

Building User Interfaces

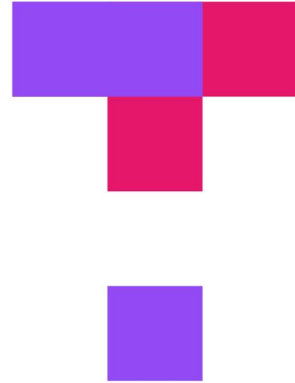
# **Design Patterns and Design Languages**

Professor Bilge Mutlu

# What we will learn today?

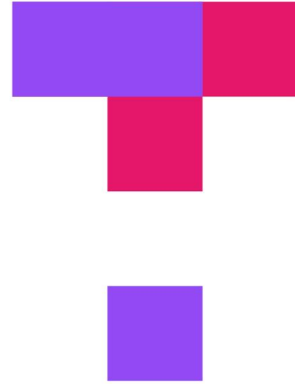
- >> Design Patterns
- >> Design Languages
- >> Assignment Preview

# TopHat Attendance



**TOP HAT**

# TopHat Questions



**TOP HAT**



# Design Patterns

# Recap: Design Patterns

**Definition:** A design pattern is a general, reusable solution to a commonly occurring problem within a given context.

Originally developed by Christopher Alexander (1977; *Pattern Language*) to address problems in architecture and city planning.<sup>1</sup>

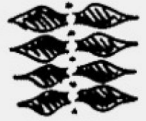
*came from architecture - think about city planning patterns*



2. Strong Centers



3. Boundaries



4. Alternating Repetition



7. Local Symmetries



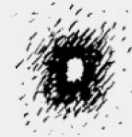
8. Deep Interlock



9. Contrast



12. Echoes



13. The Void



14. Inner calm

<sup>1</sup>Smart Cities Dive

# Recap: Design Patterns in UX

In the last decade, designers have also developed and refined patterns for overall structure and organization, components and controls.<sup>2</sup>

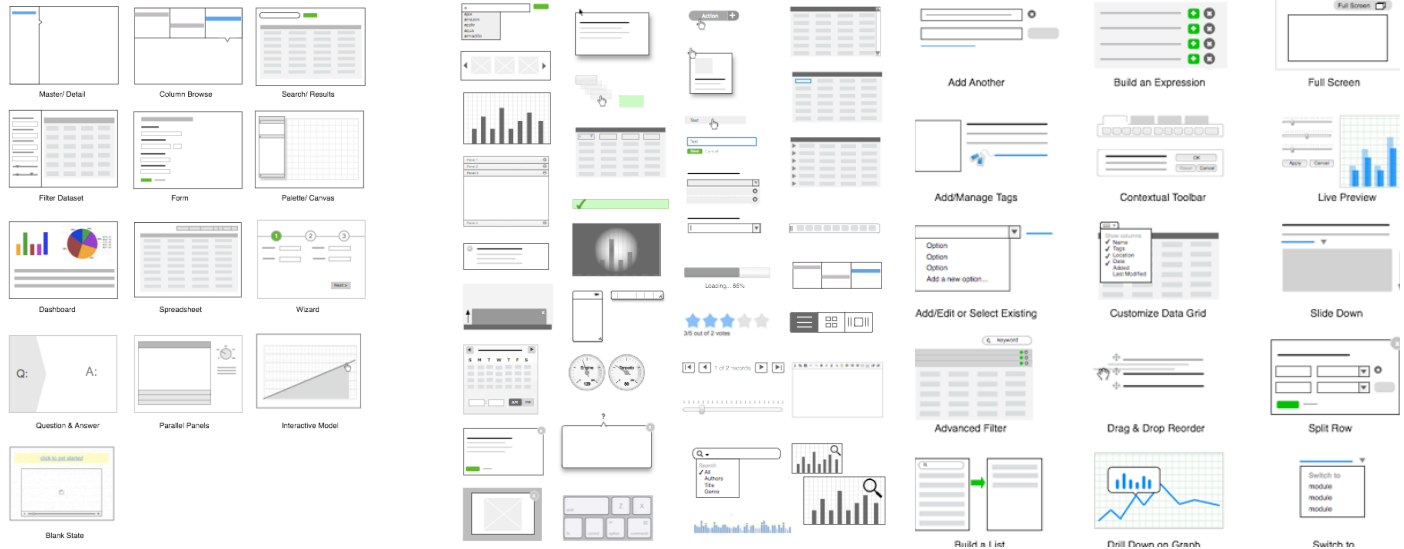
*Problems tend to follow patterns → use a standardized set of solutions*

*Common in email / message apps*



<sup>2</sup> Neil, 2010, 12 Standard Screen Patterns

# Source<sup>3</sup>




<sup>3</sup>Neil, 2010, 12 Standard Screen Patterns

# The Problem with Patterns

**Problem 1.** Can I piece together different patterns to make a complete design? **No**, as this eclectic design would lack coherence.

**Problem 2.** How do I choose which pattern to use? Are patterns interchangeable? **No**, there has to be a *principle* to the selection of patterns.

Choices must be  
thoughtful!



# Enter Pattern Languages

**Define:** A complete and hierarchical collection of patterns for a family of design problems.

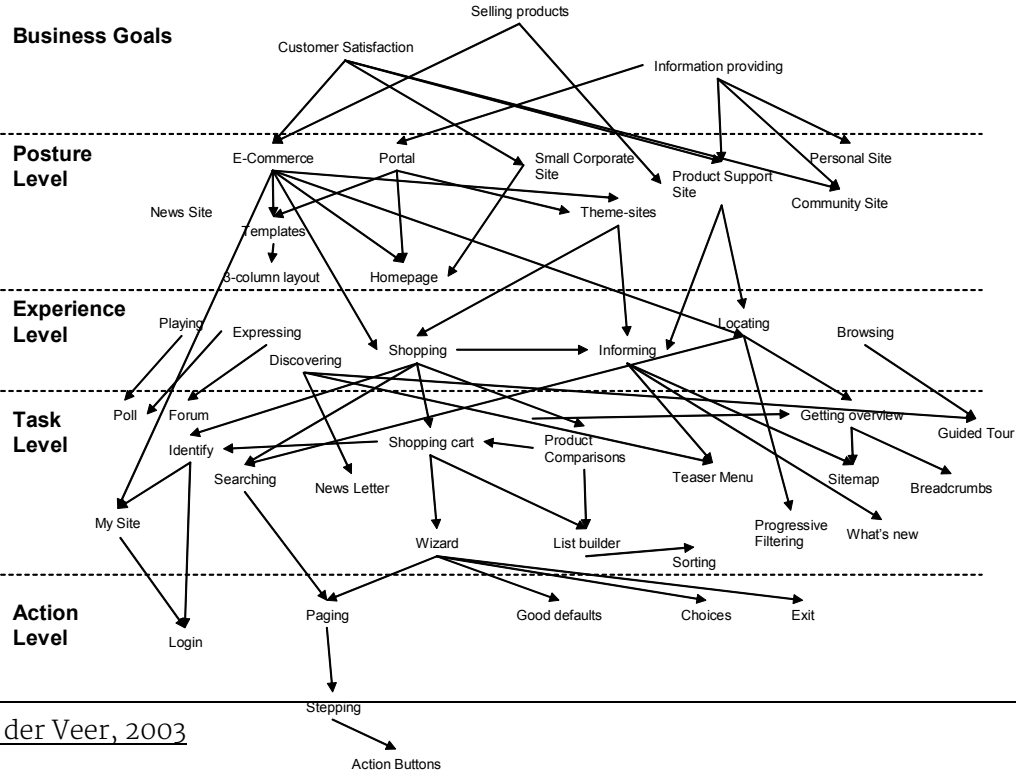
Patterns are *words* (e.g., a component) that are connected with grammar rules to make *sentences* (e.g., a screen) and eventually *language* (e.g., user experience).<sup>4</sup>

The pattern language can be thought of as **patterns being applied at different levels**. Let's see an example.

---

<sup>4</sup>Kruschitz & Hitz, 2009

Source<sup>5</sup>



<sup>5</sup>van Welie & van der Veer, 2003

# Business Goals

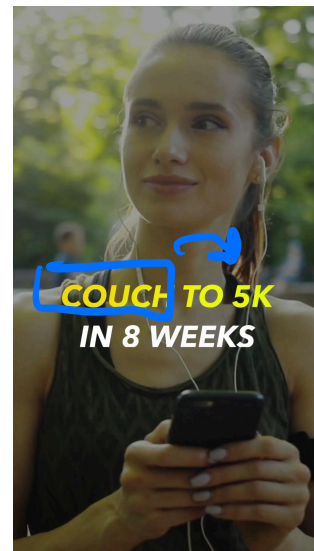
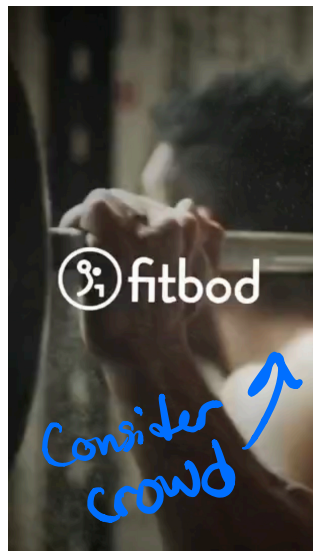
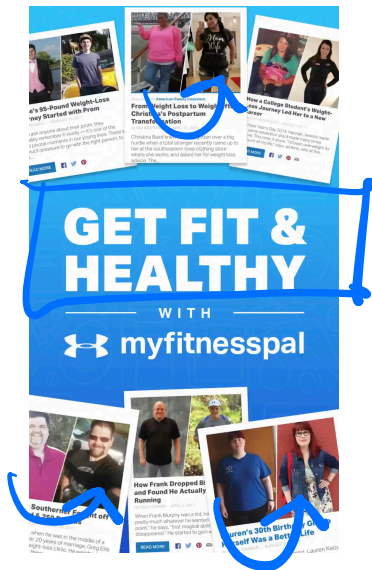
**Definition:** Conceptual design that captures the role that the design plays in user's life, i.e., the *mission* of the application, e.g., "helping users achieve fitness goals."

What do we want the  
user to be able to do?



Source<sup>6</sup>

Some examples



<sup>6</sup>Image sources (iOS App Store): myfitnesspal, fitbod, 5K runner

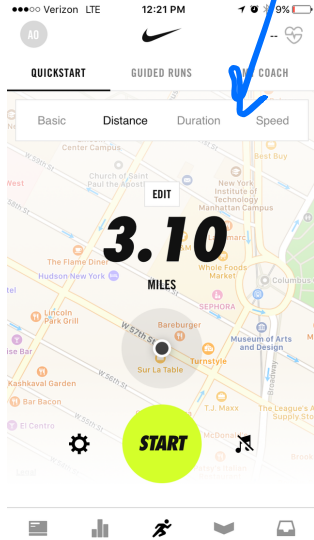
# Posture-Level Patterns

**Definition:** The *structure* that an application follows, i.e., what *type* of application it is, e.g., "a calorie tracking app," "a a step counter app," or "a life coaching app."

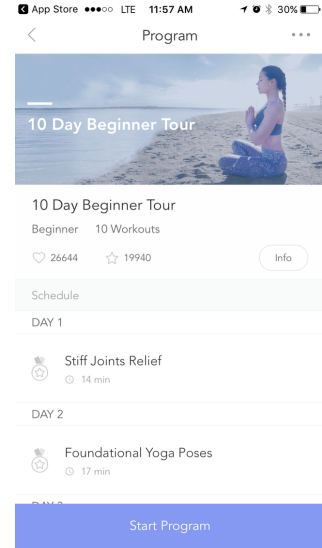
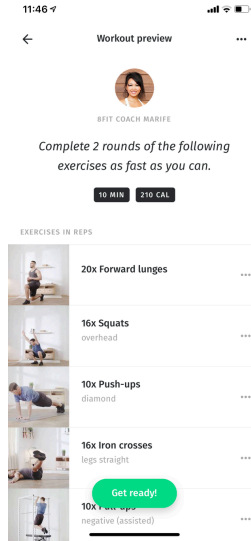
What kind of a system  
is this?

Source<sup>7</sup>

activity /  
run  
tracker



activity  
provider



<sup>7</sup> Source for images

# Elements of a Posture-level Pattern

Once we determine the posture of an application, it gives us guidance on:

- >> Structure
- >> Components
- >> User experience
- >> Alternatives/competitors

**Structure:** Central canvas with supporting panels<sup>8</sup>

**Components:** Canvas, dashboard, score panel, data summary

**UX:** Measurement during the activity, review later

**Competitors:** Strava, RunKeeper



<sup>8</sup> Image source

# Experience-Level Patterns

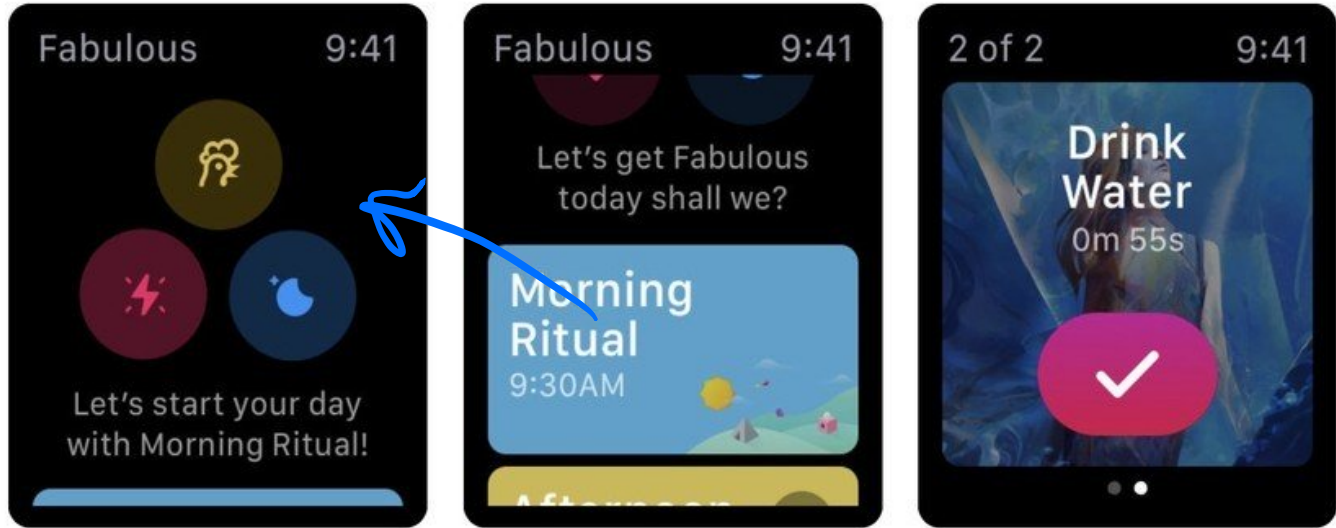
**Definition:** The *user goals* that make up the *user experience* that the application supports, e.g., activity tracking, coaching, and reviewing.

Experience-level patterns can also capture the *quality* of the user experience, e.g., *motivational* coaching.

what activities  
will they be doing?

# Motivational app

Source<sup>9</sup>



<sup>9</sup>[Image source](#)

## Elements of an Experience-Level Pattern<sup>10</sup>

- » Primary goals, e.g., activity tracking
- » Secondary goals, e.g., community building



*social (secondary)*



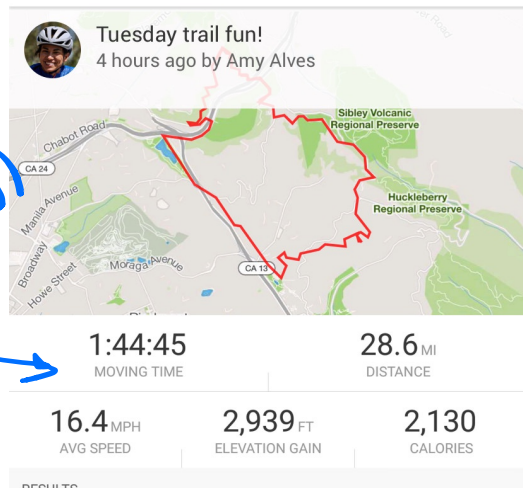
Amy Alves tagged you in this activity.

Accept to save it to your profile and show on your activity feed.  
You can also edit it to give it a custom name and add photos.

Ignore

Accept

*activity tracking (primary)*



<sup>10</sup>[Image source](#)



so far, mostly conceptual. This begins the more concrete levels

## Task-Level Patterns

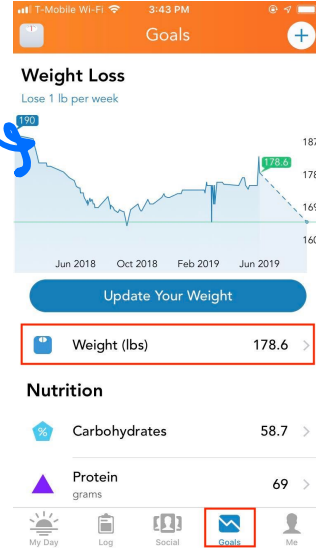
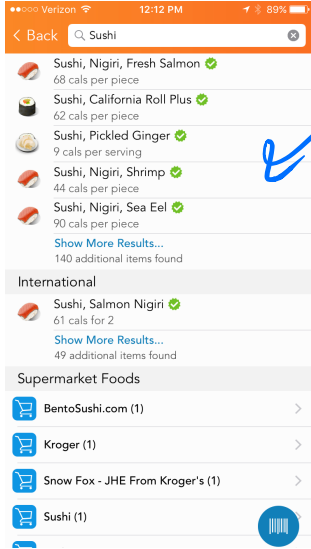
**Definition:** Design solutions that help users accomplish sequences of actions that make up user tasks, e.g., logging a meal, capturing a run, or completing a workout.

Tasks point to specific application components. E.g., meal logging can be done through a "search-and-filter" component, activity tracking can be done through a "scoreboard" component.

Source<sup>11</sup>

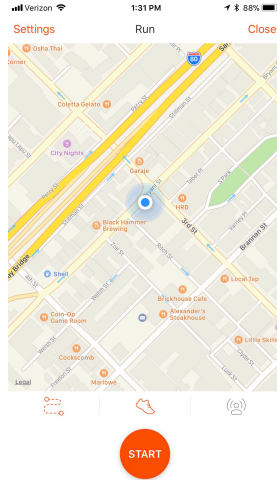
Meal logging

activity tracking

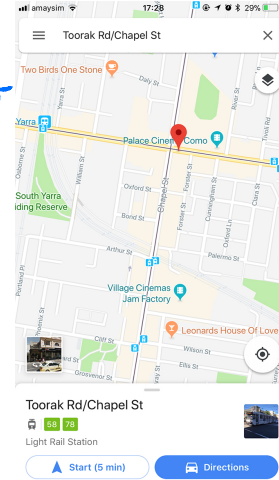


<sup>11</sup> Image sources: [left](#), [right](#)

Task-level patterns can be domain independent. Business goals and posture-level patterns set the context for these patterns.<sup>12</sup>



*Both use maps, but for a different purpose (set by more conceptual levels above)*



<sup>12</sup>Image sources: [left](#), [right](#)

# Action-Level Patterns

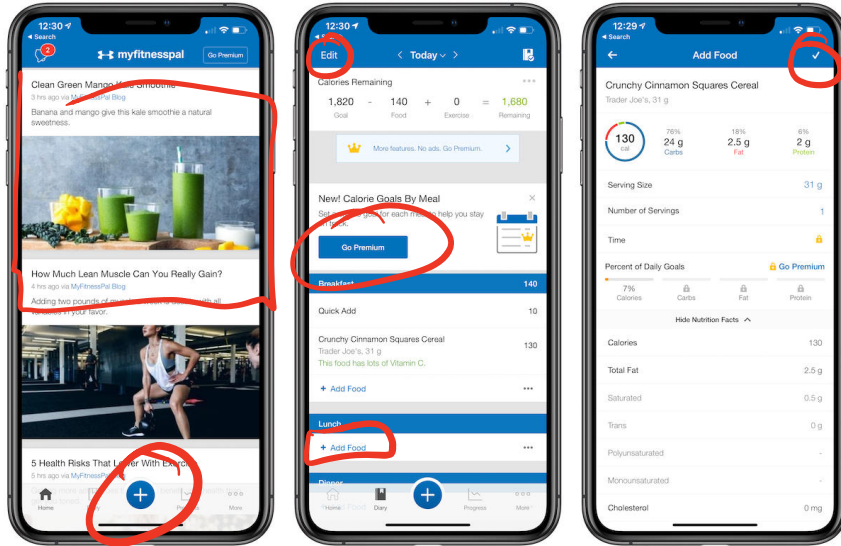
**Definition:** Design solutions that support the actions taken to complete the steps(s) of the user's task, e.g., a "start" button to initiate activity tracking, a selectable list entry for a food item.

Action-level patterns are the lowest level of building blocks for a design. They are often called *widgets* or *components* (as in React).

Low level  
components

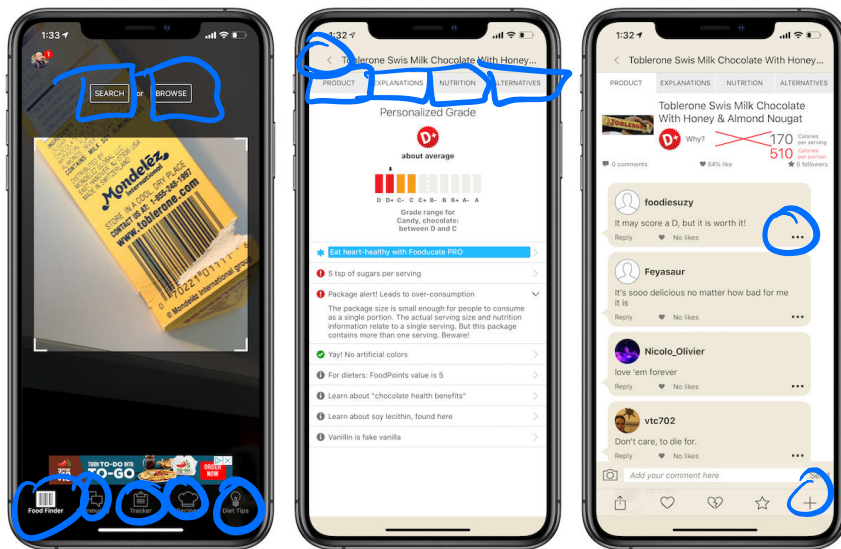
# Action-level patterns for a food tracking app:<sup>13</sup>

Pretty much anything that can be interacted with can be an action-level pattern



<sup>13</sup> Image source: [My Fitness Pal](#)

# Action-level patterns for a food education app:<sup>14</sup>

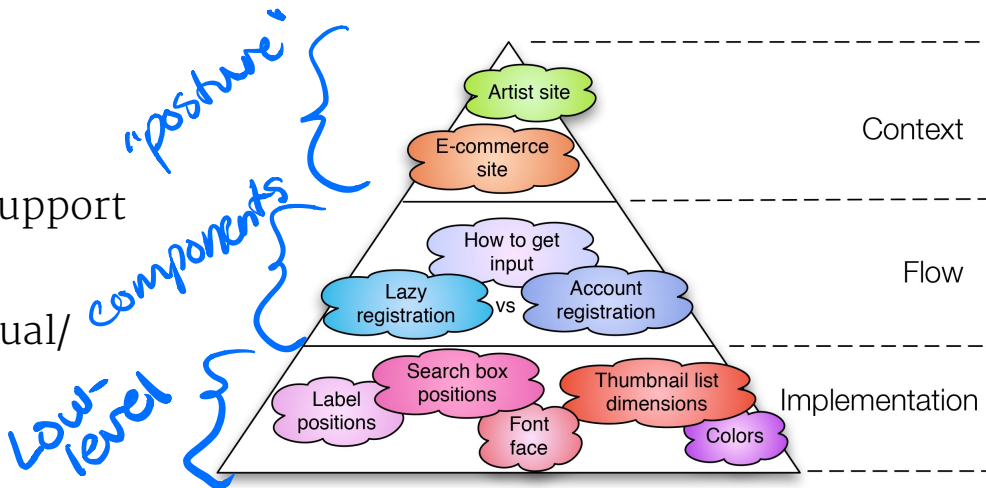


<sup>14</sup> Image source: [Fooducate](#)

# A Simplified Model<sup>15 16</sup>

Three-levels of patterns:

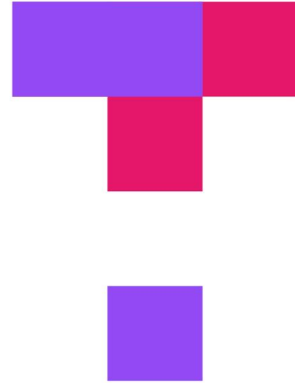
1. **Context:** Type of app
2. **Flow:** Components that support specific functions
3. **Implementation:** The visual/behavioral elements that implement the functions



<sup>15</sup> [Anders Toxboe](#)

<sup>16</sup> [More on the three-levels of patterns by Jerry Cao](#)

# TopHat Quiz



**TOP HAT**



# How do we use pattern languages?

**Common practice:** Patterns in the higher levels are defined informally, and the task- and action-level patterns are adopted through experimentation and trial and error.

**The problem:** Ineffective (e.g., lack of coherence across different levels) and inefficient (wasted effort in experimentation).

**The solution:** Defining patterns top to bottom will "generate" the design when patterns are available across all levels.<sup>5</sup>

---

<sup>5</sup>van Welie & van der Veer, 2003

# Where do we find patterns?<sup>17</sup>

Task- and action-level patterns are organized into catalogues/collections based on functional similarity.

## User Interface Design Patterns

<b>Getting input</b> <ul style="list-style-type: none"><li><b>Forms</b><ul style="list-style-type: none"><li>WYSIWYG</li><li>Password Strength Meter</li><li>Input Feedback</li><li>Captcha</li><li>Calendar Picker</li><li>Structured Format</li><li>Fill in the Blanks</li><li>Expandable Input</li><li>Keyboard Shortcuts</li><li>Input Prompt</li><li>Drag and drop</li><li>Autosave</li><li>Forgiving Format</li><li>Morphing Controls</li><li>Inplace Editor</li><li>Good Defaults</li><li>Preview</li><li>Undo</li><li>Settings</li></ul></li><li><b>Explaining the process</b><ul style="list-style-type: none"><li>Wizard</li><li>Steps Left</li><li>Completeness meter</li><li>Inline Help Box</li></ul></li><li><b>Community driven</b><ul style="list-style-type: none"><li>Vote To Promote</li><li>Pay To Promote</li><li>Wiki</li><li>Rate Content</li><li>Flagging &amp; Reporting</li></ul></li></ul>	<b>Navigation</b> <ul style="list-style-type: none"><li><b>Tabs</b><ul style="list-style-type: none"><li>Navigation Tabs</li><li>Module Tabs</li></ul></li><li><b>Jumping in hierarchy</b><ul style="list-style-type: none"><li>Notifications</li><li>Breadcrumbs</li><li>Modal</li><li>Fat Footer</li><li>Home Link</li><li>Shortcut Dropdown</li></ul></li><li><b>Menus</b><ul style="list-style-type: none"><li>Vertical Dropdown Menu</li><li>Horizontal Dropdown Menu</li><li>Accordion Menu</li></ul></li><li><b>Content</b><ul style="list-style-type: none"><li>Carousel</li><li>Tag Cloud</li><li>Progressive Disclosure</li><li>Cards</li><li>Event Calendar</li><li>Adaptable View</li><li>Article List</li><li>Continuous Scrolling</li><li>Archive</li><li>Categorization</li><li>Tagging</li><li>Thumbnail</li><li>Favorites</li><li>Pagination</li></ul></li><li><b>Gestures</b><ul style="list-style-type: none"><li>Pull to refresh</li></ul></li></ul>	<b>Dealing with data</b> <ul style="list-style-type: none"><li><b>Tables</b><ul style="list-style-type: none"><li>Table Filter</li><li>Alternating Row Colors</li><li>Sort By Column</li></ul></li><li><b>Formatting data</b><ul style="list-style-type: none"><li>Dashboard</li><li>Copy Box</li><li>Frequently Asked Questions (FAQ)</li></ul></li><li><b>Images</b><ul style="list-style-type: none"><li>Slideshow</li><li>Gallery</li><li>Image Zoom</li></ul></li><li><b>Search</b><ul style="list-style-type: none"><li>Autocomplete</li><li>Search Filters</li></ul></li></ul>	<b>Social</b> <ul style="list-style-type: none"><li><b>Reputation</b><ul style="list-style-type: none"><li>Collectible Achievements</li><li>Leaderboard</li><li>Testimonials</li></ul></li><li><b>Social Interactions</b><ul style="list-style-type: none"><li>Friend list <small>View</small></li><li>Activity Stream</li><li>Follow</li><li>Auto-sharing <small>View</small></li><li>Chat</li><li>Friend</li><li>Reaction</li><li>Invite friends</li></ul></li></ul>
			<b>Miscellaneous</b> <ul style="list-style-type: none"><li><b>Shopping</b><ul style="list-style-type: none"><li>Product page</li><li>Pricing table</li><li>Coupon</li><li>Shopping Cart</li></ul></li><li><b>Increasing frequency</b><ul style="list-style-type: none"><li>Tip A Friend</li></ul></li></ul>
		<b>Onboarding</b> <ul style="list-style-type: none"><li><b>Guidance</b><ul style="list-style-type: none"><li>Walkthrough</li><li>Blank Slate</li><li>Playthrough</li><li>Coachmarks</li><li>Guided Tour</li><li>Inline Hints</li></ul></li><li><b>Registration</b><ul style="list-style-type: none"><li>Lazy Registration</li><li>Account Registration</li><li>Paywall</li></ul></li></ul>	

<sup>17</sup> [Image source](#)

# Online Pattern Libraries

- >> UIPatterns.io
- >> UI-Patterns
- >> Mobbin
- >> UI Garage
- >> Welie

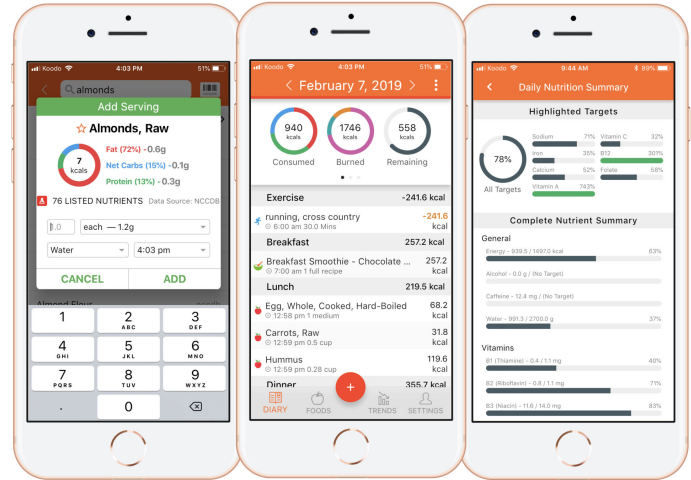


feel free to  
browse

we will not be going over  
creating patterns  
in this class

# In-Class Activity

# Source<sup>18</sup>



<sup>18</sup> Image sources: [left](#), [right](#)

## Business Goals

Mission of the application

## Posture Level

"Type" of application

## Experience Level

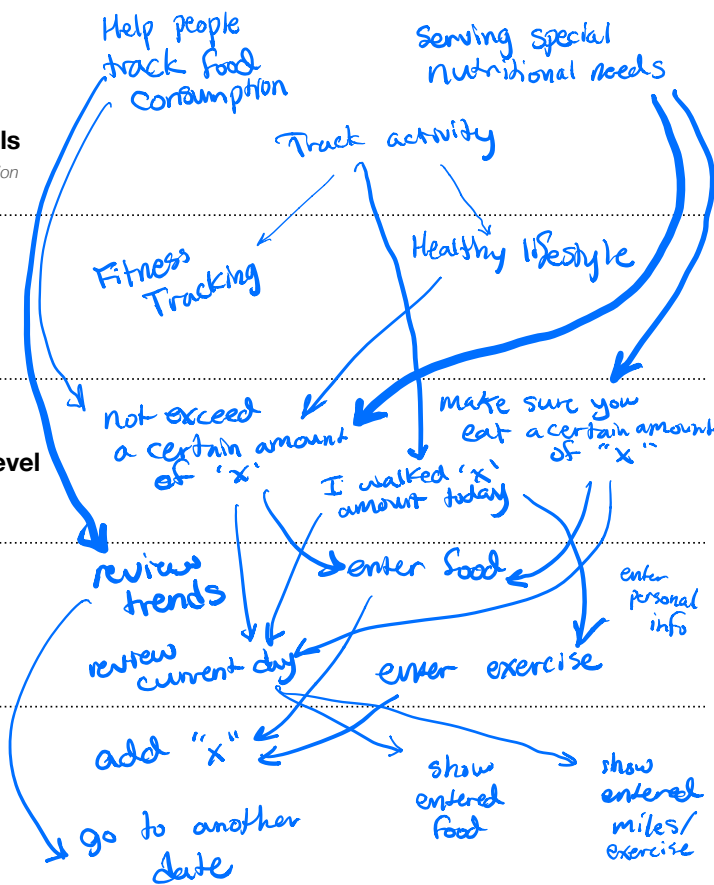
User goals

## Task Level

Task sequences

## Action Level

User actions



cronometer

HOME BLOG FORUMS EMAIL

LOGIN

## Track Your Nutrition, Fitness, & Health Data

Log your Diet, Exercise, Biometrics and Notes

SIGN UP FOR FREE



# Design Languages

**The problem:** Pattern languages help you create a design that is consistent vertically. How do we create a system that is consistent *horizontally*? I.e., how do we achieve visual and behavioral consistency in designs?

**The solution:** Design languages!



# Recap: Design Languages

**Definition:** A vocabulary of design elements that are repeatedly applied to interaction design problems.

*Non-digital example:* NASA Graphics Standard Manual.<sup>19</sup>

*Nasa build a book to articulate vocabulary* →



---

<sup>19</sup>NASA

# Source<sup>19</sup>

## NASA Uniform Patches

Personnel identification is an important facet of the NASA identification program. An embroidered patch incorporating the logotype is available for application on a wide variety of uniforms and clothing. Two patch designs, shown to the right, are available.

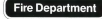
For general personnel, a white patch with a NASA flag logotype is available. This achieves the simplest and most effective identification on various types and colors of clothing that may include other badges or name tags. The patch is applied on the right front side of the garment approximately 1 1/2" (3.8 cm) directly above the breast pocket or in a comparable position on garments without pockets. On a blouse (fig. 6), the top edge of the patch aligns with the left breast pocket.

A few specific color recommendations are made for NASA uniforms: royal blue for flight suits, white for lab coats, headsets, and helmets. A 7" wide (17.8 cm) logotype may be embroidered on the back of a white lab coat (fig. 4). On a white headset or helmet, a 3" wide (7.6 cm) NASA flag detail of the logotype may be centered on the front (fig. 5).

To distinguish emergency/security personnel (security guards, firemen, etc.) is distinctive NASA flag patch with a white border, white logotype and the installation identification in black is available. The name of the emergency/security service (i.e. Fire Department) appears in white centered within a smaller black patch that is positioned 3/4" (1.9 cm) under the red patch. This configuration is worn on both shoulders of the uniform, on both shirts (fig. 8) and outer-jackets. A light blue shirt and hat with dark blue trousers or skirt is recommended.



General personnel patch



Emergency/security patches



<sup>19</sup> NASA

# UX Design Languages

**Definition:** Task- and action-level interface components that follow a consistent look and feel in appearance and behavior.

# Commonly Used Design Languages<sup>20</sup>

- >> Material Design ← commonly used
- >> Fluent Design System ← Microsoft
- >> Materialize ← beefier material
- >> Ant Design ← bit more subdued
- >> Grommet } simpler options
- >> Flat Remix }



<sup>20</sup> [Image source](#)

Fluent



# Case Studies of Design Language Use

>> Material studies examples

>> Fluent design case studies

# Assignment Preview

# Design Assignment 07: Design Patterns

The assignment will help us generate design ideas for the React Native module deliverable: a *mobile fitness/calorie-tracking* application. In a two-part assignment, you will:

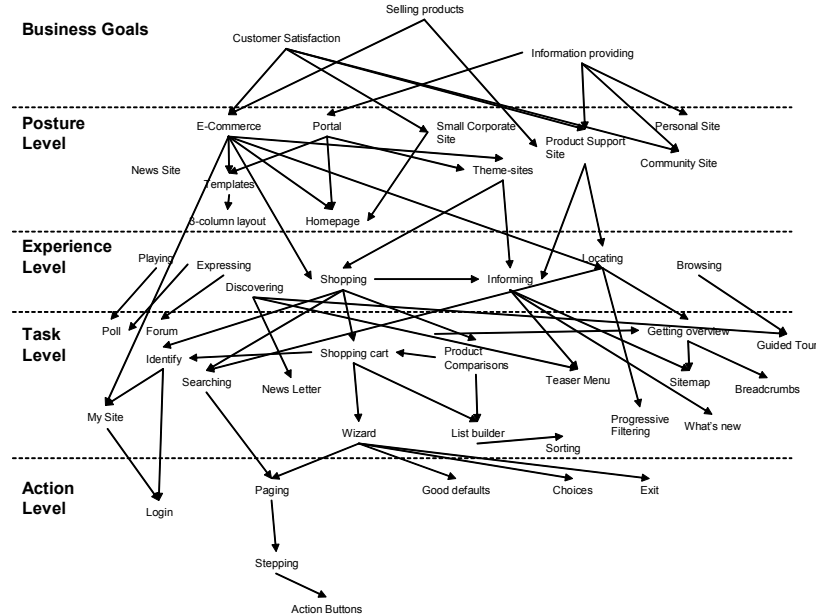
**Step 1.** Analyze the design patterns in an existing application

**Step 2.** Generate a design for a new application



# Step 1. Analyze an existing design

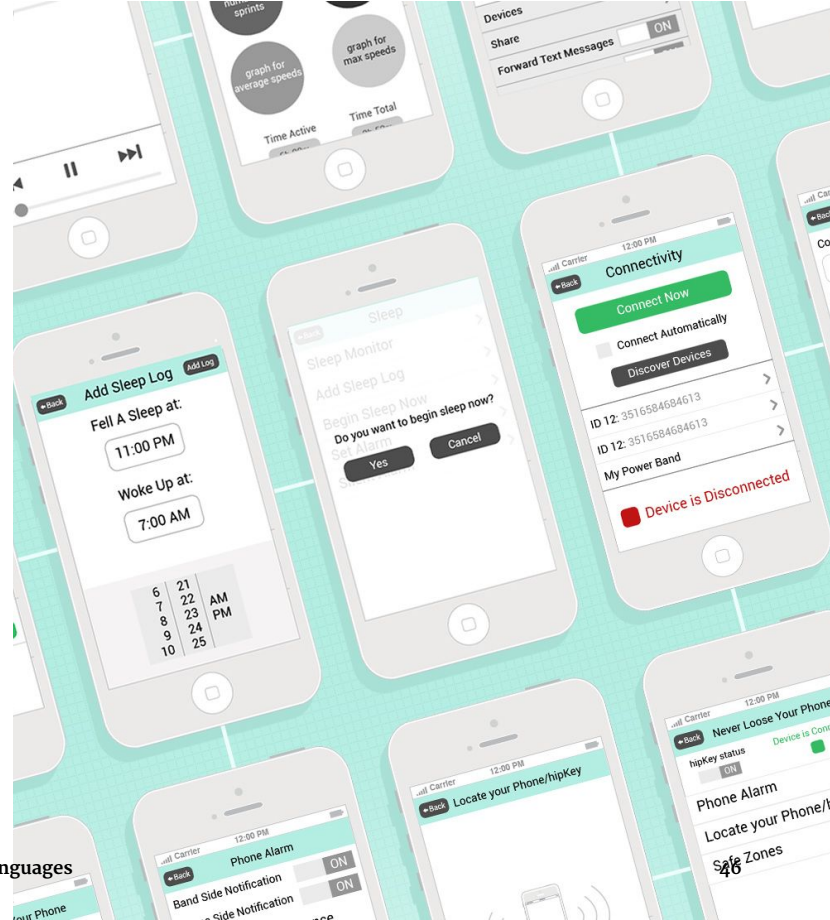
- >> Analyze the application for all levels of design patterns, from top to bottom.<sup>5</sup>
- >> Analyze screen designs for task- and action-level patterns.



<sup>5</sup>van Welie & van der Veer, 2003

## Step 2. Generate a new design<sup>22</sup>

- » Using the pattern language analysis approach, make design decisions at all levels of your application, from top to bottom.
- » Visualize your decisions at the task- and action-levels by creating wireframes.



<sup>22</sup> [Image source](#)

# What did we learn today?

- >> Design Patterns
- >> Design Languages
- >> Assignment Preview