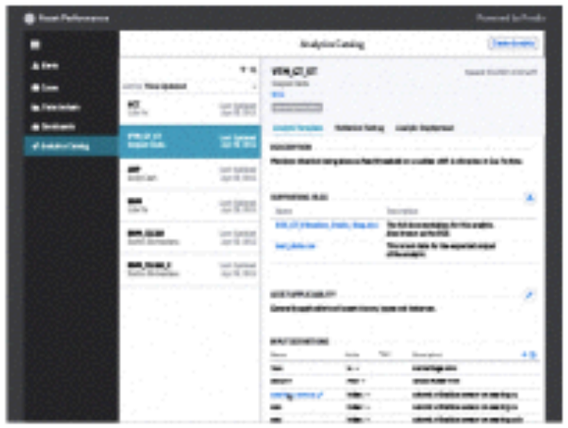
Pages | Templates | Organisms | Molecules | Atoms

Atomic Design



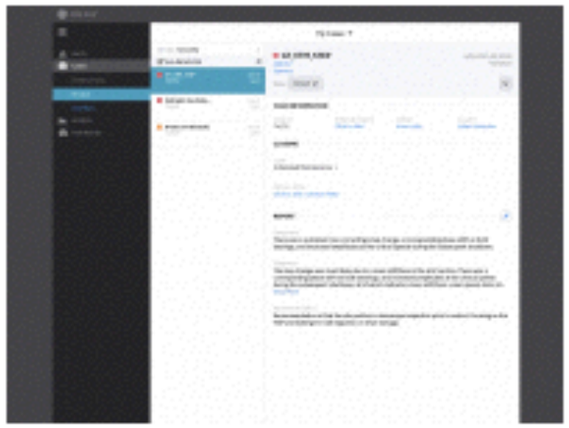
Asset Performance Management

Product Applications



Analytics

Product Features

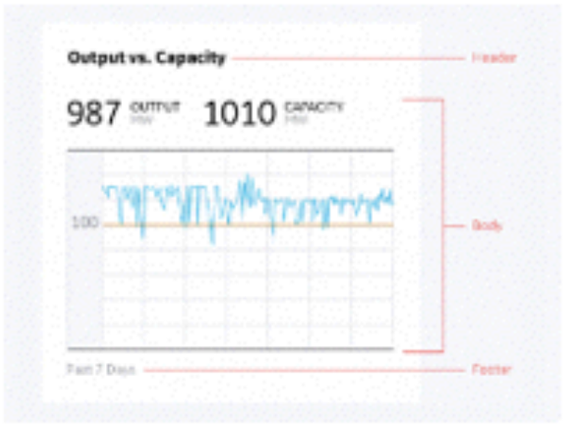


Cases

Product Features

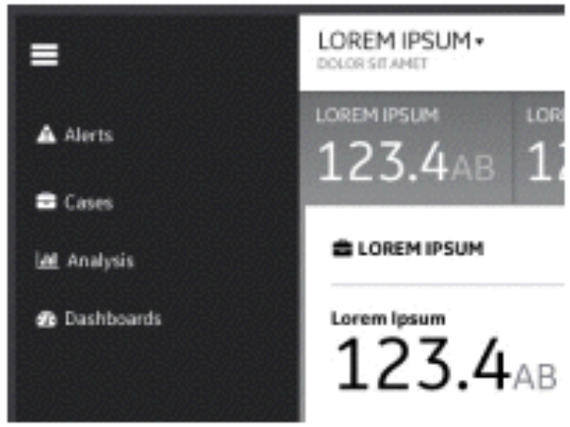
Data Analysis

Product Features



Cards and Widgets

Product Templates

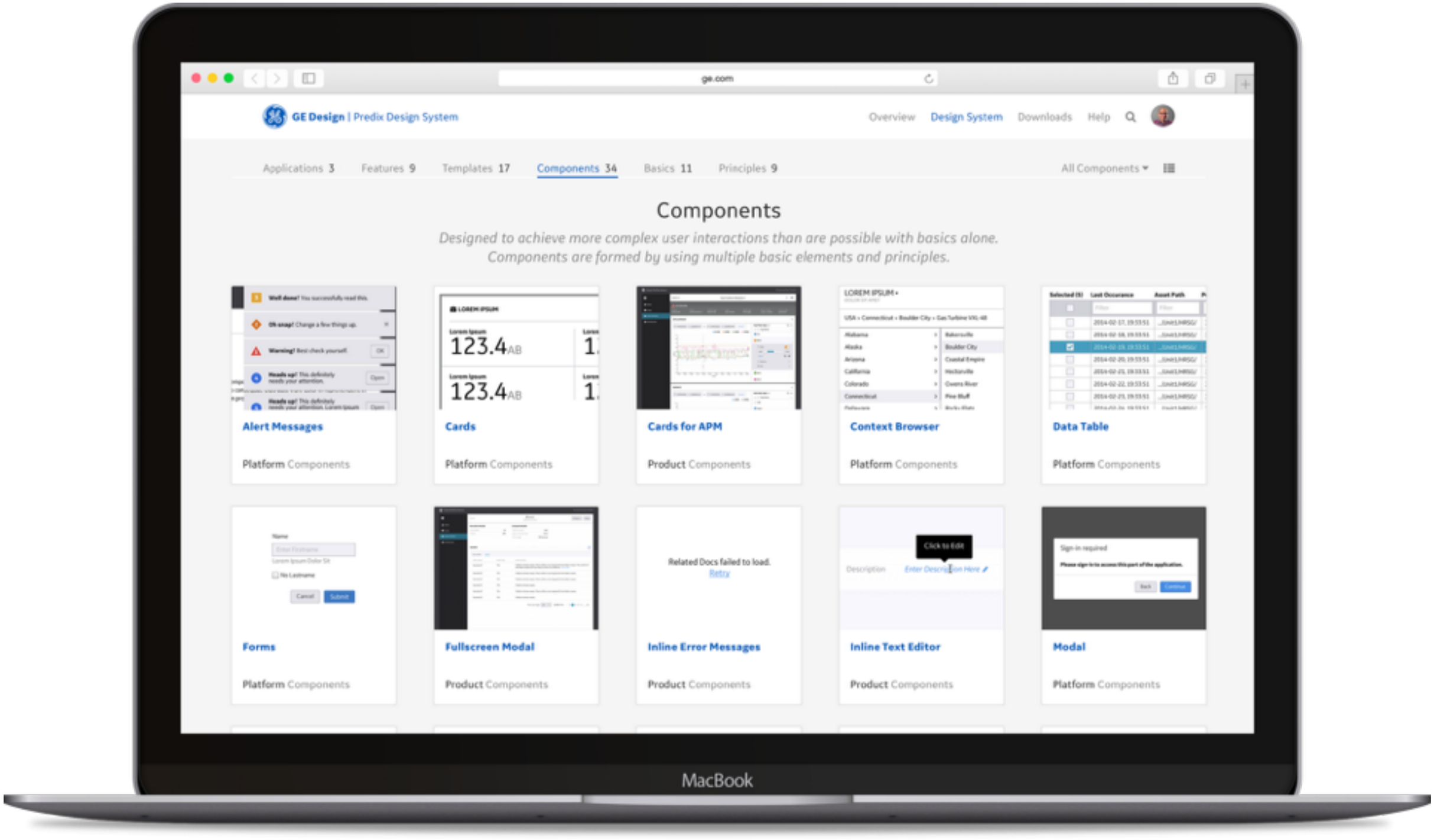


Dashboard

Platform Templates

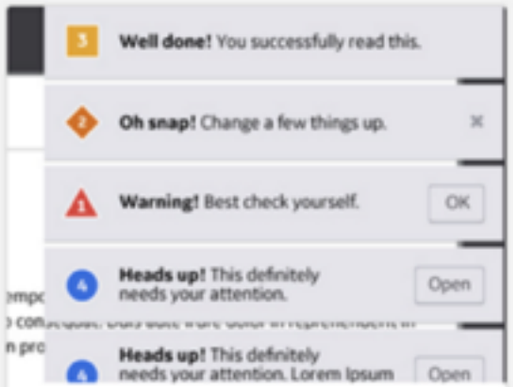
Log-in

Platform Templates


ge.com
GE Design | Predix Design System
Overview
Design System
Downloads
Help
Applications 3
Features 9
Templates 17
Components 34
Basics 11
Principles 9
All Components
Components
Designed to achieve more complex user interactions than are possible with basics alone.
Components are formed by using multiple basic elements and principles.
Alert Messages
Platform Components
Cards
Platform Components
Cards for APM
Product Components
Context Browser
Platform Components
Data Table
Platform Components
Forms
Platform Components
Fullscreen Modal
Product Components
Related Docs failed to load.
Retry
Inline Error Messages
Product Components
Click to Edit
Inline Text Editor
Product Components
Sign-in required
Modal
Platform Components
MacBook

Components

Designed to achieve more complex user interactions than are possible with basics alone. Components are formed by using multiple basic elements and principles.



Alert Messages

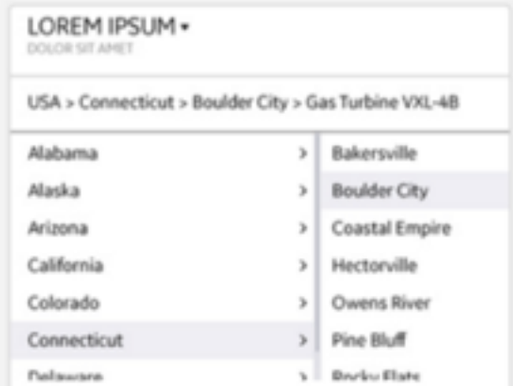
Platform Components

Cards

Platform Components

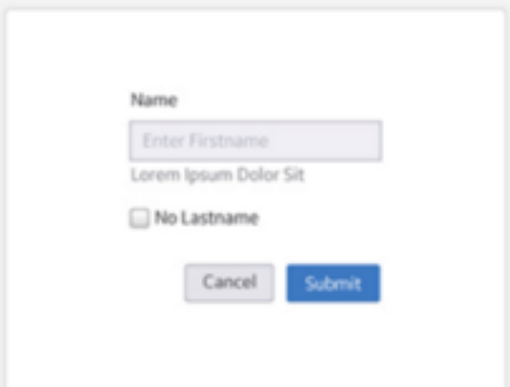
Cards for APM

Product Components



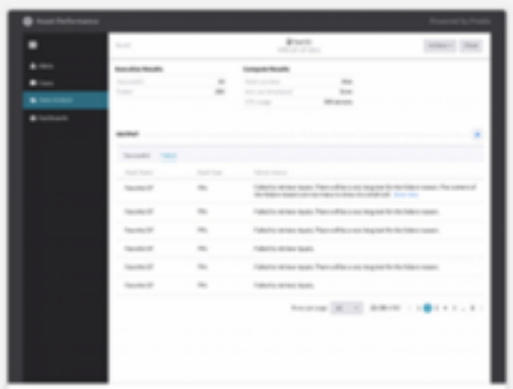
Context Browser

Platform Components



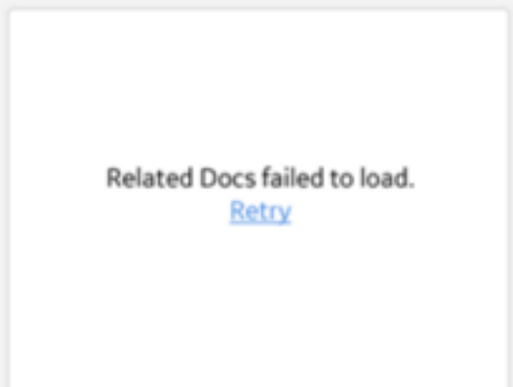
Forms

Platform Components



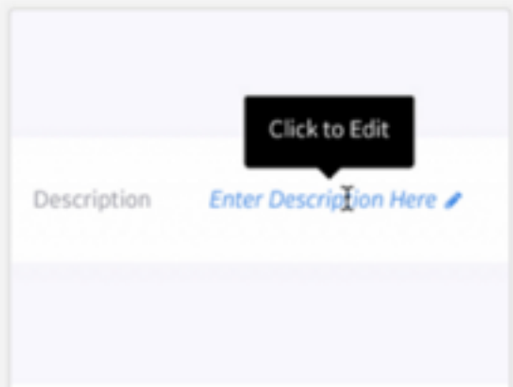
Fullscreen Modal

Product Components



Inline Error Messages

Product Components



Inline Text Editor

Product Components

Architecting The PL

- The general approach for building pattern libraries is *linear and straightforward:*
  - *Audit* the design with an interface inventory (native?),
  - *Cut-up workshop:* identify, group and name modules,
  - *Art boards:* put old/new flows on the wall,
  - *Set up foundation first:* type, colors, icons, spacing, IA.
  - *Organisms:* proceed with advanced components,
  - *Lean PL first:* add only most essential modules first,
  - *Family of modules:* refine relationships, regroup items.
  - *Design assets:* sync Sketch/PSD files via Dropbox.

PARTS

NAME

DATE

VISUAL LANGUAGE
- Color
  - Palettes, by category
  - Contrast
  - Meaning
  - Swatches
- Typography
  - Hierarchy
  - Weights / types
  - Web fonts
  - Baseline grid
- Iconography
  - Pictograms
  - Illustration
  - Icon fonts
  - Alignment
- Motion
  - Principles
  - Timing
  - Types
  - Transitions
- Space
  - Units / measurements
  - Metrics and keylines
  - Structure
- Photography

UI ELEMENTS
- Paragraph
- Lead paragraph
- Block quote
- Headers (H1 - H6)
- Lists
- Links
- Code
- Buttons
  - Primary
  - Secondary
  - Button groups
  - Menu buttons
  - Split buttons
  - FAB (floating action)
- Form controls
  - Text
  - Variants (email, etc)
  - Text area
  - Radio
  - Checkbox
  - Select
- Form controls, cont'd
  - File upload
  - Labels
  - Floating labels
  - Microcopy
  - Required
  - Validation
- Divider / rule
- Switch
- Slider
- Image
  - Block
  - Full bleed
  - Inline with positions

UI COMPONENTS
- Action bar/sheet
- Back to top
- Badges
- Breadcrumbs
- Calendar picker
- Caption
- Cards
- Chat lines
- Code block
- Comments
- Contact us
- Cookie notification
- Data tables
- Dialog
- Drawer/accordion
- Filters
- Footer
- Footnotes
- Header
- Hero billboard
- Hero carousel
- Legend
- Loading / spinner
- Local navigation
- Maps
- Menu
- Messaging
  - Toast
  - Block
  - Inline error
- Modal
- Nav Megamenu
- Pagination
- Progress bar
- Pull quote
- Ratings
- Reviews
- Rich text editor
- Search
- Sidebar
- Site/App navigation
- Social networking
- Status (alpha, beta...)
- Tabs
- Tags / chips / pills
- Thumbnails / # up
- Tips / tutorial
- Toolbar
- Tooltips
- Video & media

UI PATTERNS
- Authentication
- Create account
- Database connection
- Form structure
- Launch
- Permissions
- Purchase/checkout
- Settings
- Site / app structure
- Swipe to refresh

PAGE TEMPLATES
- Home
- Category
- Product
- Dashboard
- Article
- Search results
- Contact us
- Getting started
- Discussion
- Gallery

LAYOUT SYSTEMS
- Grids for sites
- Grids for apps
- Paneling/transitions
- Responsive/breakpoints
- Screen sizes

EDITORIAL
- Voice & tone
- Writing for the web
- Word list
- Capitalization
- Punctuation

BRANDING
- Identity
- Logo(s)
- Taglines(s)
- Partnerships
- Themes

RESEARCH & USERS
- User needs
- Personas
- Research techniques
- Surveys
- A/B, multivar testing

PRACTICES
- Accessibility
- Prototyping
- Performance
- SEO
- Design thinking

CODING STANDARDS
- Style
- Browsers & devices
- Environments
- Prog. enhancement
- Version control

DOWNLOADS
- Templates (PSD...)
- Swatch palettes
- Fonts
- Icons
- Code

GETTING INVOLVED
- Request New Parts
- Give Feedback
- Request Audit
- Present at a Share
- Contribute

ABOUT THE PROGRAM
- Home page
- Principles / pillars
- Getting started
- FAQ
- Version history

ABOUT THE TEAM
- Team
- Recruiting / jobs

UTILITIES
- Variables / tokens
- Mixins
- Helpers
- Customize

OTHER?

PARTS

NAME _____ DATE _____

VISUAL LANGUAGE

- ☐ Color
  - ☐ Palettes, by category
  - ☐ Contrast
  - ☐ Meaning
  - ☐ Swatches
- ☐ Typography
  - ☐ Hierarchy
  - ☐ Weights / types
  - ☐ Web fonts
  - ☐ Baseline grid
- ☐ Iconography
  - ☐ Pictograms
  - ☐ Illustration
  - ☐ Icon fonts
  - ☐ Alignment
- ☐ Motion
  - ☐ Principles
  - ☐ Timing
  - ☐ Types
  - ☐ Transitions
- ☐ Space
  - ☐ Units / measurments
  - ☐ Metrics and keylines
  - ☐ Structure
- ☐ Photography

UI ELEMENTS

- ☐ Paragraph
- ☐ Lead paragraph
- ☐ Block quote
- ☐ Headers (H1 - H6)
- ☐ Lists
- ☐ Links
- ☐ Code
- ☐ Buttons
  - ☐ Primary
  - ☐ Secondary
  - ☐ Button groups
  - ☐ Menu buttons
  - ☐ Split buttons
  - ☐ FAB (floating action)
- ☐ Form controls
  - ☐ Text
  - ☐ Variants (email, etc)
  - ☐ Text area
  - ☐ Radio
  - ☐ Checkbox
  - ☐ Select

Form controls, cont'd

- ☐ File upload
- ☐ Labels
- ☐ Floating labels
- ☐ Microcopy
- ☐ Required
- ☐ Validation
- ☐ Divider / rule
- ☐ Switch
- ☐ Slider
- ☐ Image
  - ☐ Block
  - ☐ Full bleed
  - ☐ Inline with positions

UI COMPONENTS

- ☐ Action bar/sheet
- ☐ Back to top
- ☐ Badges
- ☐ Contact us
- ☐ Cookie notification
- ☐ Data tables
- ☐ Hero carousel
- ☐ Legend
- ☐ Loading / spinner
- ☐ Modal
- ☐ Nav Megamenu
- ☐ Pagination
- ☐ Site/App navigation
- ☐ Social networking
- ☐ Status (alpha, beta,...)

Type

Title 1 · 44/56

Title 2 · 32/36

Title 3 · 24/28

Large · 19/24

Regular · 17/22

Small · 14/18

MICRO 1 · 8/8

A11y Color

Rausch
#FF5A5F
3.05:1

A11y Babu
#00A699
3.03:1

A11y Arches
#FC642D
3.0:1

A11y Hof
#484848
9.14:1

A11y Foggy
#767676
4.54:1

Spacing

8 · tiny

16 · small

24 · base

48 · large

64 · x-large

User Marquee

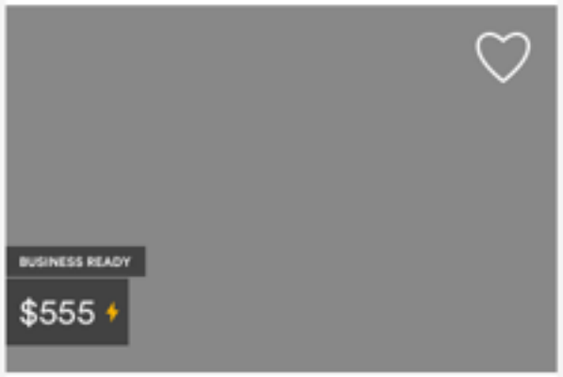
Optional caption

Name
Apr, 2016

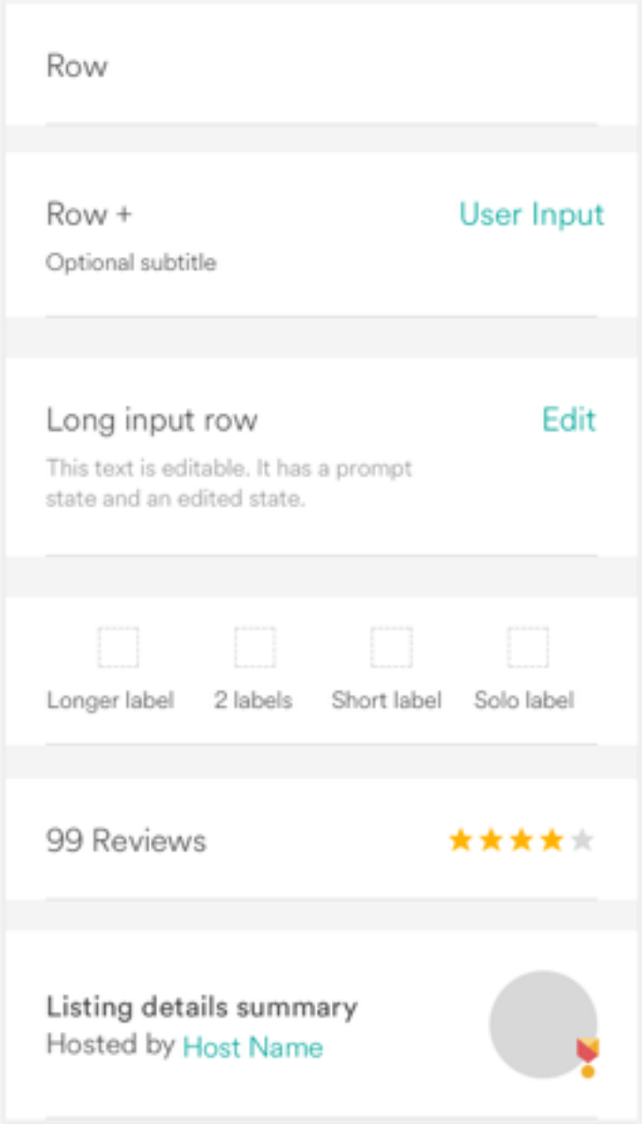
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla consectetur quam dui, interdum varius sem viverra eu. Praesent pellentesque ut ex at eleifend. Praesent neque magna, efficitur eget feugiat a, auctor id leo.

Paragraph two

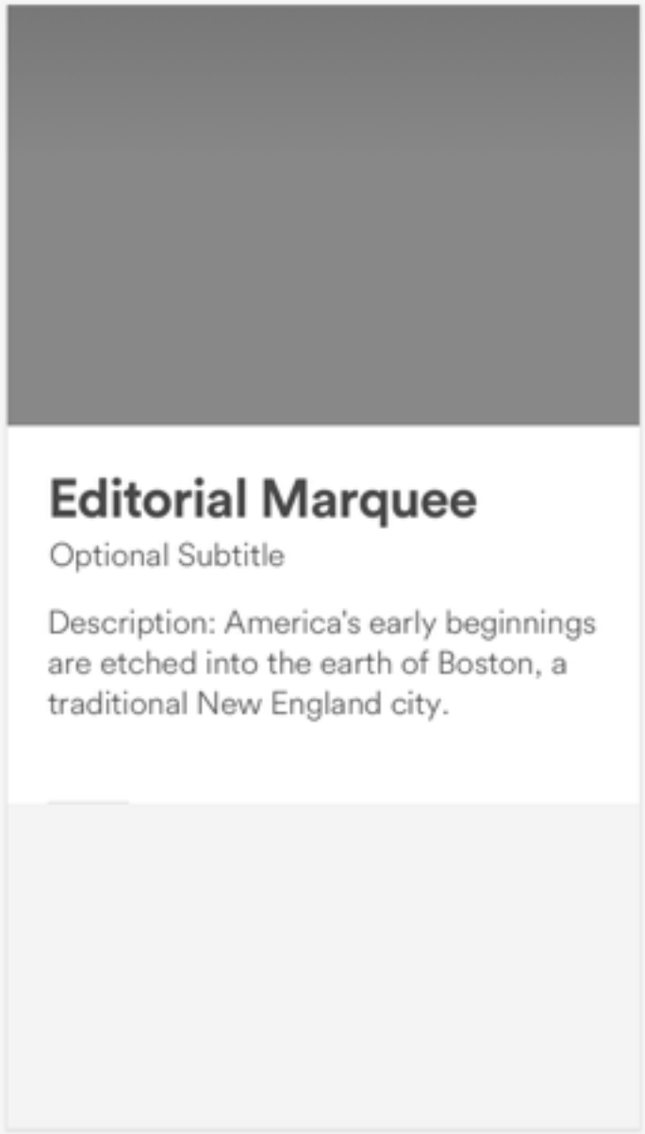
Cards

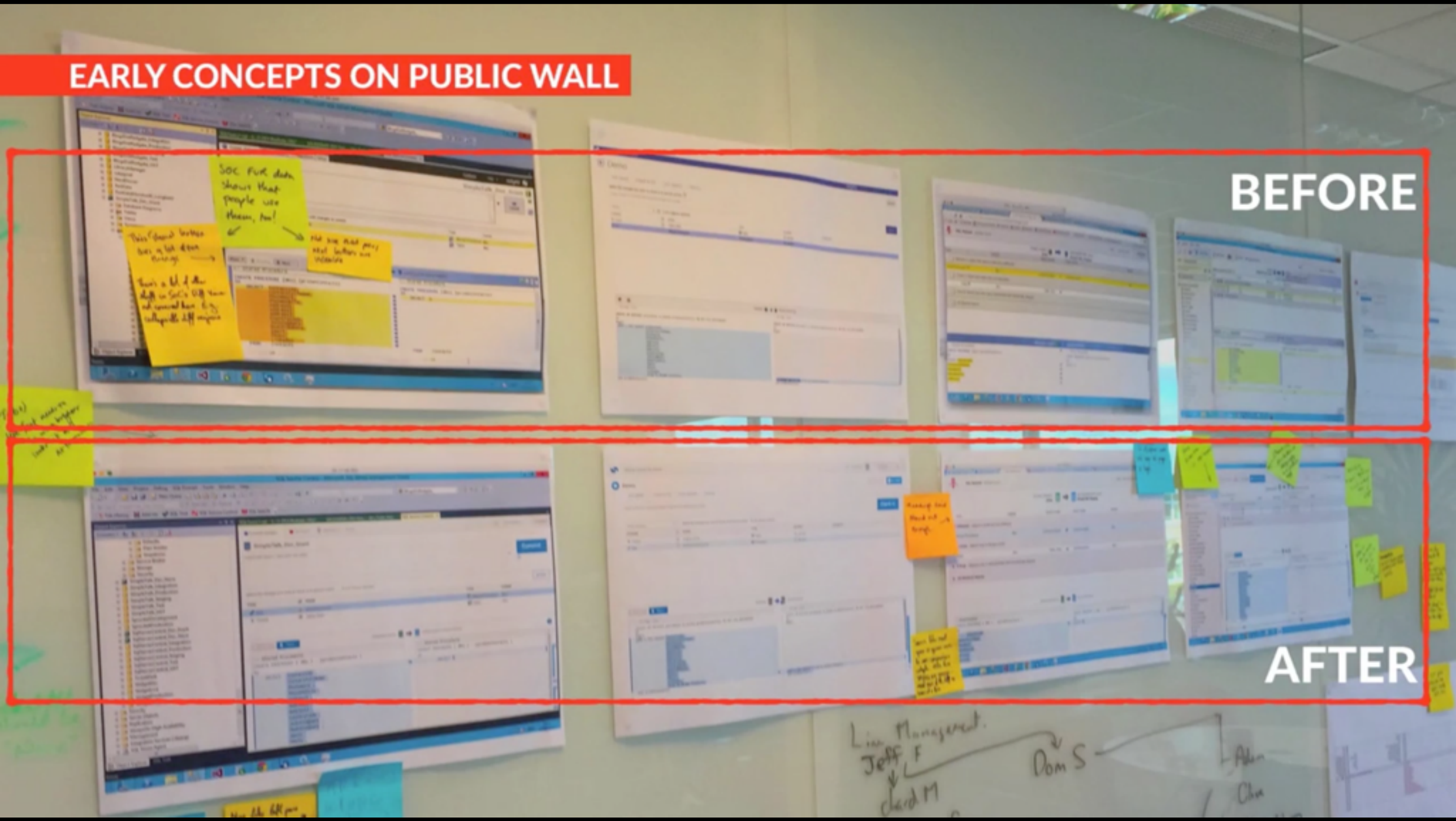


Rows



Editorial Marquee




EARLY CONCEPTS ON PUBLIC WALL
BEFORE
AFTER

DLS Content Components

Content Components

Sect. Headers

Section Headers

Rows

Standard Rows

Tween Rows

Specialty Rows

Text Rows

Messaging

Messag...w Rows

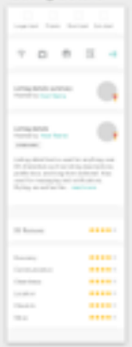
Input Bars

Message Thread

Specialty

Exotics

Listing Details



Listing...view Row

Toastifications

Toasts

Stacking Toasts

DLS Navigation

Navigation

Android

Androi...ide Nav

iOS

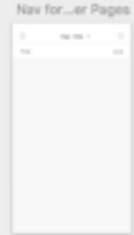
IOS Tab Bar

Pages

Nav for...arquees

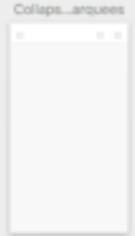
Nav for...arquees

Nav for...er Pages



Collaps...arquees

Collaps...arquees



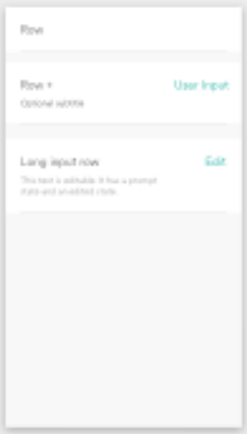
Content Components

Sect. Headers

Section Headers

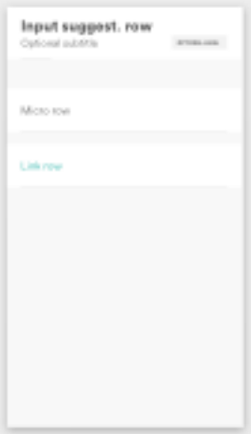
Rows

Standard Rows

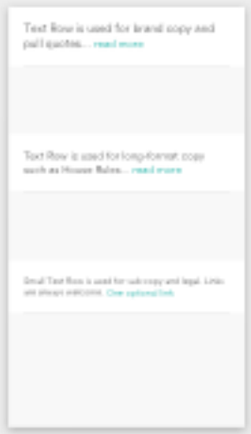


Tween Rows

Specialty Rows

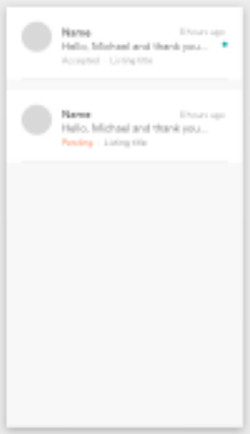


Text Rows



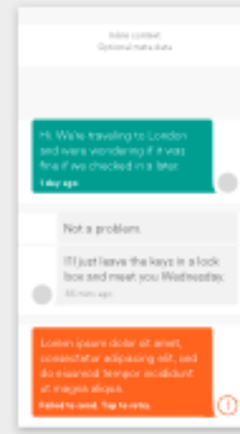
Messaging

Messag...w Rows



Input Bars

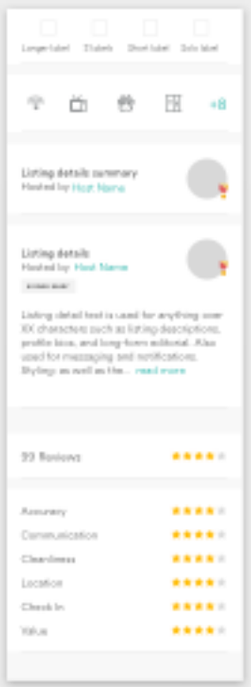
Message Thread



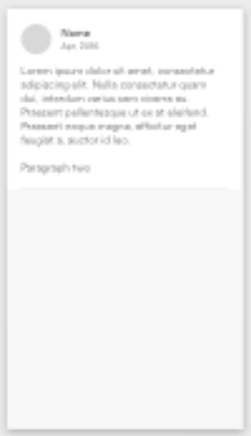
Specialty

Exotics

Listing Details

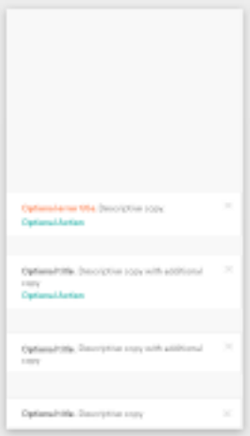


Listing...view Row

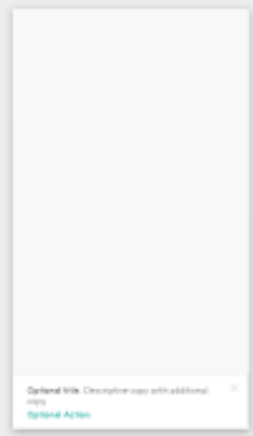


Toastifications

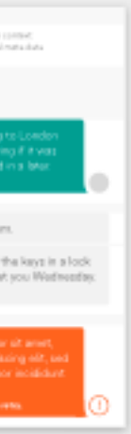
Toasts



Stacking Toasts



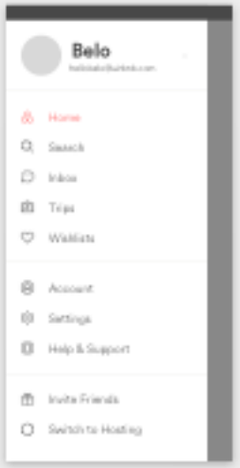
e Thread



Navigation

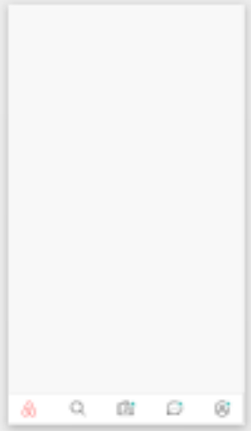
Android

Androi...ide Nav



iOS

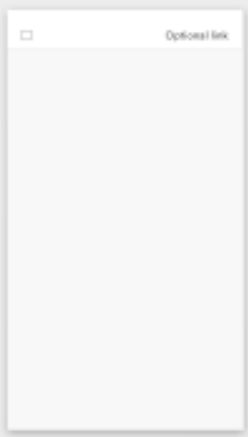
iOS Tab Bar



Pages

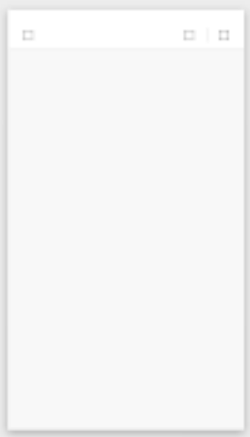
Nav for...arquees

Collaps...arquees

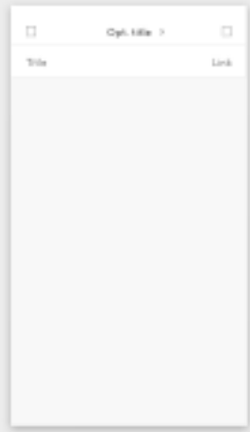


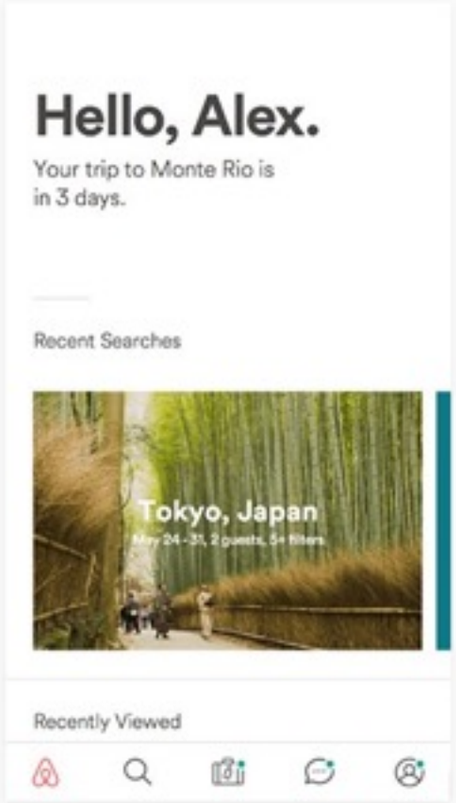
Nav for...arquees

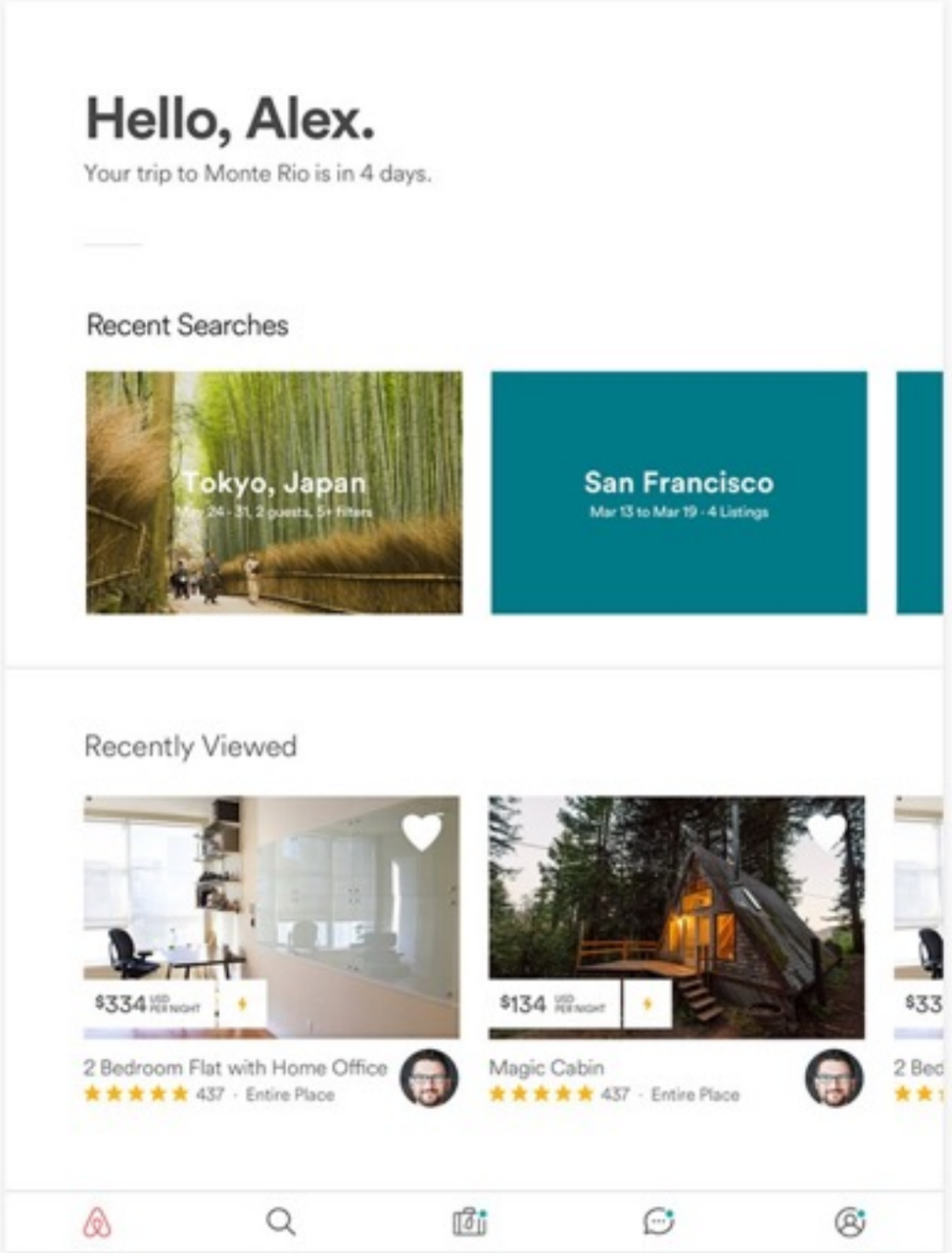
Collaps...arquees



Nav for...er Pages




Hello, Alex.
Your trip to Monte Rio is in 3 days.
Recent Searches
Tokyo, Japan
May 24 - 31, 2 guests, 54 filters
Recently Viewed


Hello, Alex.
Your trip to Monte Rio is in 4 days.
Recent Searches
Tokyo, Japan
May 24 - 31, 2 guests, 54 filters
San Francisco
Mar 13 to Mar 19 - 4 Listings
Recently Viewed
$334 USD PER NIGHT
2 Bedroom Flat with Home Office
★★★★★ 437 · Entire Place
$134 USD PER NIGHT
Magic Cabin
★★★★★ 437 · Entire Place

> We collected components in a master Sketch file. After a week or two we began to see huge leaps in productivity by using the library when iterating on designs...
>
> *— Karri Saarinen, AirBnB*
>
> *http://airbnb.design/co-creating-experiences-with-our-community/*

> ...One day, while putting together a last-minute prototype, our team was able to create nearly 50 screens within just a few hours by using the framework our library provided.

— Karri Saarinen, AirBnB

http://airbnb.design/co-creating-experiences-with-our-community/

Architecting The PL

- The general approach for building pattern libraries is *linear and straightforward:*
  - *Audit* the design with an interface inventory (native?),
  - *Cut-up workshop:* identify, group and name modules,
  - *Art boards:* put old/new flows on the wall,
  - *Set up foundation first:* type, colors, icons, spacing, IA.
  - *Organisms:* proceed with advanced components,
  - *Lean PL first:* add only most essential modules first,
  - *Family of modules:* refine relationships, regroup items.
  - *Design assets:* sync Sketch/PSDs via Github/Dropbox.

Architecting The PL

- The general approach for building pattern libraries is *linear and straightforward:*
- A *design pattern* usually contains a name, context, problem, solutions, related patterns.

- The general approach for building pattern libraries is *linear and straightforward:*

- A *design pattern* usually contains a name, context, problem, solutions, related patterns.

- Usually components are cut out by appearance.

Wouldn’t it be more useful to sort by *function?*

- A *design pattern* usually contains a name, context, problem, solutions, related patterns.

- Usually components are cut out by appearance. Wouldn't it be more useful to sort by *function?*

- Basically it means isolating a component and looking into its *semantics:* function and name.

- Basically it means isolating a component and looking into its *semantics:* function and name.

  - *Names shouldn't be descriptive.*
    E.g. not 'shiny-button', 'big-button', 'red-headline'.

  - *Names shouldn't contain references to context or location.*
    E.g. not 'button-footer', 'sidebar-navigation'.

- *City-block-sizes could define a family of modules.*
  Pick a standard size (100) and define variations of it.

- *Base names on their global function.*
  Focus on the content within the entire system, not page.

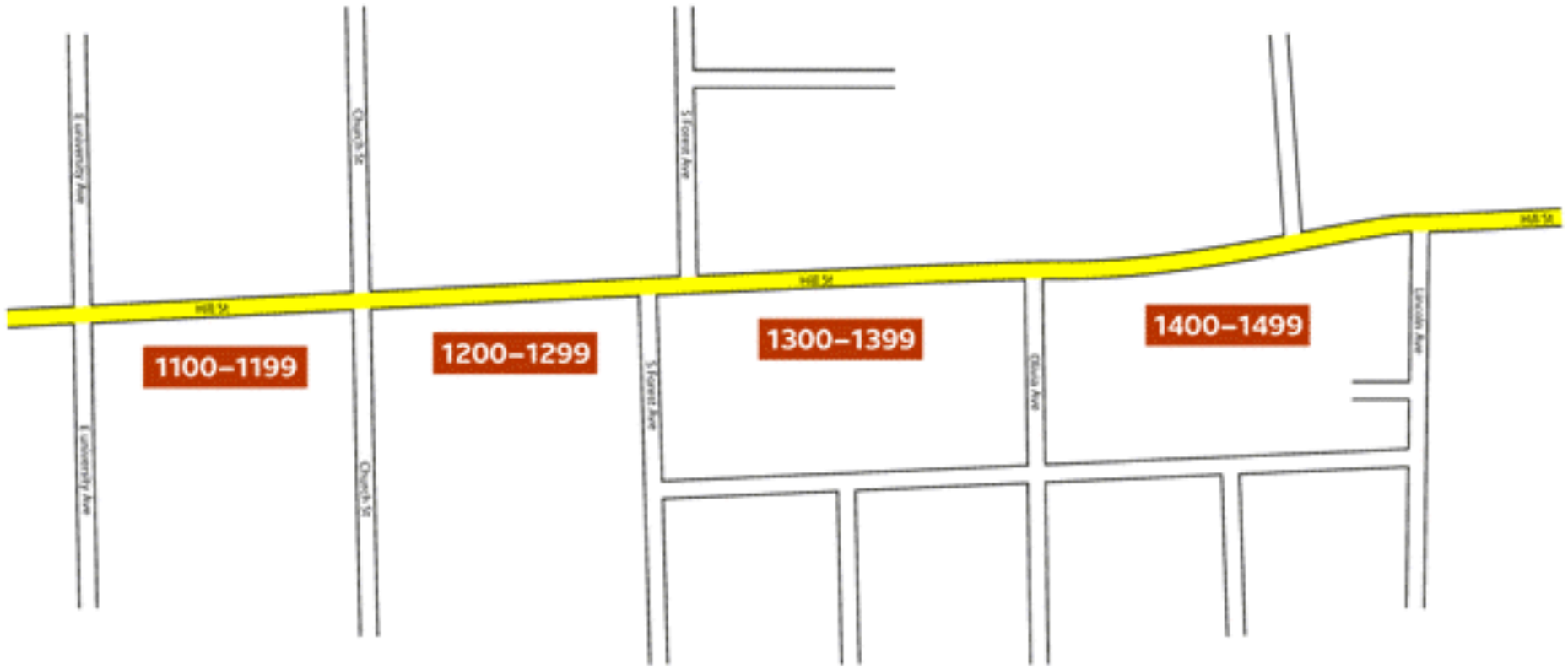
- Basically it means isolating a component and looking into its *semantics:* function and name.

  - *Things that behave the same should look the same.*
    Different things look different. Naming reflects that.

  - *Would component keep the name in various contexts?*
    With very short/very lengthy names, small/large images?

- *Not all components are created equal.*
  Focus on components that repeat often and refine them.

- *No obvious name? Rethink the component entirely.*
  Maybe its function is overloaded or too diluted.


E University Ave
Church St
S Forest Ave
Hill St
Hill St
Hill St
Lincoln Ave
1100–1199
1200–1299
1300–1399
1400–1499
E University Ave
Church St
S Forest Ave
Olivia Ave

Submit

Button50

Submit

Button100

Button150

Submit

Button200

Submit Button 25

Submit Button 50

Submit Button 75

Submit Button 100

Primary

Secondary

Submit

Submit

Button 50

Submit

Submit

Button 100

grey400

black50

black100

red100

red200

Slide 100

Slide 200

★★½☆☆ Rating 100

★★½☆☆ Rating 50

32 34 36 38 40 42 44 46 selTiles 100

32 34 36 38 40 42 44 46 selTiles 50

Fehler! Beim Login ist ein Fehler aufgetreten. Bitte versuchen Sie es zu einem späteren Zeitpunkt erneut. Message 100, Error

Warnung! Ihr Passwort ist wurde schon seit 1 Jahr nicht geändert. Um ihr Konto umfassend zu schützen, sollten Sie dieses umgehend ändern. × Message 100, Warning

Erfolg! Sie haben sich erfolgreich abgemeldet. × Message 100, Success

Lorem ipsum dolor sit amet ^ Accordion 100

Lorem ipsum dolor sit amet, adip iscing elit. Aenean commodo ligula eget dolor. Aenean massa nasce tur ridiculus mus.

Lorem ipsum dolor sit amet ˅

Lorem ipsum dolor sit amet ˅

> It's common to think that every component in *every variation* has to be properly documented but it just bloats up the library. A *pattern* is a solution to a problem that's *repeated* in different contexts (with different content). Headers and footers shouldn't be there, for example.
>
> — *Wolf Brüning, Otto.de*
>
> *http://www.produktbezogen.de/bauanleitung-pattern-library-2-die-perfekte-loesung/*

- Common challenges with design systems root in *technical or organizational* issues:

  - *Architecting the pattern library.*
    Finding, extracting, grouping and naming modules.

  - *Building the pattern library.*
    A strategy and tooling for building components.

  - *Dealing with maintenance.*
    Keeping all the assets up-to-date (PSD/CSS/PDF).

  - *Team workflow issues.*
    Responsibilities and ownership for updates.

Building A Pattern Library

- PLs usually have many *stakeholders:* designers/devs, QA, managers, external agencies, subsidiaries.
- To keep a library *up-to-date,* we need to provide value for everybody involved — in *a single* instance.
- So we have to combine *brand experience guide,* visual design samples and code snippets in one PL.

Building A Pattern Library

- The general approach for building pattern libraries is *linear and straightforward:*
  - *Organize* the code base by encapsulated modules,
  - *Extend* modules with BEM-convention,
  - *Living styleguide:* the PL is generated automatically,
  - *Augment context:* the code is extended with notes,
  - *Slow rollout:* first release has only critical modules,
  - *Same codebase:* CSS/JS are shared by PL and live site,
  - *Pattern Library API:* avoid PL decay with remote teams.

Lightning Design System

Vitaly

https://www.lightningdesignsystem.com/components/process/

salesforce Design System

Search

- Getting Started
- Design
- Components
  - Activity Timeline
  - Badges
  - Breadcrumbs
  - Buttons
  - Button Groups
  - Cards
  - Data Tables
  - Datepickers
  - Docked Composer
  - Feeds
  - Forms
  - Grid System
  - Icons
  - Images
  - Lookups
  - Media Objects

Process

Component Overview

A process component communicates to the user the progress of a particular process.

Sales Path Coach

PROTOTYPE: IN PROGRESS

NOT COMPATIBLE WITH S1 MOBILE

✓ ✓ Unqualified Nurturing Closed

```
<div class="slds-grid">
  <div class="slds-tabs--path" role="application">
    <ul class="slds-tabs--path__nav" role="tablist">
      <li class="slds-tabs--path__item slds-is-complete" role="presentation">
        <a class="slds-tabs--path__link" id="tabs-path-1" aria-controls="content-path-1" aric
          <span class="slds-tabs--path__stage">
            <svg aria-hidden="true" class="slds-icon slds-icon--x-small">
              <use xlink:href="/assets/icons/utility-sprite/svg/symbols.svg#check"></use>
            </svg>
            <span class="slds-assistive-text">Stage Complete</span>
          </span>
          <span class="slds-tabs--path__title">Contacted</span>
        </a>
      </li>
      <li class="slds-tabs--path__item slds-is-complete" role="presentation">
```

VARIANTS & STATES

- Sales Path Coach
  - Without Coaching
  - With Coaching Available
  - Current Stage Coaching Visible
  - With Different Stage Selected
  - Lost
  - Won
- Wizard

salesforce-ux/sass-deprec ×

Vitaly

GitHub, Inc. [US] https://github.com/salesforce-ux/sass-deprecate

This repository Search Pull requests Issues Gist

salesforce-ux / sass-deprecate

Watch 12 Star 138 Fork 3

Code Issues 1 Pull requests 0 Pulse Graphs

Let Sass warn you about the pieces of your UI that are deprecated, providing a clear upgrade path for developers https://salesforce-ux.github.io/sass-deprecate/

64 commits 2 branches 6 releases 3 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

kaelig Remove index.html Latest commit ca7807a 16 days ago

assets	Compile SassDoc	16 days ago
sassdoc	Compile SassDoc	16 days ago
test	Rename variable	16 days ago
.editorconfig	Add editorconfig	5 months ago
.gitignore	Ignore npm logs	16 days ago
.sassdocrc	Add sassdoc	4 months ago
.travis.yml	Test the build in all stable versions of Node	16 days ago
CONTRIBUTING.md	Add contribution guidelines	4 months ago
README.md	Use PNG instead to ensure fonts display correctly	16 days ago
index.scss	Semver -> SemVer	3 months ago
package.json	Remove old file from ignored list	16 days ago
sache.json	Add to Sache	4 months ago

README.md

Building A Pattern Library

- On an enterprise level, it's useful to identify *shared attributes* across all products, both web and native.
- *Design tokens* are shared across products using a JSON object. *Theo* outputs code in all formats:
  - *JSON* > Salesforce's special internal CSS syntax
  - *JSON* > Sass
  - *JSON* > LESS/Stylus
  - *JSON* > iOS
  - *JSON* > Android

salesforce-ux/theo: A set o

Vitaly

GitHub, Inc. [US] https://github.com/salesforce-ux/theo

This repository Search

Pull requests Issues Gist

salesforce-ux / theo

Watch 25 Star 354 Fork 28

Code Issues 3 Pull requests 1 Wiki Pulse Graphs

A set of Gulp plugins for transforming and formatting Design Tokens

215 commits 1 branch 1 release 15 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

aputinski Merge pull request #51 from salesforce-ux/node42 ... Latest commit 9ac1e74 27 days ago

assets	Update dependencies	27 days ago
src	Fix dependency name on case sensitive systems	27 days ago
test	Fix whitespace, typos and markup	28 days ago
.babelrc	Run tests on node 6	27 days ago
.editorconfig	Add editorconfig	28 days ago
.gitignore	Ignore log files	28 days ago
.npmignore	remToPx only converts "rem"	a year ago
.travis.yml	Also test in earlier versions of node	27 days ago
CONTRIBUTING.md	Point to the latest version of the CLA	28 days ago
LICENSE	License	a year ago
README.md	Fix whitespace, typos and markup	27 days ago
package.json	Specify the node engine	27 days ago

README.md

Modularising CSS

- “Because these pieces of code [can] both contain the *same data*, they should both be the *same component*.”
- “Components should be *data-agnostic*. Design a structure/pattern that you can fill with any data.”

readwrite

HACK
The ReadWrite Gift Guide For The DIY Maker

NEWSLETTER
Sign Up To Get ReadWrite In Your Inbox

PLAY

Gear For The DIY Maker: ReadWrite's 2014 Gift Guide

LAUREN ORSINI · NOV 25, 2014

Web Apps, Web Technolog

readwrite.com

readwrite

HACK
The ReadWrite Gift Guide For The DIY Maker

NEWSLETTER
Sign Up To Get ReadWrite In Your Inbox

WEB CLOUD MOBILE SOCIAL START PLAY WORK HACK

The same HTML!

Gear For The DIY Maker: ReadWrite's 2014 Gift Guide

LAUREN ORSINI · NOV 25, 2014

- Common challenges with design systems root in *technical or organizational* issues:
  - *Abstract based on appearance.*
    Don’t build based on content but visual patterns.
  - *Don’t tie into use cases.*
    Don’t include layout or location in component styles.
  - *Abstract any one-off requests into global options.*
    Instead of polluting a family of modules, isolate one-offs.
  - *Team workflow issues.*
    Responsibilities and ownership for updates.

Modularising UIs

Content vs. Context

```
<!-- Incorrect -->
<div class="masthead">
  <h1 class="masthead__title">...</h1>
  <a href="#" class="masthead__btn">...</a>

</div>
```

Modularising UIs

Content vs. Context

```
<!-- Correct -->
<div class="masthead">
  <div class="masthead__content">
    <h1>...</h1>
    <a href="#" class="btn">...</a>
  </div>

</div>
```

Selectors

Namespaces

" I don't know what this does. I don't know where else it's used. I don't know if I can delete it. I don't know if I can change it. I don't know if I can use it elsewhere. I don't know anything!

Selectors

Namespaces

```
.o-object-name {}
.c-component-name {}
.u-utility-name {}
.t-theme-name {}
.is-state {}
._hack {}
.qa-hook {}
.js-javascript-hook {}
```

Selectors

Namespaces

```
<body class="t-light">

  <div class="c-modal  c-modal--wide  jsModal">

    <h1 class="c-modal__header">…</h1>

    <p class="u-text-center">...</p>

  </div>

</body>
```

Selectors

Namespaces

```
.c-modal {
  background-color: white;

  .t-light & {
    background-color: black;
  }

  &.is-open {
    display: block;
  }
```

Working Files

• index.html
main.css

better-ui

_generic.normalize.scss
_generic.reset.scss
_generic.shared.scss
_layer.partial.scss
_objects.box.scss
_objects.flag.scss
_objects.layout.scss
_objects.tables.scss
_settings.colors.scss
_settings.debug.scss
_settings.global.scss
_settings.responsive.scss
_tools.functions.scss
_tools.griddle.scss
_tools.mixins.scss
_tools.responsive.scss
_trumps.clearfix.scss
_trumps.colors.scss
_trumps.debug.scss
_trumps.type.scss
_trumps.widths-responsive.
_trumps.widths.scss
.sass-cache
main.css
main.scss
watch

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, minimum-scale=1.0">

    <title>Better UI - Better Collective CSS Framework and UI Toolkit</title>

    <link rel="stylesheet" href="css/main.css">

    <style>
        html {
            padding: 1.5rem;
            padding-bottom: auto;
        }
    </style>

</head>
<body>

    <div class="o-"
```

- .o-layout--middle
- .o-flag
- .o-flag__minor
- .o-flag__major
- .o-flag--rev
- .o-flag--small
- .o-flag--large
- .o-table--fixed

Working Files

• index.html
main.css

better-ui

_generic.normalize.scss
_generic.reset.scss
_generic.shared.scss
_layer.partial.scss
_objects.box.scss
_objects.flag.scss
_objects.layout.scss
_objects.tables.scss
_settings.colors.scss
_settings.debug.scss
_settings.global.scss
_settings.responsive.scss
_tools.functions.scss
_tools.griddle.scss
_tools.mixins.scss
_tools.responsive.scss
_trumps.clearfix.scss
_trumps.colors.scss
_trumps.debug.scss
_trumps.type.scss
_trumps.widths-responsive.
_trumps.widths.scss
.sass-cache
main.css
main.scss
watch

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, minimum-scale=1.0">

    <title>Better UI - Better Collective CSS Framework and UI Toolkit</title>

    <link rel="stylesheet" href="css/main.css">

    <style>
        html {
            padding: 1.5rem;
            padding-bottom: auto;
        }
    </style>

</head>
<body>

    <h1 class="c-he"
```

.c-heading-ribbon
.c-heading-ribbon--large

• index.html (better-ui) — Brackets

Working Files

- index.html
- main.css

better-ui

- _generic.normalize.scss
- _generic.reset.scss
- _generic.shared.scss
- _layer.partial.scss
- _objects.box.scss
- _objects.flag.scss
- _objects.layout.scss
- _objects.tables.scss
- _settings.colors.scss
- _settings.debug.scss
- _settings.global.scss
- _settings.responsive.scss
- _tools.functions.scss
- _tools.griddle.scss
- _tools.mixins.scss
- _tools.responsive.scss
- _trumps.clearfix.scss
- _trumps.colors.scss
- _trumps.debug.scss
- _trumps.type.scss
- _trumps.widths-responsive.
- _trumps.widths.scss
- .sass-cache
- main.css
- main.scss
- watch

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, minimum-scale=1.0">

    <title>Better UI - Better Collective CSS Framework and UI Toolkit</title>

    <link rel="stylesheet" href="css/main.css">

    <style>
        html {
            padding: 1.5rem;
            padding-bottom: auto;
        }
    </style>

</head>
<body>

    <h1 class="c-heading-ribbon  c-hea"
```

c-heading-ribbon--large