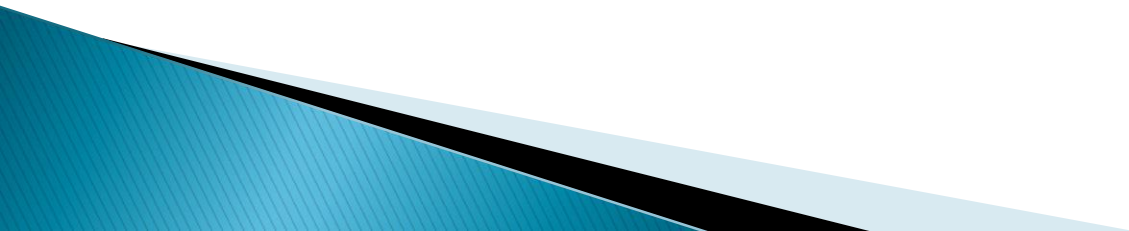


Professors d'IDI - UPC

# IDI – Design Principles



# Design Principles

Based on the **Bruce Tognazzini** document:

*<http://asktog.com/atc/principles-of-interaction-design>*

- ▶ **Effective interfaces:**

- Instilling in their users a sense of control
- Do not concern the user with the inner working of the system,

- ▶ **Effective applications:**

- Perform a maximum of work while requiring a minimum of information from users.

# 1. Aesthetics

## Fashion should never trump usability

- ▶ Aesthetical appearance is appealing
- ▶ But design based on fashion will artificially generate obsolescence
- ▶ A new fashion should not detract from user-performance
- ▶ The current trend of contrast reduction is really painful (see examples)

In any case, user test



# 1. Aesthetics

## Fashion should never trump usability

- ▶ User knows standard controls. Non-standard controls require cognitive effort and produce errors

pepper catsear brussels sprout sweet pepper daikon spring onion aubergine broccoli rabe quandong mustard celery corn groundnut peanut. Mung bean fennel eggplant water spinach bunya nuts sierra leone bologi epazote okra caulie groundnut black-eyed pea parsnip fava bean squash.

Parsnip tomatillo swiss chard garbanzo gourd potato silver beet. Celery swiss chard melon zucchini arugula pea quandong beet greens radish artichoke black-eyed pea endive winter purslane horseradish garlic amaranth collard greens chickpea. Rock melon pumpkin collard greens celery broccoli rabe endive nori brussels sprout gourd courgette sea lettuce artichoke desert raisin coriander chard.

Collard greens ricebean horseradish wattle seed chard epazote potato peanut gram earthnut pea spinach yarrow desert raisin salad mung bean summer purslane fennel. Water spinach celery cucumber grape cauliflower nori daikon sweet pepper endive lentil turnip greens catsear leek beet greens. Melon seakale parsnip soybean bamboo shoot fennel scallion. Corn groundnut squash corn aubergine bitterleaf azuki bean dandelion courgette broccoli rabe. Chickpea chickweed groundnut nori okra lentil water

# 1. Aesthetics

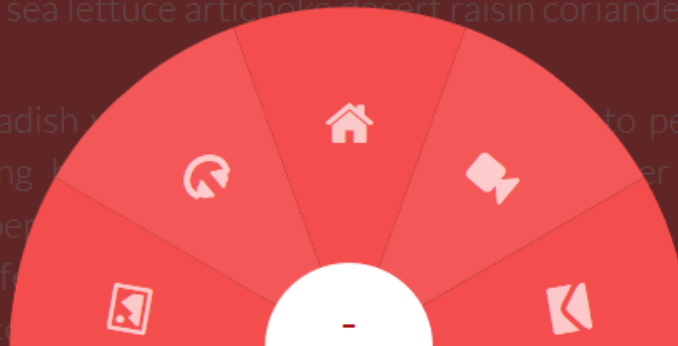
## Fashion should never trump usability

- ▶ User knows standard controls. Non-standard controls require cognitive effort and produce errors. **Who could imagine this?**

pepper catsear brussels sprout sweet pepper daikon spring onion aubergine broccoli rabe quandong mustard celery corn groundnut peanut. Mung bean fennel eggplant water spinach bunya nuts sierra leone bologi epazote okra caulie groundnut black-eyed pea parsnip fava bean squash.

Parsnip tomatillo swiss chard garbanzo gourd potato silver beet. Celery swiss chard melon zucchini arugula pea quandong beet greens radish artichoke black-eyed pea endive winter purslane horseradish garlic amaranth collard greens chickpea. Rock melon pumpkin collard greens celery broccoli rabe endive nori brussels sprout gourd courgette sea lettuce artichoke raisin coriander chard.

Collard greens ricebean horseradish to peanut gram earthnut pea spinach yarrow desert raisin salad mung cauliflower nori daikon sweet per leek beet greens. Melon seakale parsnip soybean bamboo shoot f groundnut nori okra lentil water bean dandelion courgette broc



# 1. Aesthetics

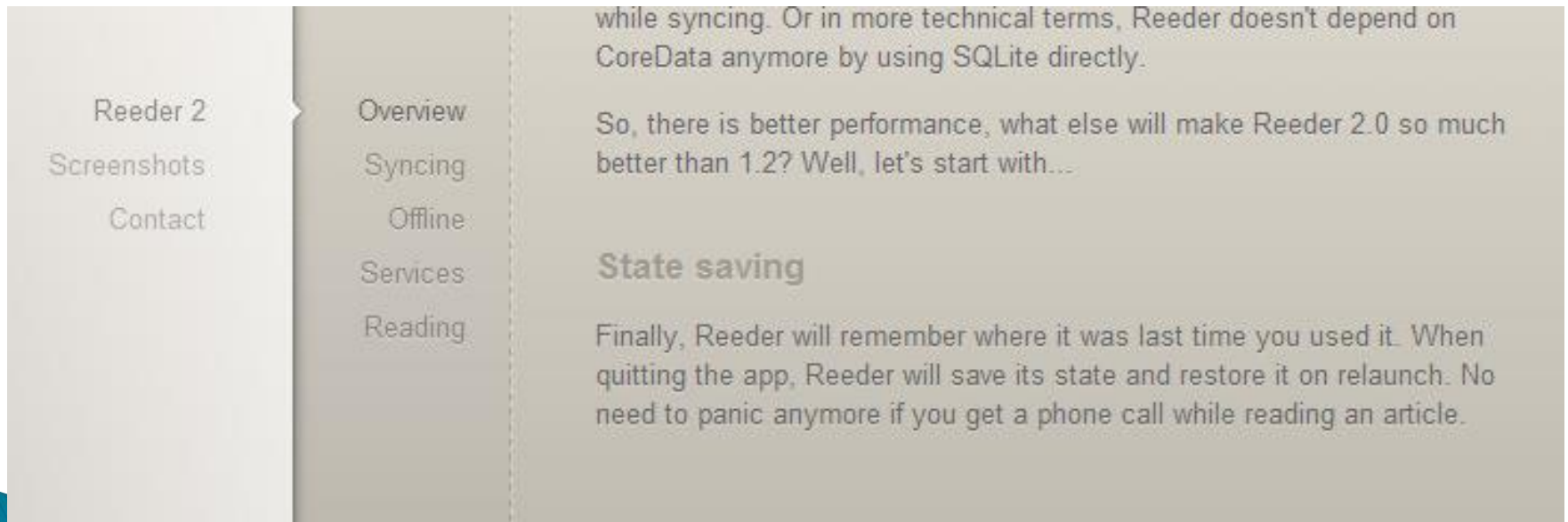
Fashion (gestures) trumping usability (lack of affordances):



# 1. Aesthetics. *Contrast*

Text that must be read should have high contrast:

- ▶ Check [contrastrebellion.com/](http://contrastrebellion.com/)
- ▶ This is unacceptable:



# 1. Aesthetics. *Contrast*

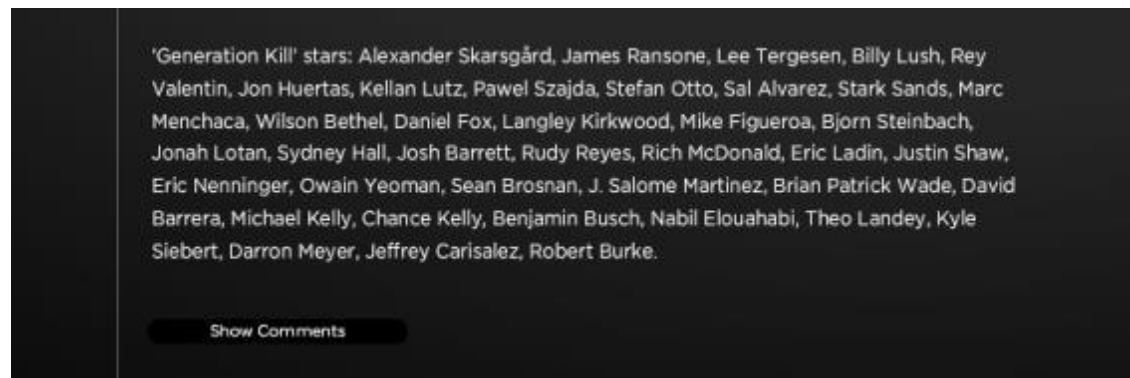
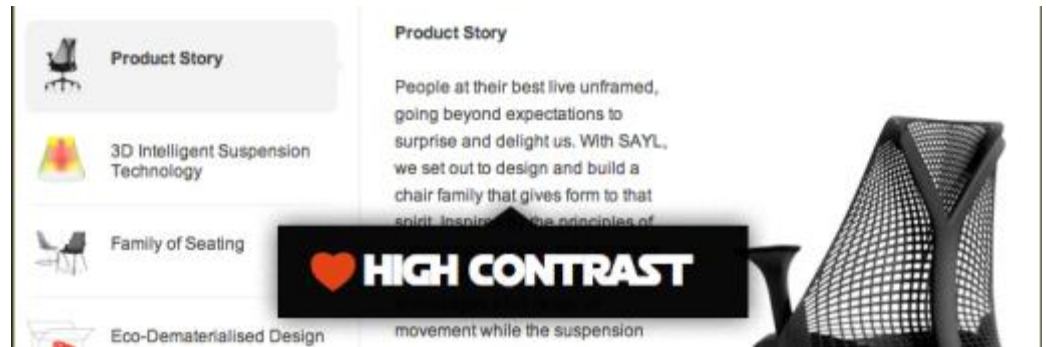
Text that must be read should have high contrast:

- ▶ Favor black text on white or pale yellow backgrounds. Avoid gray backgrounds.
- ▶ Use font sizes that are large enough to be readable on standard displays
- ▶ Favor particularly large characters for the actual data you intend to display
  - As opposed to labels and instructions.



# 1. Aesthetics. *Contrast*

High contrast can be aesthetically pleasant



## 2. Anticipation

Bring to the user all the information and tools needed for each step of the process

- ▶ Should anticipate the user's needs
- ▶ Information in place & visible  
(if the user cannot find it, it will never be used)
- ▶ Requires deep understanding of both the task domain and the users
- ▶ The penalty may be the complete loss of the user or client

In any case, user test



## 2. Anticipation

Bring to the user all the information and tools needed for each step of the process

- ▶ Anticipate user's needs
  - Put the weather first...



## 2. Anticipation

Bring to the user all the information and tools needed for each step of the process

- ▶ Information visible



## 2. Anticipation

Bring to the user all the information and tools needed for each step of the process

- ▶ Information visible: Pinterest makes Pinit, Send & Love buttons visible **only on hover**



# 3. Autonomy<sub>1</sub>

Computer interface, and task environment all “belong” to the user but user–autonomy doesn’t mean we abandon rules

- ▶ Give users some breathing room
  - e.g. provide a certain degree of customization –desktop–
- ▶ Enable the users make their own decisions
  - Otherwise they may feel constrained and frustrated

# 3. Autonomy<sub>2</sub>

## Keep the user informed

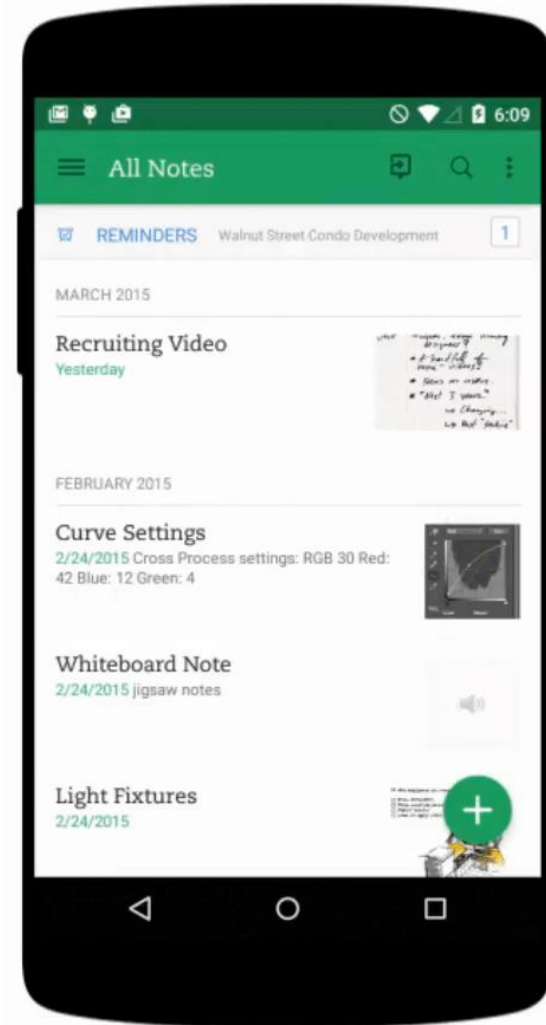
- ▶ Autonomy/Control cannot be exerted in the absence of sufficient information
  - Provide information on current state, tasks
- ▶ Keep the information timely, and accurate
  - Progress indicators that are inaccurate are annoying  
E.g. Updated indicator showing a 5' task that turns an hour!!!!
  - Lying the user is never a good practice



# 3. Autonomy<sub>2</sub>

Keep the user informed. Current state

- ▶ E.g.: button clicked
  - No feedback means the user guess
    - Maybe wrong

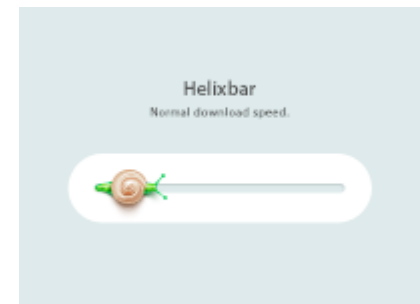
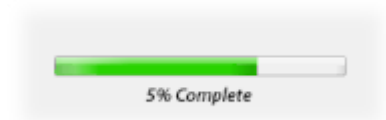




# 3. Autonomy<sub>2</sub>

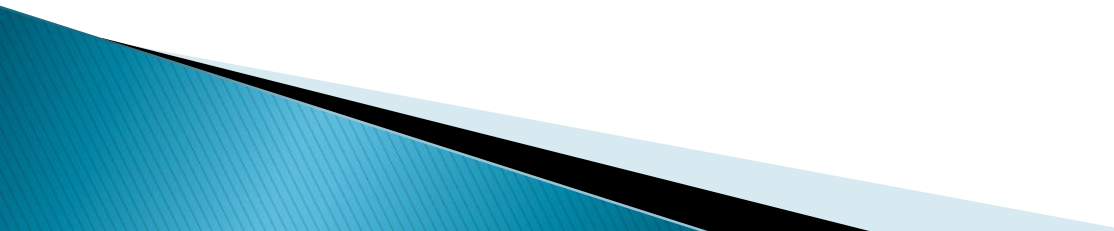
Keep the user informed. Progress indicators

- ▶ Should appear timely and be informative
  - Tasks that require  $> 1 \text{ seg}$   $\Rightarrow$  cursor busy
  - Tasks that require enough time ( $> 10 \text{ seg}$ ) should show progress indicator
    - Indicate the computer is busy
    - Should estimate time or % of task
    - Tell users what's happening
    - Let the user do another thing



# 3. Autonomy<sub>2</sub>

Keep the user informed. Error messages

- ▶ Should provide information on the reason for the error
  - ▶ Should provide a clue on how to act
  - ▶ Should not be fast enough so that the user cannot read them
- 

## 4. Color

When using color to convey information in the interface, also use clear, secondary cues

- ▶ Approximately 10% males and up to 1% females have some form of color blindness
- ▶ With age, people start having vision problems
- ▶ Use websites such as <http://enably.com/chrometric/> to test
- ▶ Use color, but use it wisely

In any case, user test after aesthetic color changes



# 4. Color

Warning: color meaning can change depending on the culture

## Yellow:

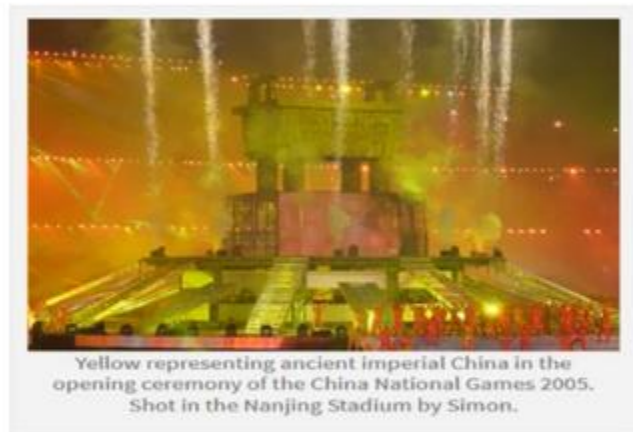
We're addressing yellow second, because like red, the color has some quite distinctive and problematic meanings.

Yellow can be dangerous to use in your marketing materials. If your copywriting refers to a product or service and that is connected to yellow, great care should be taken.

When the term for "yellow", "黄" (huáng) is used in connection to any kind of publication or media, it means the thing it describes is pornographic. For example, 黄图 (huáng tú, "yellow picture") means pornographic pictures and graphics, 黄书 (huáng shū, "yellow book") means pornographic writings, 黄片 (huáng piān, "yellow clips") means pornographic movies, and 黄网 (huáng wǎng, "yellow web") means pornographic websites.

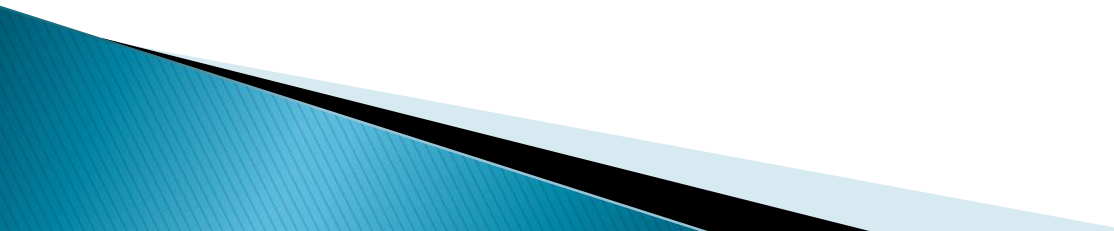
So the next time you see Chinese police officers wielding a big banner saying 扫黄打非 (sǎo huáng dǎ fēi, "sweep the yellow and beat the un-"), you will know it's time for one of their raids on pornography.

(Yes, that direct translation is a little obtuse — such is the habit of writing contemporary Mandarin in the mainland — more accurately, the police banner expresses the People's Armed Police's will to "sweep the street of yellow publications and beat unlawful activities").



# 5. Consistency

Must be analyzed in different dimensions

- ▶ Levels of consistency
  - ▶ Induced inconsistency
  - ▶ Induced continuity
  - ▶ Consistency with user expectations
- 

# 5. Consistency

Must be analyzed in different dimensions

- ▶ Levels of consistency:
  - Platform consistency,
  - Across suite of products,
  - In-app: in a single app/web,
  - Visible structures,
  - Invisible structures,
  - User behavior
- ▶ Induced inconsistency
- ▶ Induced continuity
- ▶ Consistency with user expectations

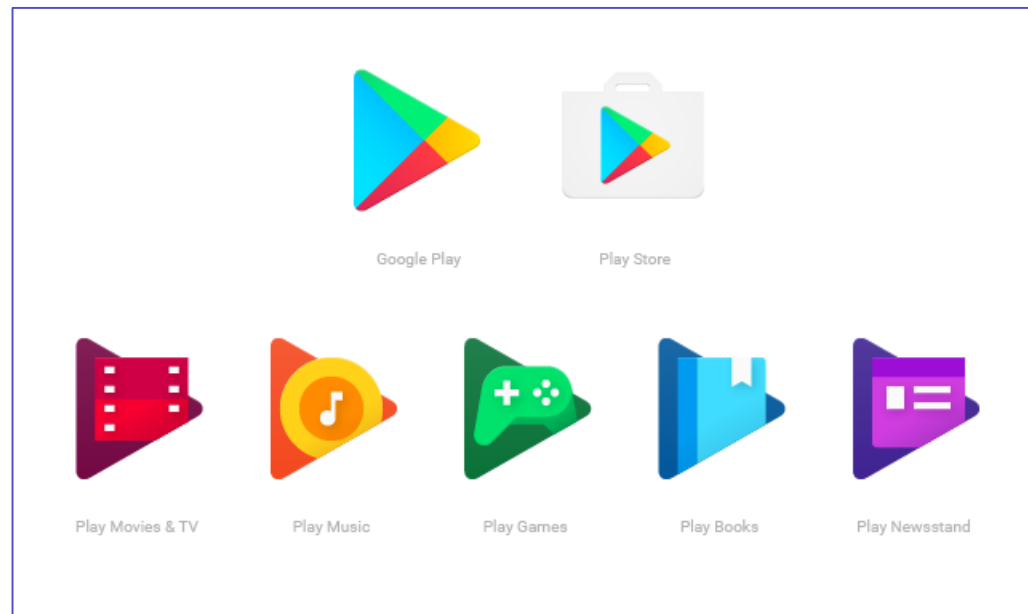
# 5<sub>1</sub>. Consistency. *Levels of Consistency*

## ► Platform consistency

- Guidelines (e.g. UX Android, iOS...) and *in-house* (same company...)
- Unwritten rules (assumed by the community)
- Keep a general *look & feel* across products/services
  - Communicates brand
  - Makes adoption easier

# 5<sub>1</sub>. Consistency. *Levels of Consistency*

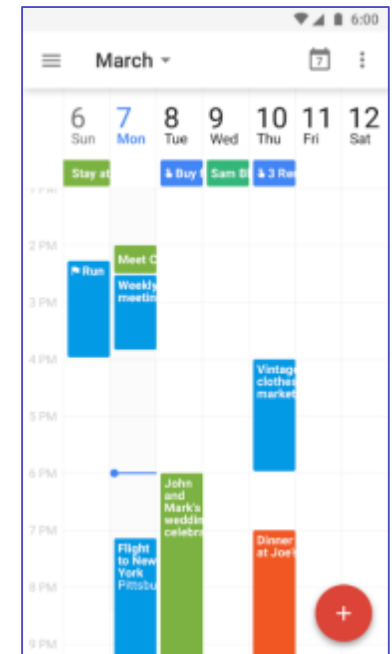
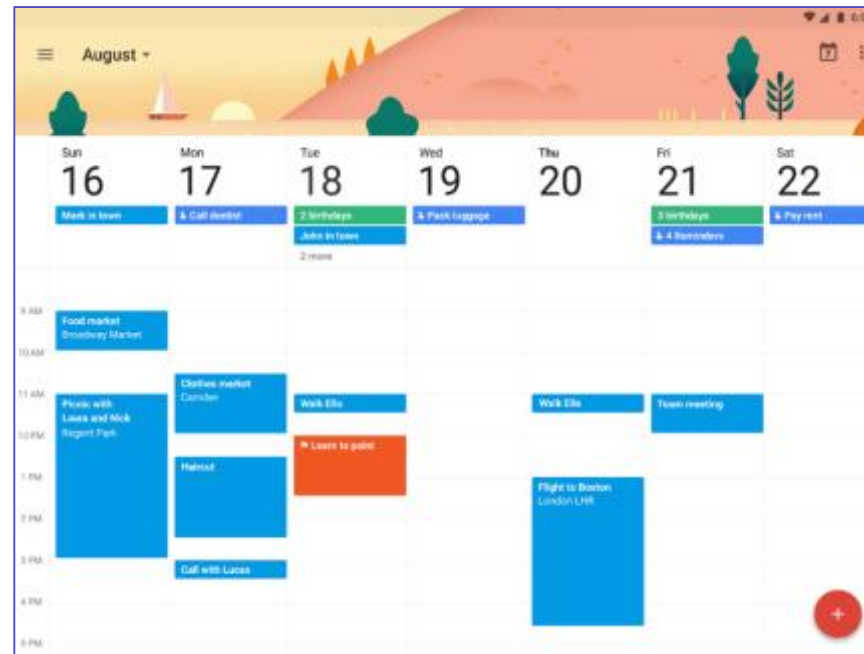
- ▶ Across suite of products:
  - Communicates family (e.g. Microsoft Office, Google apps)





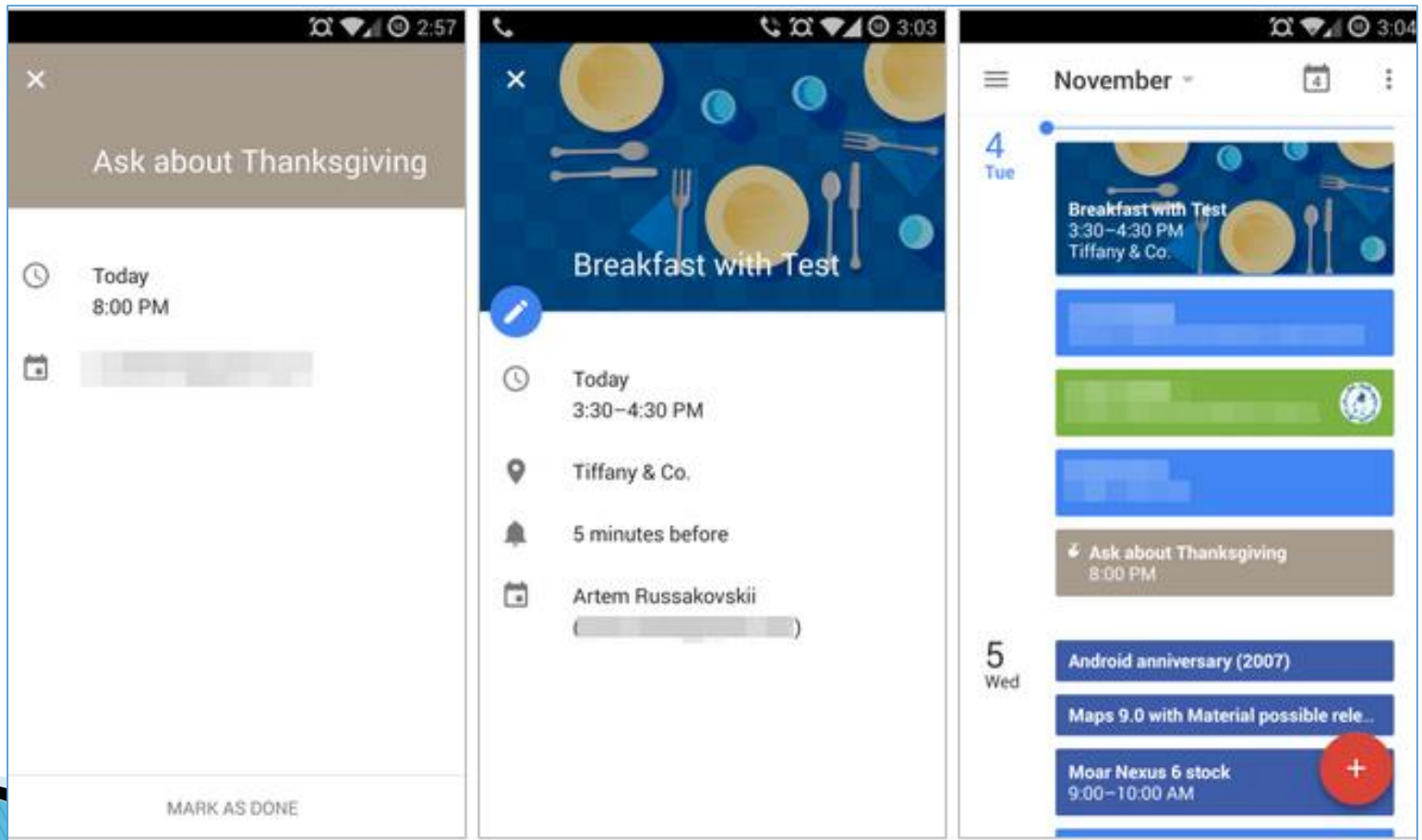
# 5<sub>1</sub>. Consistency. *Levels of Consistency*

- ▶ In-app: in a single app/web
  - specific look & feel



# 5<sub>1</sub>. Consistency. *Levels of Consistency*

- In-app: in a single app/web



# 5<sub>1</sub>. Consistency. *Levels of Consistency*

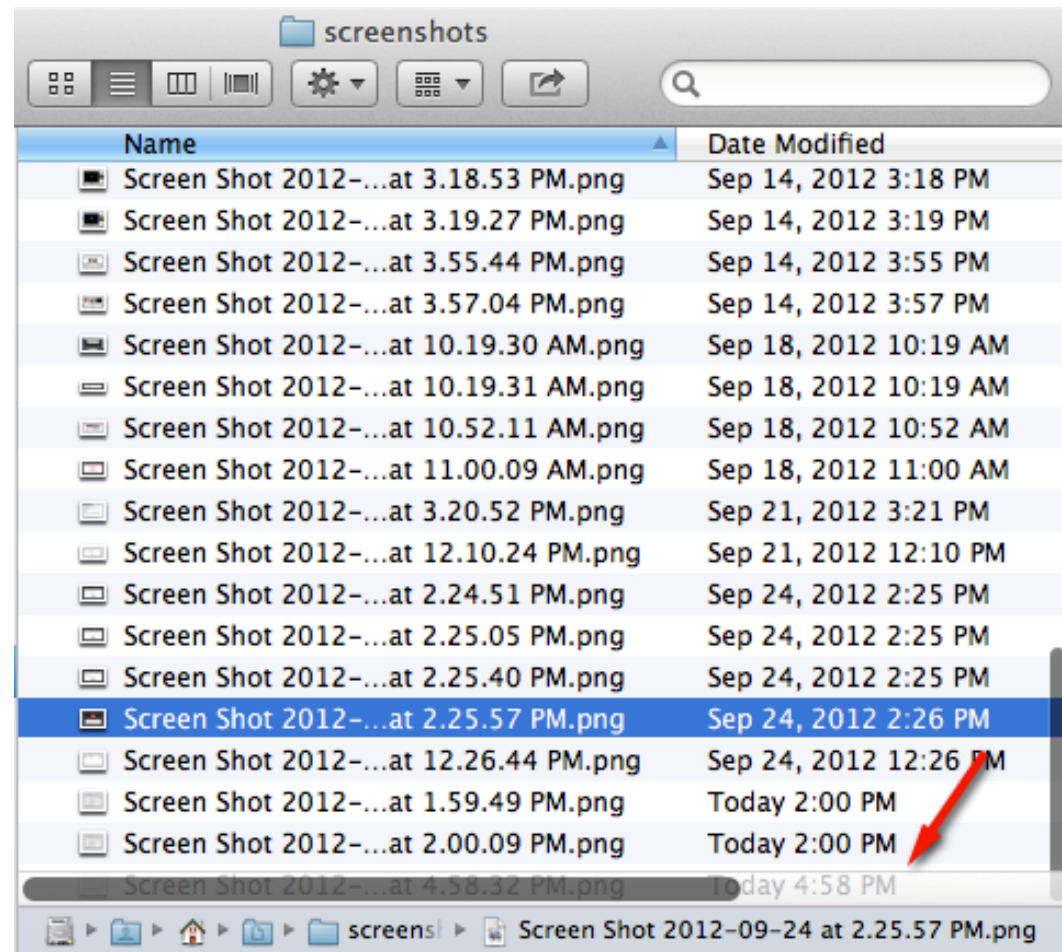
- ▶ Visible structures: Icons, symbols...
  - The appearance must be strictly controlled
  - Positioning must also be similar
    - Ensures what people learn is valid across the app/webpage
    - Improves learn ability

# 5<sub>1</sub>. Consistency. *Levels of Consistency*

## ▶ Invisible structures

- If implemented, make them strictly consistent everywhere
- In any case, using invisible structures (hello Microsoft, hello Apple) just makes their use obscure and difficult
  - Expecting the user will google for learning your product features is not the solution

# 5<sub>1</sub>. Consistency. *Levels of Consistency*



# 5<sub>1</sub>. Consistency. *Levels of Consistency*

- ▶ Interpretation of the user behavior:
  - Never change the meaning of an habitual action
    - It is one of the worst things you can do to the user
    - User take a long time to learn things
    - Such actions become subconscious with time
      - E.g. Changing a learnt gesture is extremely frustrating

## 5<sub>2</sub>. Consistency. *Induced inconsistency*

Make objects different if they act different

- ▶ E.g. if the trash can is destroying the document, make it appear different than a trash can

If your app/webpage has changed substantially, design can be changed to enforce this fact

- ▶ Otherwise the user might not notice and continue using the app/webpage the same way (and it might not work)

## 5<sub>3</sub>. Consistency. *Induced continuity*

Over time, strive for continuity, not for consistency

- ▶ If your renewed app does a lot of different things and the look and feel is exactly the same, users will use it the same old way

New versions of products may change big areas (e.g. new features...)

- ▶ Make them slightly different from previous version
- ▶ Previous knowledge may serve the user to guide their path
  - E.g. maintain the familiar look: same button icons for the things that have not changed...



## 5<sub>4</sub>. Consistency. *With user expectations*

If users expect something in a certain way, do not force them to learn a new way

- ▶ Even a new button/task may have some user expectations
- ▶ If all the users expect the same, and it is different from what you are offering, go for the path of minimum resistance
- ▶ It doesn't matter how fine a logical argument you can put together for how something should work.

Unless your new way offers clear advantage

(p.e. Xerox Drag rule)



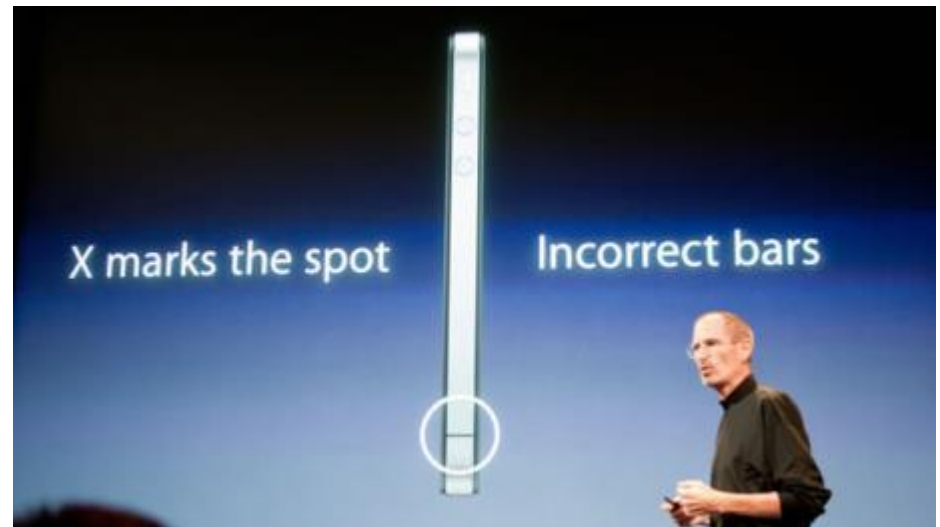
## 5<sub>4</sub>. Consistency. *With user expectations*

- Recall iPhone 4 death grip. The problem:



## 5<sub>4</sub>. Consistency. *With user expectations*

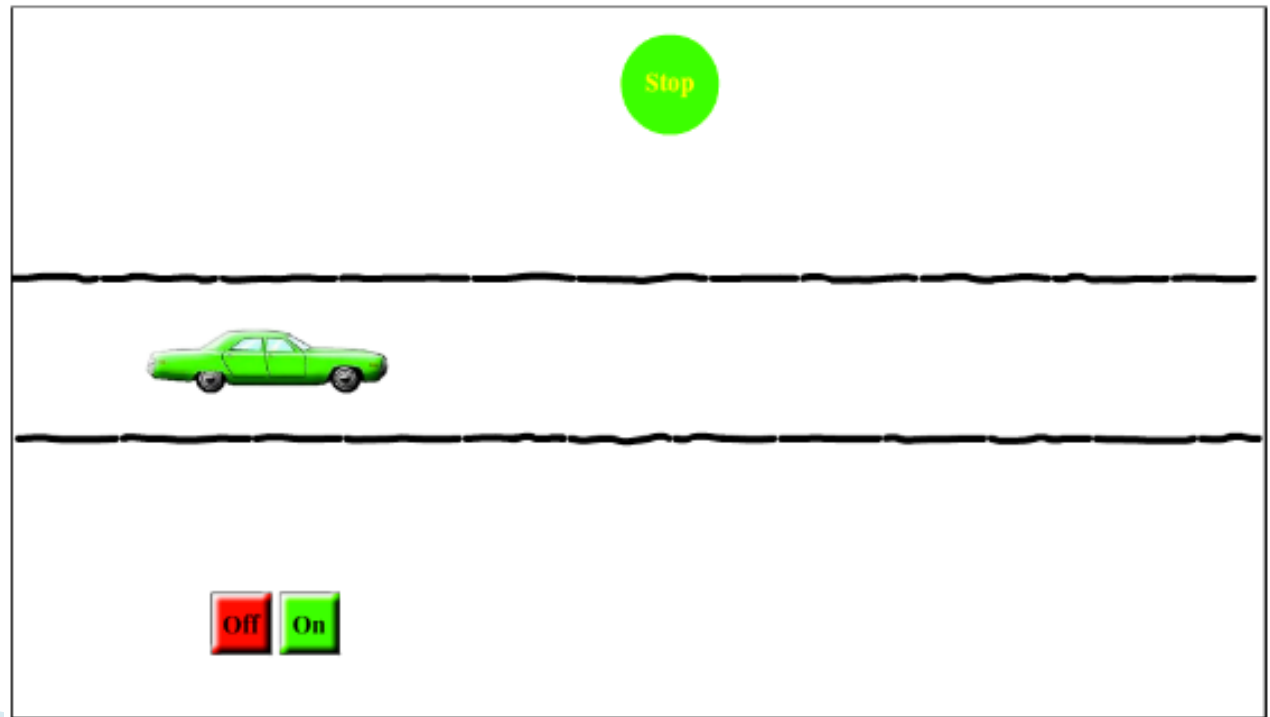
- ▶ Recall iPhone 4 death grip: The excuse and first solution
  - Issue a fix that increases the bars being shown!!!



## 5<sub>4</sub>. Consistency. *With user expectations*

Green and red colors have *meaning*

- ▶ Using them inconsistently leads to larger reaction times



## 6. Default values

Avoid the cursor appear in unpredictable positions

Should be easy to rewrite

- ▶ E.g. automatic selection of the default text on a field

Not all fields require a default value

- ▶ If no clear winner/advantage, do not put a default value at all

# 7. Discoverability<sub>1</sub>


Any attempt to hide complexity will serve to increase it

- ▶ Generating the illusion of simplicity actually does not simplify things
  - E.g. invisible Mac scroll bars

If the user cannot find it, it does not exist

- ▶ Note the difference between the buyer and the actual user

Use active discovery offer (not tested) features to the user

- ▶ Guide people to more advanced features
  - ▶ Mention a feature that exists
  - ▶ Recall it at intelligently spaced intervals
  - ▶ Stop mentioning it once explored or adopted
- 

# 7. Discoverability<sub>2</sub>

All controls should be visible and not over the content area

- ▶ Some exception only if space is limited (e.g. smartphones or tablets)
- ▶ Should provide a standard trigger that will expose all controls
  - Don't do this in desktop
- ▶ Communicate your gestural vocabulary with visual diagrams
  - On-screen hints or training required

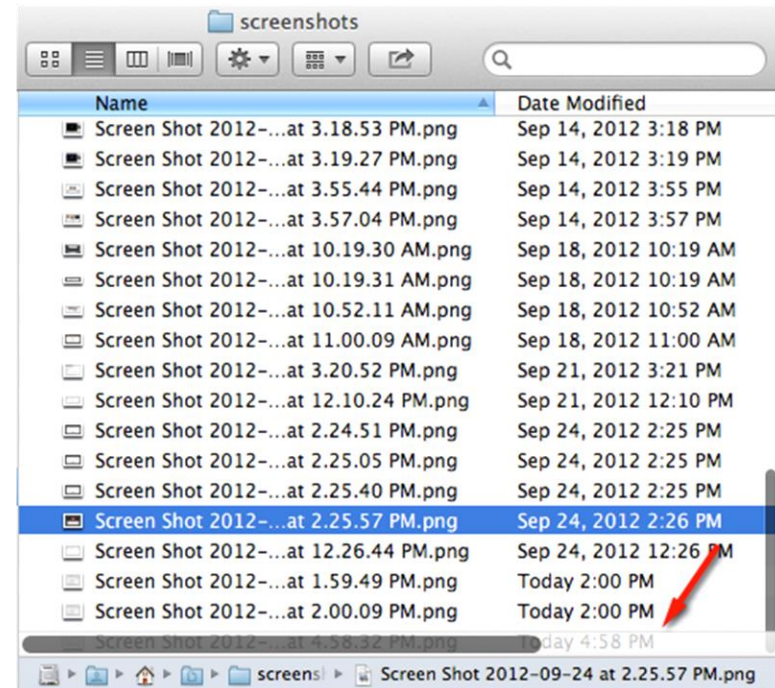
User test for discoverability





# 7. Discoverability<sub>2</sub>

- ▶ Apple example: Invisible scroll bars
  - Scroll bars serve two different purposes:
    - Informing
    - Navigating
  - Making them invisible does not inform the user on the relative position
    - Or where there is more content to be explored
  - Even worse: Apple renders scroll bars over the contents!
    - Making them occlude or prevent their selection





# 8. Efficiency<sub>1</sub>

Look at the user's productivity, not the computer's

- ▶ Keep the user occupied:
  - Typically the highest expense by far in a business is labor cost
- ▶ Maximize every user efficiency
  - Don't improve IT's productivity by pushing work to users
  - Think organization-wise
- ▶ The great efficiency breakthroughs in software come from fundamental architecture changes
  - Not in the surface design of the interface

# 8. Efficiency<sub>1</sub>

Look at the user's productivity, not the computer's

- ▶ Don't ask for the same information twice
  - It is very annoying and unproductive
  - Store it for future use
    - Sometimes not done due to programmers laziness
- ▶ Gather the information as needed

# 8. Efficiency<sub>1</sub>

Look at the user's productivity, not the computer's

- ▶ Provide default values when possible
  - May accelerate interaction
  - May help novice users
  - Provides tips for valid values, formatting
  - Sometimes the answers will be valid in multiple cases
    - Default folder for program installation

# 8. Efficiency<sub>2</sub>

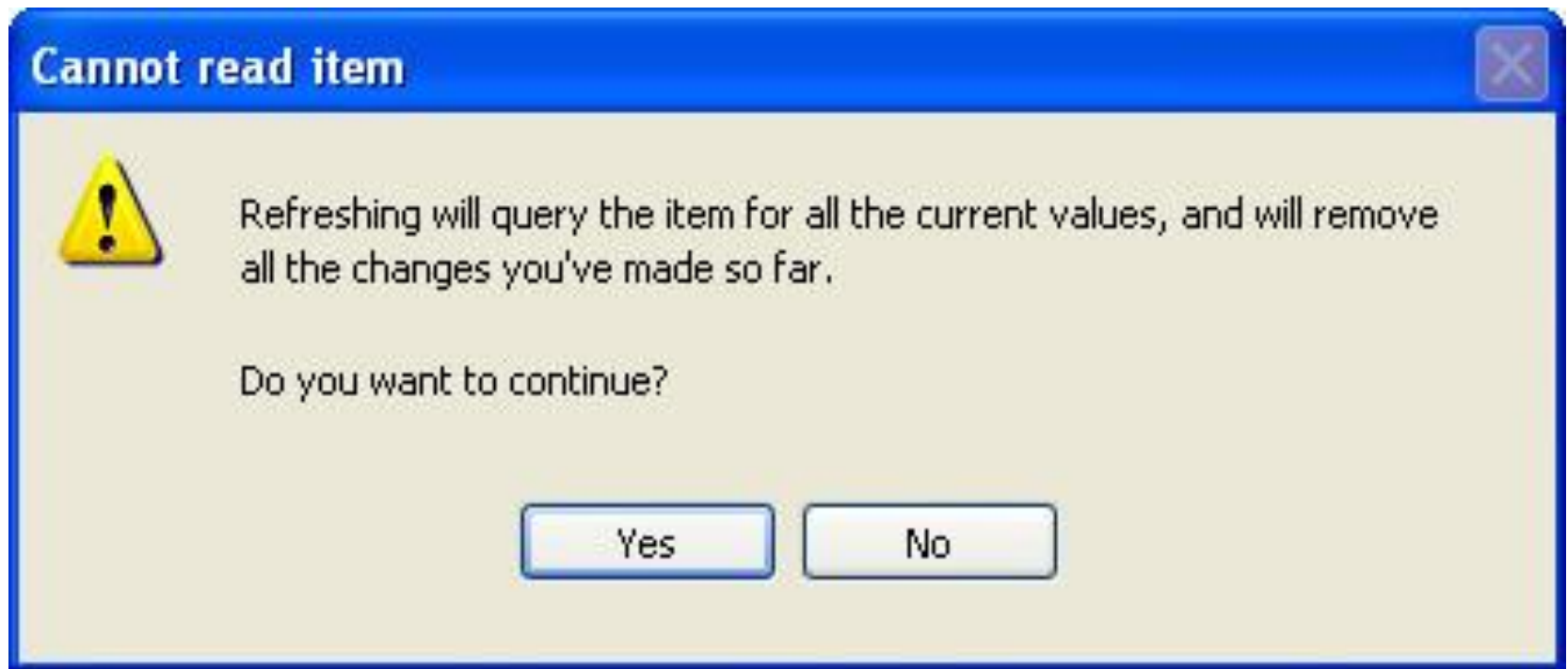
## Error messages should actually help

- ▶ Explain what's wrong
- ▶ Tell the user specifically what to do about it
- ▶ Leave open the possibility the message is improperly being generated by a deeper system malfunction

“Error-1264” does not mean anything to the average user

## 8. Efficiency<sub>2</sub>

Error messages should actually help



# 9. Explorable interfaces

Users want to feel free when exploring interfaces

- ▶ Two important principles that facilitate exploration are:
  - Make Actions reversible
    - E.g. “Back” in a webpage, cancelling long actions...
    - Promotes exploration
    - Otherwise, a perfect user is a slow user
  - Always allow “Undo”
    - Otherwise you need to confirm everything
      - Again, slow
  - Users should always have an easy way out

# 10. Fitts's Law

The time to acquire a target is a function of the distance to and the size of the target


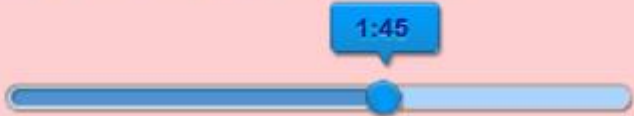
- ▶ Large objects for important functions
- ▶ Small objects for functions you would prefer users not perform
- ▶ Reduce the number of targets to acquire
  - Not only their distances

More on this later...



# 11. Informing users<sub>1</sub>

Keep users informed when they face delay

Expected Delay	Indication
1/2 to 2 seconds	Use animated mouse cursor or other "busy" indicator 
> 2 seconds	Tell them potential length of wait
> 5 seconds	Use an animated progress indicator  Process must end by the time indicator is full!
> 10 seconds	Keep users a) informed & b) entertained
> 15 seconds	Same as >10 plus add at end a noticeable sound & strong visual indication so users know to return



# 11. Informing users<sub>2</sub>

Acknowledge all button clicks by visual clue within 50 ms

Start making everything faster

- ▶ Eliminate any element of the application that is not helping
- ▶ Be ruthless

Wearables come with an even higher level of expectation:

- ▶ No one waits to see what time it is
- ▶ Or to see who is calling, what the temperature is outside...



# More principles for usability

Choose metaphors that will enable users to instantly grasp the finest details of the conceptual model

Try making your concepts visually apparent in the software itself

- ▶ Buttons are pressed, sliders dragged...

Expand beyond literal interpretation of real-world

- ▶ If a metaphor is holding you back, abandon it



# More principles for usability

Metaphors. iOS browser vs iOS compass



# More principles for usability

## Ensure that users never lose their work

- ▶ This principle is all but absolute
- ▶ Users should not lose their work as a result of
  - error on their part,
  - the vagaries of Internet transmission,
  - or any other reason other than the completely unavoidable (e.g. travel sites)

Professors d'IDI - UPC

# IDI – Design Principles




# Bonus track: The case of Apple

- ▶ Jonathan Ive (Apple's Chief Design Officer) is a fan of Bauhaus school's Dieter Rams minimalistic design
- ▶ Dieter Rams worked for many years for the German company Braun
  - Responsible of many of its simple-to-use and aesthetically pleasant product designs

# Bonus track: The case of Apple

## Dieter Rams' principles of good design

1. Innovative
  2. Makes a product useful: usefulness is essential
  3. Aesthetic but well-executed
  4. Makes a product understandable
  5. Unobtrusive
  6. Honest
  7. Long-lasting
  8. Thorough down to the last detail
  9. Environmentally friendly
  10. As little design as possible
- 

# Bonus track: The case of Apple

- ▶ “As little design as possible” is the 10<sup>th</sup> principle!!!
  - Can be applied after the other ones
- ▶ Apple has sacrificed most of the qualities of its well-known usable design to this 10<sup>th</sup> principle



# Bonus track: The case of Apple

## ▶ Apple has sacrificed usability to aesthetical simplicity:

- from “easy-to-use, easy-to-understand” to “no manual included but it is necessary”
  - Tiny unreadable fonts
  - Lack of undo
  - Hiding/removing fundamental features
  - Lack of consistency in mouse/trackpad behavior
  - ...
- Read:  
<https://www.fastcodesign.com/3053406/how-apple-is-giving-design-a-bad-name>

# Bonus track: The case of Apple

- ▶ E.g. Today's iPhones and iPads are a study in visual simplicity.
  - Beautiful fonts (helvetica)
  - A clean appearance, uncluttered by extraneous words, symbols, or menus
    - Thin fonts with little contrast

1 / 2 iOS 7 preview: horrible type. Low foreground/background contrast & lighter weight Helvetica trending illegible.

— Thomas Phinney (@ThomasPhinney) June 13, 2013

2 / 2 Existing iOS Helvetica UI font was already anti-legibility. iOS 7 choices could make me run for the hills.

— Thomas Phinney (@ThomasPhinney) June 13, 2013

# Bonus track: The case of Apple

- ▶ E.g. Today's iPhones and iPads are a study in visual simplicity.
  - Beautiful fonts
  - A clean appearance, uncluttered by extraneous words, symbols, or menus
    - Thin fonts with little contrast
  - So what if many people can't read the text? It's beautiful.
    - People require to use Apple's assistive tool even if they do not have defective vision
    - Turns out that then the fonts are too big in many apps

# Bonus track: The case of Apple

E.g. No perceived affordance. iOS examples:

## UNDO

- **Disappeared!**
- People complained
  - Then, put undo back in...
  - ... if you violently shake your phone or tablet
  - But undo is not universally implemented, and there is no way to know except by shaking
  - Even then, you don't know if you didn't shake properly or undo is not implemented for that particular situation

# Bonus track: The case of Apple

No perceived affordance. iOS examples: UNDO

- **Disappeared!**
  - Sort of now by only shaking the device !!!???
  - Tablet shaking? WTF?
  - Not universally implemented!
- Touch-sensitive screens, especially on relatively small devices, offer multiple opportunities for things to go wrong when an active link or button is accidentally touched
- Back button should be universally available

# Bonus track: The case of Apple

## ► Use of modes:

- The ultimate in simplicity is a **one-button controller**: very simple, but because it has only a single button, its power is very limited unless the system has modes, e.g.:
  - One click → selection
  - Double click → erase
  - Triple click → undo
  - Long click → redo



# Bonus track: The case of Apple

- ▶ E.g. mouse in different devices and situations

...zooming on a trackpad involves putting pulling your thumb and one finger together in a pinching motion to make things smaller, or pushing them apart to make things larger.

Double-tapping on the trackpad with two fingers will also *zoom within some apps*, and *repeating the tap will zoom out again*.

This double tap feature also works on the Magic Mouse, just **remember** not to click the top, just tap the middle of the device instead....”

# Bonus track: The case of Apple

E.g. No perceived affordance. iOS examples.  
GESTURES

- The absence of any signals on the screen (signifiers, affordances) makes gestures undiscoverable
  - Swipe up or down, left or right, use one finger or two or three or four or five, or tap once, twice, or three times, or tap long or short?
- Users must memorize these gestures after first being told, "read the manual" (what manual?) or discovering them by accident.



# Bonus track: The case of Apple

## E.g. Hiding fundamental features

- E.g. the lock rotating screen is difficult to understand:
  - It is grayed or not, but difficult to interpret its meaning
  - Turns out that Apple uses text to say which, but in tiny little letters somewhat removed from the icon itself
- Frequent operations should not require so long to be learnt

# Bonus track: The case of Apple

## ▶ Other examples:

- iOS keyboard not showing capitalization
  - The design decision that when a keyboard is in uppercase mode, it should display uppercase letters and, when in lowercase mode, it should display lowercase letters, is so obvious that the failure to provide this simple feedback on the current mode defies all credulity
- It took Apple 9 versions of the OS to fix it!!!

# Bonus track: The case of Apple

- ▶ Products that *inspired* Apple



# Bonus track: The case of Apple

- ▶ Braun products that inspired Apple products

BRAUN T3 POCKET RADIO  
(1958)



VS

APPLE IPOD  
(2001)



# Bonus track: The case of Apple

- ▶ Braun products that inspired Apple products



# Bonus track: The case of Apple

- ▶ Braun products that inspired Apple products



# Other UX errors

- ▶ **Leave the UX for too late:**
  - Maximum benefits are achieved when UX introduced early in the development process
- ▶ **Leave the user feedback for too late:**
  - Unfinished versions or prototypes of the product can be tested