Presentation Notes

¿Qué tratamos de solucionar?

- Complejidad del proyecto
- Gap de comunicación entre ux y dev
- Reutilización de componentes

La idea de esta charla surge principalmente de un problema recurrente que fuimos notando tanto con los equipos de FE como con el equipo de UI, que es el manejo de la complejidad de los componentes de UI en el proyecto, y por otro un gap de comunicación entre lo definido por el equipo de UX y el equipo de desarrollo, así como también el gap de comunicación interna entre devs.

Si bien en teoría, todos deberíamos estar pensando en cómo reutilizar componentes, dentro de un proyecto, ya sea por cuestiones de tiempo o falta de agreement del cliente, muchas veces se hace dificil reutilizar algo de la forma que teníamos pensado (especialmente a medida que avanza la complejidad del proyecto), y esto además consume no solo tiempo de el dev, sino tambien de el ux asignado, al cual tenemos que consultar constantemente sobre cómo implementar features.

¿Cómo encaramos la solución?

- Opción de styleguide estática descartada (mucho esfuerzo de creación y mantenimiento)
- Necesidad de hacer una live styleguide que use exactamente los mismos componentes que estamos usando en la live app.
- Cómo manejamos la complejidad? Por medio del enfoque atomic design.

Hablando con el equipo de UX, primero pensamos en una style guide, que es una práctica común en la industria y teóricamente debería ayudarnos a reducir los problemas mencionados. El problema es que no solo es dificil de hacer en un principio, sino que también requiere mucho trabajo mantenerla sincronizada con lo que realmente estamos haciendo.

De ahí surge la idea de hacer una live style guide que use los mismos componentes que en la app, y de manejar la complejidad de este proceso por medio del enfoque que nos provee atomic design.

¿Cual es el objetivo de esta charla?

- Proveer marco conceptual sobre cómo pensar la ui de una forma que la complejidad sea manejable
- Ver cómo aplicarlo en nuestro flujo de trabajo diario (ahí entran las live style guides)

El objetivo de la charla es darnos tanto un marco conceptual como herramientas concretas y aplicables para hacer más metódico el diseño y desarrollo de interfaces, reduciendo asi la incertidumbre y complejidad de estos procesos, y aumentando la reutilización de componentes. Veremos entonces primero los conceptos presentados en Atomic Design.

Atomic Design

Primero empezamos con Atomic Design. El mismo consiste en dividir el proceso de diseño en niveles de creciente complejidad, tomando como analogía cómo se compone la materia en la naturaleza. No define ninguna implementación específica de esta conceptualización, sólo nos da un marco para pensarlo.

Nivel Atomico

Es el nivel más bajo, indivisible y de menor complejidad de todos.
Puede ser un tag como label, input o un botón. En este nivel también se pueden incluir cosas más abstractas como la paleta de colores a usar.

Nivel Molecular

Combina elementos del nivel atómico para crear elementos más complejos, como un search box que combina un input, un label y un botón.

Nivel Organismo

Acá ya vamos llegando a algo que utilizariamos en nuestra app. Un organismo es una sección de nuestra app, como un header, un sidenav, un footer, etc.

Nivel Template

Acá rompemos la analogía con el mundo de la química para ir a algo muy cercano a lo que vamos a utilizar. La idea del template es juntar organismos en un layout que luego se usará para una vista en particular. Un ejemplo de un template puede ser uno para la visualización de resultados de una búsqueda.

Nivel Página

Es una instancia en particular de un template, por ejemplo la visualización de resultados de la búsqueda de materiales.

¿Tengo que utilizar todo?

No. Si bien el modelo de capas tiene sentido como marco teórico, cualquier framework sólo tiene sentido si es aplicable a nuestro trabajo.

Ninguno de estos pasos es mandatorio, pero sí ayudan a entender cómo disminuir la complejidad yendo de componentes muy simples al resultado final. De hecho al implementarlo van a notar que el nivel template es algo que no desarrollan formalmente, sino que les ayuda a pensar cómo pasar del organismo a la página que hacen.

Por ejemplo, si estamos utilizando angular material, muchos de los “átomos” ya estan dados por el framework, y no necesitamos desarrollarlos.

Style Guide

- Que es una styleguide?
- Es buena como input inicial, pero cuesta mucho mantenerla actualizada

Es por eso que preferimos tener una live style guide.

¿Qué es una live style guide?

- Una o varias paginas dentro de nuestro proyecto, que utilizan las mismas librerías y estilos que el resto del proyecto.
- Ver ejemplos

Proceso iterativo

1. Discover the New App Features

Este punto se realiza como lo venimos haciendo hasta ahora, pero a medida que avanza el proyecto, podemos utilizar componentes de las etapas siguientes para ofrecer al cliente soluciones ya creadas.

2. Abstract Into Components

Ya existen los componentes que quiero usar? Puedo reutilizar alguno existente o extenderlo?

2.1 Understanding Stories

Me permite bajar a componentes de desarrollo las historias acordadas con el cliente

2.2 Estimación

Al usar componentes que ya conocemos, podemos estimar mejor

3. Implement

En esta etapa lo implementamos en la styleguide, utilizando componentes de menor complejidad ya existentes (o recientemente creados).

4. Plug and play

En esta etapa hacemos la implementación puntual del componente. Por supuesto que al hacerlo pueden surgir más cosas que tengamos que refinar, o más insights para agregar al componente abstracto. Esto es parte del proceso iterativo.

Preguntas que surgen...

¿Cómo combino atomic design con SGDD?

La necesidad de abstraer componentes requerida en la live style guide coincide con la complejidad en capas del atomic design.

¿Tengo que usar todo junto?

No necesariamente, atomic design es una forma de reducir la complejidad de los componentes, mientras que las live style guides son una herramienta para comunicarse mejor con los stakeholders. Juntas funcionan bien pero pueden usarse por separado.

¿Cuánto más voy a tener que trabajar?

Si bien a priori parece que tuviéramos que trabajar más, la realidad es que mucho de este trabajo ya lo estamos haciendo en la actualidad, sólo que lo estamos haciendo de forma puntual muchas veces.

¿Cómo cuadra esto con los intereses del cliente?

Si bien puede parecer que el cliente siempre quiere las cosas a su manera, tener la posibilidad de mostrarle componentes exactamente de la forma que se van a ver en la app terminada, nos ayuda a venderle mejor la forma en que tenemos pensado implementar una feature. De hecho, tener componentes que se re-usan en toda la aplicación ayuda al cliente a sentirse más familiarizado con la misma, lo que por lo general resulta en clientes más felices.

Resources

http://blog.bitovi.com/style-guide-driven-development/
https://designschool.canva.com/blog/50-meticulous-style-guides-every-startup-see-launching/
https://designschool.canva.com/blog/apple-google-starbucks-inside-the-web-design-style-guides-of-10-famous-companies/
http://webdesignledger.com/29-online-style-guides/
http://styleguides.io/