

# Mobile Interface Design Patterns

J.Serrat

102759 Software Design

June 18, 2014

## Index

1 UI patterns

2 Navigation

3 Invitations

4 Sorting

5 Antipatterns

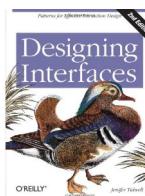
# Slides sources



*Mobile Design Pattern Gallery: UI Patterns for Mobile Applications.* T. Neil. O'Reilly, 2012.



*Android Design Patterns: Interaction Design Solutions for Developers,* G. Nudelman. Wiley, 2013.



*Designing interfaces: patterns for effective interaction design.* J. Tidwell, 2nd ed. O'Reilly, 2011.



[www.androidpatterns.com](http://www.androidpatterns.com)

UI patterns	Navigation	Invitations	Sorting	Antipatterns
<b>UI patterns</b>				

Mobile UI design has different challenges than for large screen + mouse + keyboard:

Tiny screen No sidebars, long header menus, tree controls . . .

Variable screen *width* Scrolling is easy ⇒ width matters the most.  
Controls automatically resized when turning the mobile 90°

Touch screen Main interaction mode. Targets must be large enough ( $\geq 0.7$  or 1 cm) and/or well separated

# UI patterns

Some challenges of mobile design:

**It's hard to type text** Make typing unnecessary or very limited.

Autocomplete fields and pre-fill with good default values

**Challenging environments** People use their phones in all kinds of places ⇒ large ambient lighting and noise differences.  
Tiny text difficult to read and tap in motion.

**Limited attention** Users look at the interface while doing other things : walking, eating, talking to other people, *driving*. Design for distracted users: easy, quick interaction, self-explanatory UI.

5 / 37

# UI patterns

“A user interface is well-designed when the program behaves exactly how the user **thinks** it will.”

“The applications that are easy to use are designed to be **familiar**”

Familiar UI ≠ equal but that **parts are recognizable enough** so that people can apply their previous knowledge to a novel interface.

6 / 37

# UI patterns

Patterns are

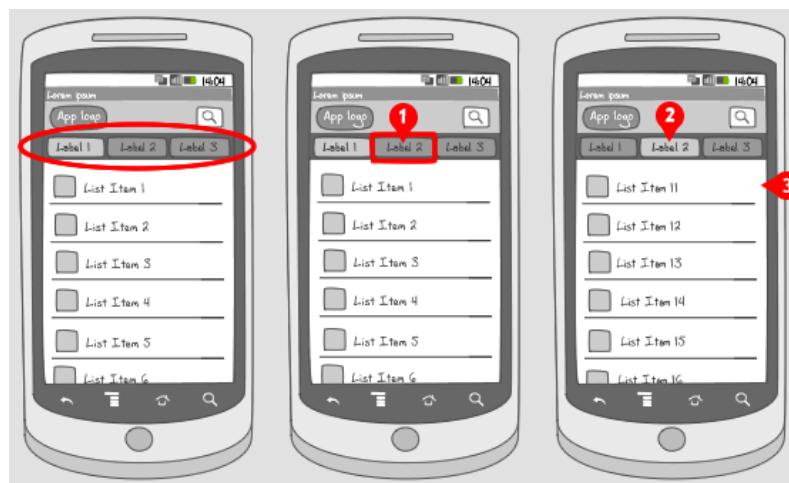
- these familiar parts in aspect or interaction, in ways you can reuse in different contexts
- solutions to common UI design problems but not off-the-shelf components: implementations differ
- not concrete controls but relationships among elements (e.g. place help next to a text field)

7 / 37

# UI patterns

Q: How to show many elements at the same level of importance and search them quickly ?

Q: **Segmented control** : place 2-5 buttons or tabs horizontally aligned, that act as filters.



# UI patterns

Q: How to show a vast amount of hierarchical data ?

A: **Expandable list** : Items are organized in a two-level list. A first level item can be expanded to show its children. An indicator shows the state, collapsed or expanded.



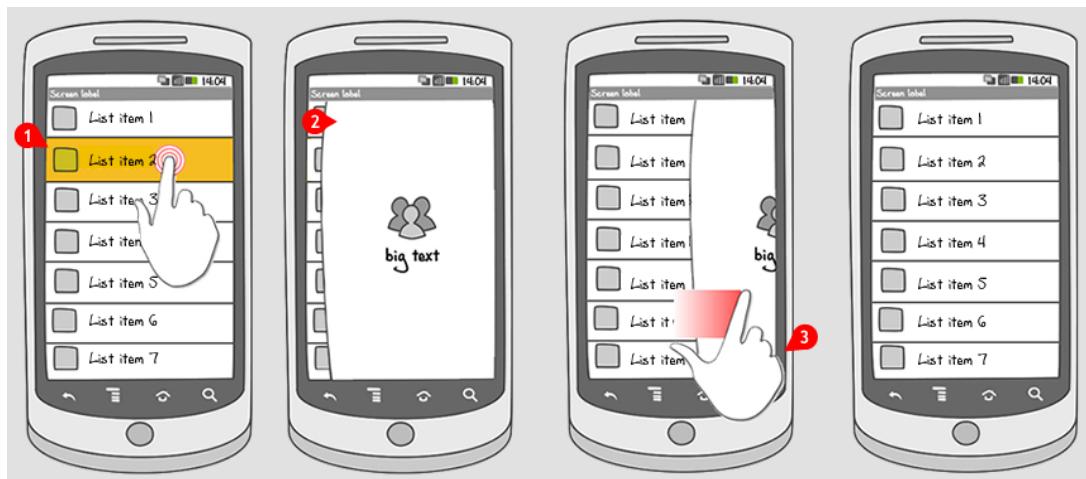
[www.androidpatterns.com](http://www.androidpatterns.com)

9 / 37

# UI patterns

Q: How to show a vast amount of hierarchical data ?

A: **Sliding layer** : after a certain trigger (button click, item selection, etc), a sticky container will slide from any side of the screen. Shall be dismissed or closed by swiping it away or tapping.



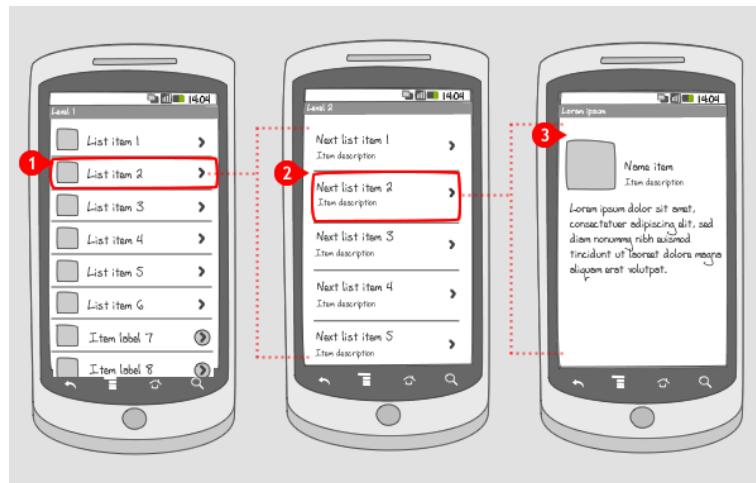
[www.androidpatterns.com](http://www.androidpatterns.com)

10 / 37

# UI patterns

Q: How to show a vast amount of hierarchical data ?

A: Drill down : tapping on an item in the list opens its children in next level



[www.androidpatterns.com](http://www.androidpatterns.com)

11 / 37

# Why UI patterns

UI toolkits + style guide + imitation of existing applications = passable interface.

General UI design principles are necessary but not sufficient: tell what to do but not how. How do you “provide feedback”, “prevent errors” ... ?

Patterns fill the gap between principles and actual UI toolkit elements.

12 / 37

# Navigation patterns

Navigation : how to move across the application screens, functionality, content ...

Poor navigation means “can’t go back”, “can’t find things”, “don’t know how to do X”, “don’t know where I am”

Good navigation, like good design, is invisible. Applications with good navigation just feel intuitive and make it easy to accomplish any task.

## Navigation

**Primary** : presentation of the main application options.

**Secondary** : presentation of additional choices derived from some/each primary navigation choice.

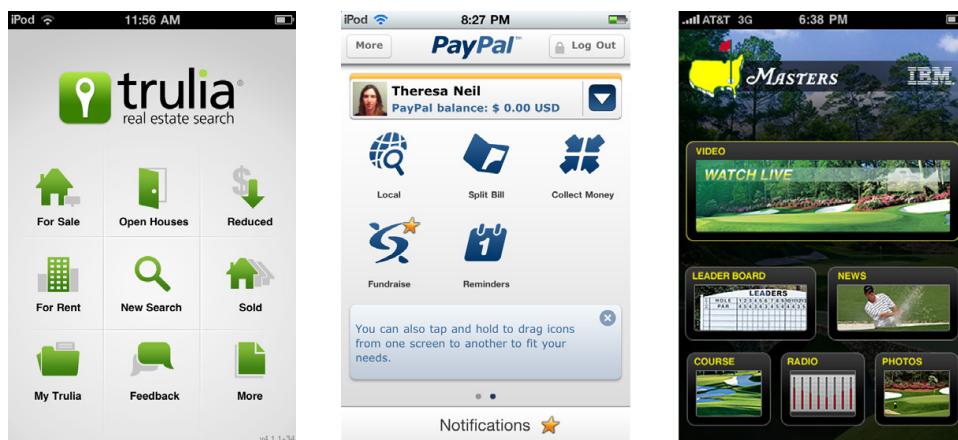
How ?

13 / 37

# Primary Navigation

## Springboard

Landing page of menu options that act as a jumping off point into the application. Use a **grid** layout for items of equal importance, or an **irregular** layout to emphasize some items more than others.



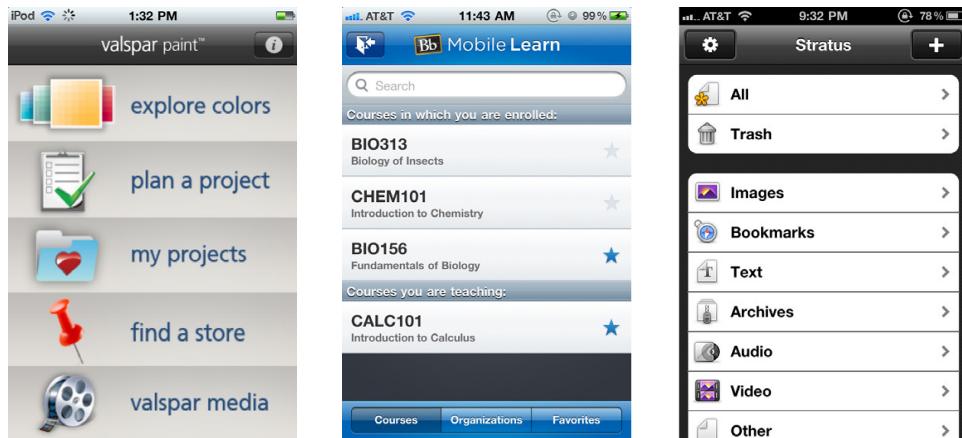
14 / 37

# Primary Navigation

## List menu

**List** of items representing the main choices. **Enhanced lists** are simple list menus with additional features for searching, browsing or filtering.

Work well for long titles or those that require sub text.

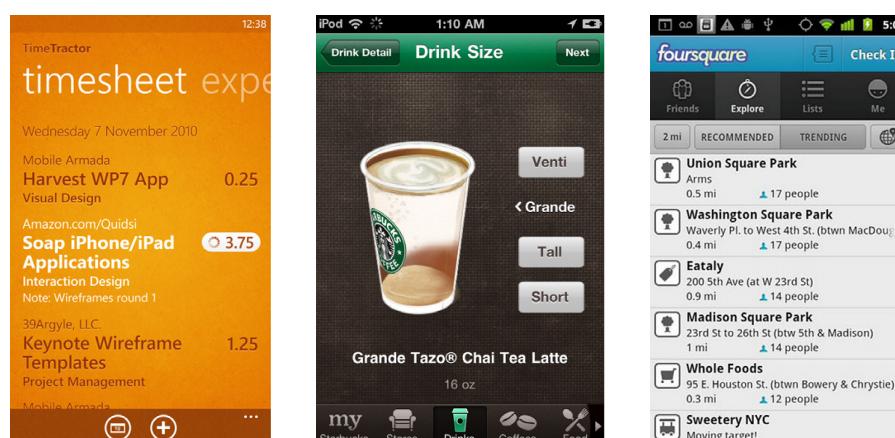


15 / 37

# Primary Navigation

## Tabs

Partition application functionality or views in different screens reachable with a single click. Horizontally scrolling tabs avoid a “More...” tab. Bottom tabs are more thumb friendly.



16 / 37

# Primary Navigation

## Metaphor

Landing page modeled to reflect the applications metaphor. Used primarily in games, but also in applications that help people **catalog and categorize** items, like notes, books...



17 / 37

# Primary Navigation

Beware of using a wrong metaphor:

- left:** educational application for exploring facts from around the globe
- right:** a globe for navigating news content? Stories are **not** surfaced from specific geographic locations. The globe is just a spinning sphere that is hard to read and harder to browse.



18 / 37

# Secondary Navigation

## Secondary Navigation

Navigation within a page or module. Any of the primary navigation patterns can be reused as secondary navigation patterns.

1 tabs, 2 springboard



1 tab, 2 list

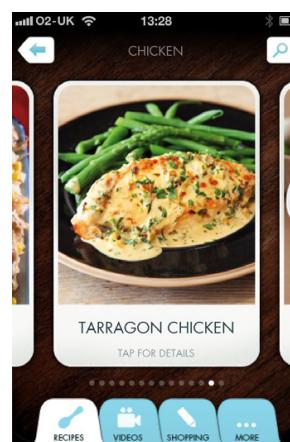


19 / 37

# Secondary Navigation

## Page carousel

Used to quickly navigate an *small* set of pages using the **flick** gesture. Add always a **page indicator** to displays how many pages are in the carousel. The two examples use the page carousel within a selected tab.

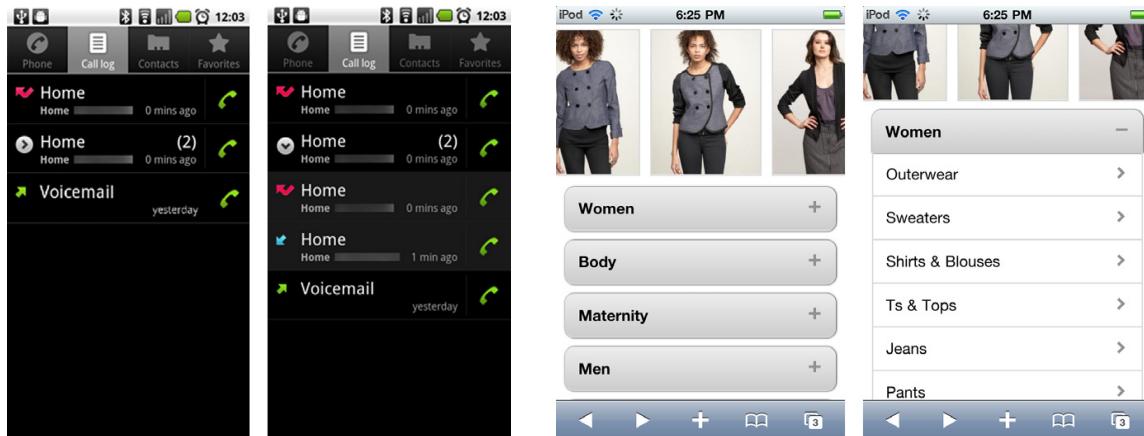


20 / 37

## Secondary Navigation

### Expanding list

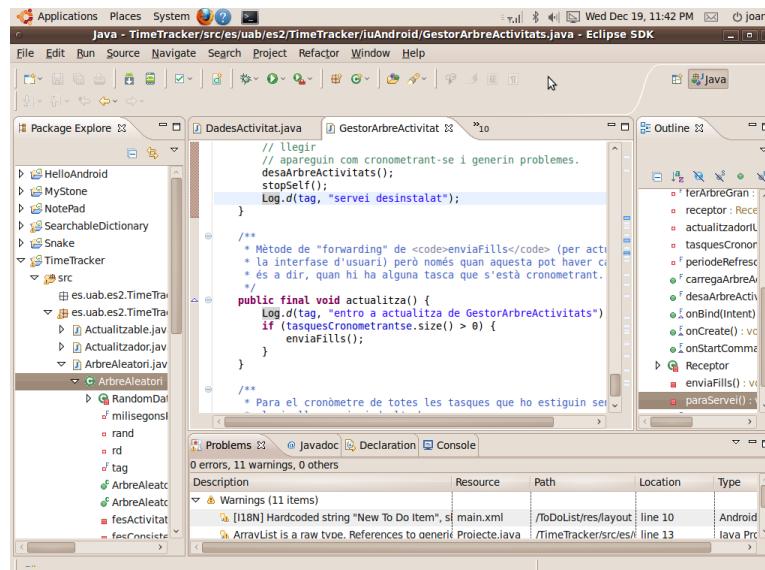
Allows a single screen drill down to reveal more information.  
Tapping the > icon expands/collapses the list to show the individual instances. No further expandable.



21 / 37

## Invitations

Remember the first time you run Eclipse ? Tens of buttons, several panels, multiple menu and submenu options, five tabs . . . “I’ll never manage to use it!” But there was **no choice**.



22 / 37

# Invitations

Mobile is different. There may be tens of competitor apps to yours and it takes less to install any of them than struggling with an intuitive interface. You must make it right from the very beginning.

## Invitations

Helpful [tips](#) that are displayed the [first time](#) a user opens an application or arrives at a new place. They suggest actions and guide the user to the intended functionality. How ?

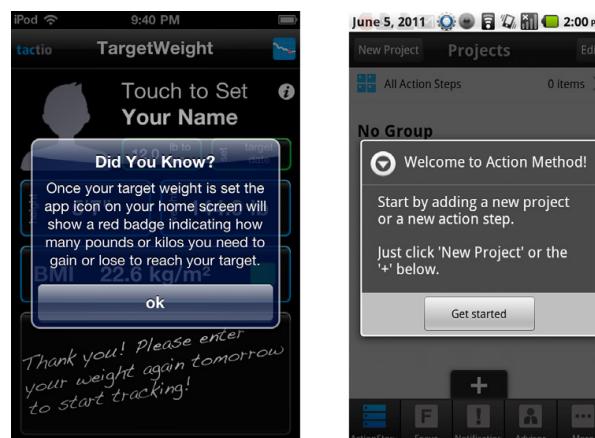
- dialog
- tip
- tour
- transparency
- first time through

23 / 37

# Invitations

## Dialog

A plain dialog with text instructions. [Most common](#) type of invitation probably because it is the easiest to program. It is also most likely to be [dismissed and ignored](#). Keep instructions [short](#).

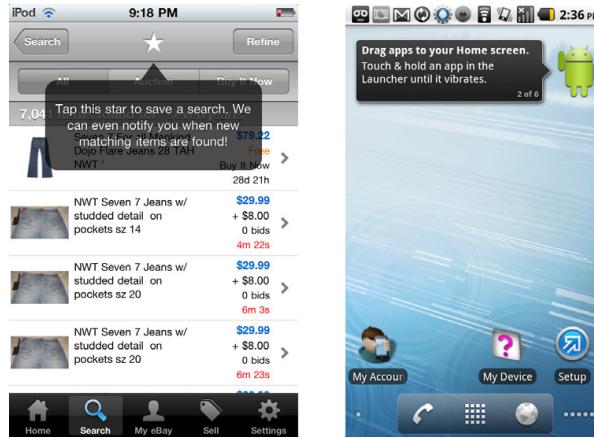


24 / 37

# Invitations

## Tip

Small globe with text. Can be implemented anywhere in the screen, making it more contextually relevant than a dialog. Keep the content short, and remove the tip once screen is touched.



25 / 37

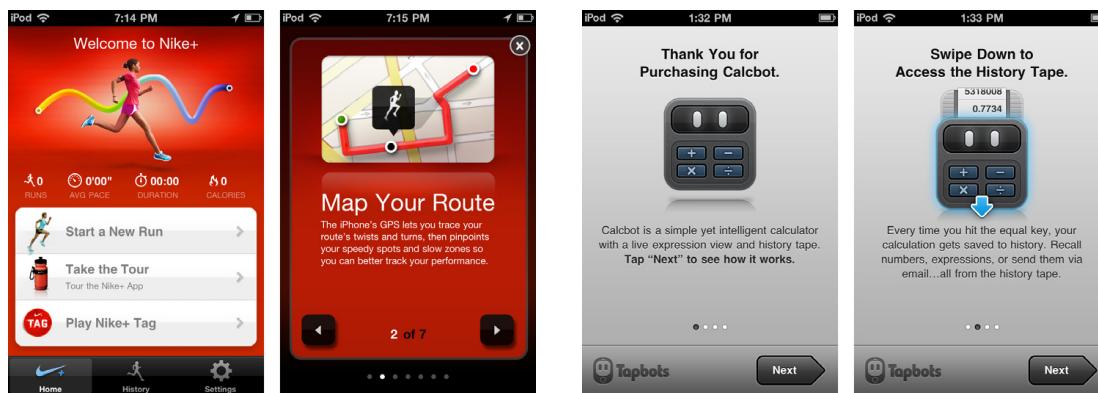
# Invitations

## Tour

Provides the ultimate invitation by offering a **screen-by-screen, feature-by-feature** exploration. Offered on the **home screen**.

Highlights key features of the application. Keep it **short** and visually engaging.

Problem with short term memory.



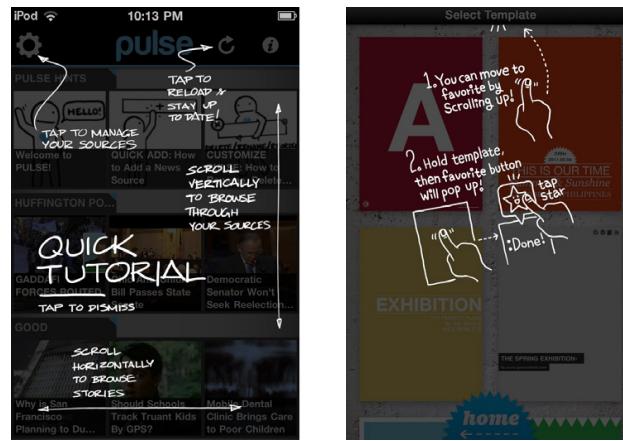
26 / 37

# Invitations

## Transparency

See through layer with an **usage diagram** over the actual screen.

Not meant to compensate for poor screen designs. Remove the transparency once the screen is touched.

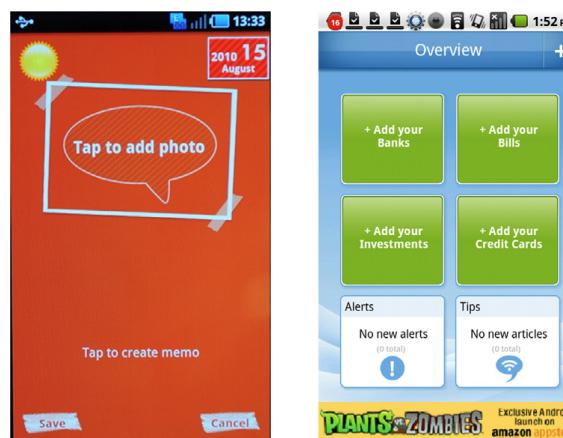


27 / 37

# Invitations

## First time through

Unlike the other invitations, **don't precede** the screen they refer to but are **built into** the screen design. They **remain** in the interface until they are overwritten with content or the action is performed. Clearly **differentiate** the invitation from other content : with icon, color, text size different from regular content.



28 / 37

# Sorting

## Sorting content

Content like lists, search results etc. need to be sorted to be **visually searched** in a fast way. Choose a reasonable **default** sort and offer sorting according to **other criteria**.

How ?

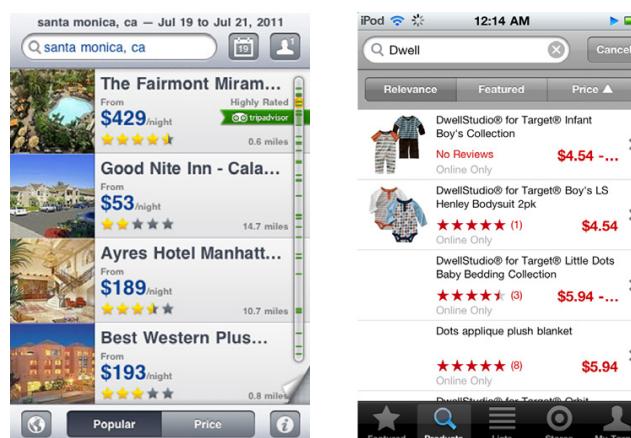
- onscreen sort
- sort order selector
- the sort form

29 / 37

## Sorting content

### Onscreen sort

A row with a few **toggle buttons** placed horizontally at the top or bottom, each corresponding to a sorting criterion. Clearly show which option is selected or “on”. Also, distinguish **ascending from descending**.

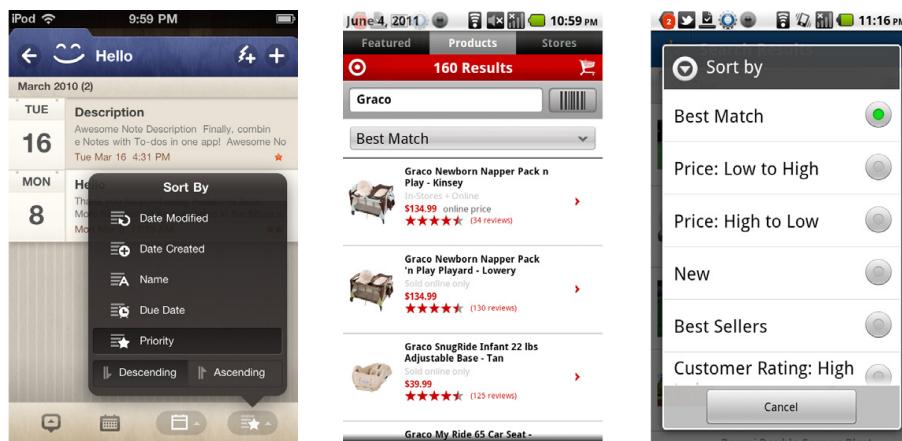


30 / 37

# Sorting content

## Sort order selector

When the number of sorting options is 5 or more, use some OS selection control like the spinner or the contextual menu in Android. The option titles can be longer, **more explicit**, and **more options** can be displayed.



31 / 37

# Sorting content

## Sort form

Some applications have **fused the sort and filter functionality** into one screen **form**. This is the most effort intensive sort pattern, requiring the user to 1) open the form, 2) select two options, 3) tap a “done” button.



32 / 37

# Antipatterns

## Antipatterns

Classes of **commonly-reinvented bad solutions** to design problems.  
They are studied as a category so they can be avoided in the future.

Which ?

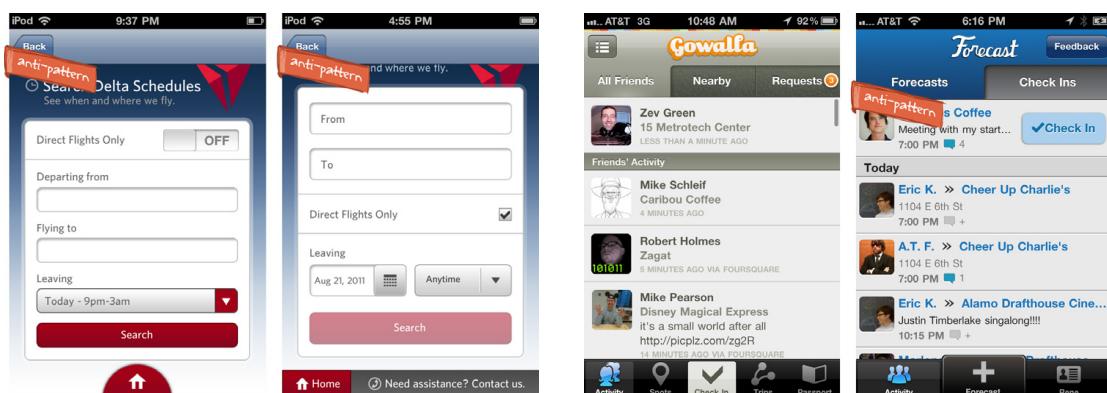
- Novel notion
- Metaphor mismatch
- Idiot box
- Oceans of buttons

33 / 37

# Antipatterns

## Novel notion

Novel designs intended to be creative and innovative. But they're just hard to understand and use. Can be found anywhere in an application, from primary navigation down to an individual control, or gesture.

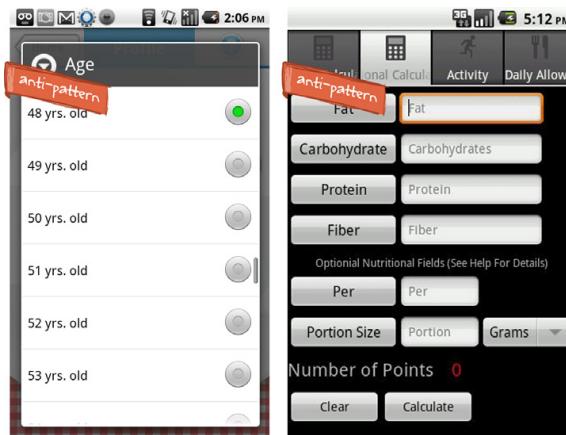


34 / 37

# Antipatterns

## Metaphor mismatch

Picking the wrong metaphor for the interface. Can occur at a low level, when a control or icon is used inappropriately, or at a high level, where the conceptual model for the application doesn't match the user's mental model.

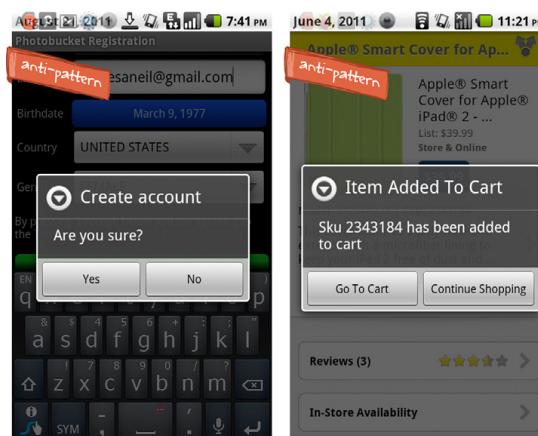


35 / 37

# Antipatterns

## Idiot box

Disrupting the user interaction when not really necessary. Avoid disrupting the user's workflow, only show a confirmation dialog when an irreparable action is being taken (like a permanent delete).



36 / 37

# Antipatterns

## Oceans of buttons

Very long button bars or grid. All the buttons of the same size and color so it is difficult to determine which one to click without [reading them all](#).

Use [contextual tools](#) when you find yourself [repeating the same buttons](#).

