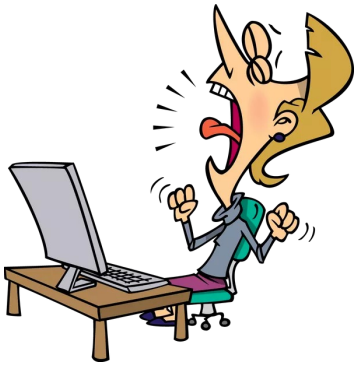
#4 Human-Computer Interaction



Outline

- Previous lectures
  - What is HCI & HCI framework
  - Design, HCI artifacts, design thinking, design methodologies
  - Modeling: cognition, mental, conceptual, activities, prototyping
- This lecture
  - What is a principle?
  - What is a design principle?
    - There are many principles that must be considered when designing interactive systems
    - We will present some of them
    - Have incorporated short videos for some (some more than others) so you can refer to them later again

Outline of today's topic

- What is a principle?
- What is a design principle?
- There are many principles that must be considered when designing interactive systems
  - We will present some of them
  - Have incorporated short videos for some (some more than others) so you can refer to them later again

Cognitive Design/Engineering

https://www.youtube.com/watch?v=NK7aaO7ixU8

Design Principles

What are design principles?

"Design principles are widely applicable laws, guidelines, biases and design considerations which designers apply with discretion.

Professionals from many disciplines—e.g., behavioral science, sociology, physics and ergonomics—provided the foundation for design principles via their accumulated knowledge and experience."

www.interaction-design.org

Perceptual principles: Gestalt laws of grouping

- Closure, common fate, continuity, proximity, similarity, symmetry

https://www.youtube.com/watch?v=LlzuJqZ797U

80 / 20 Rule

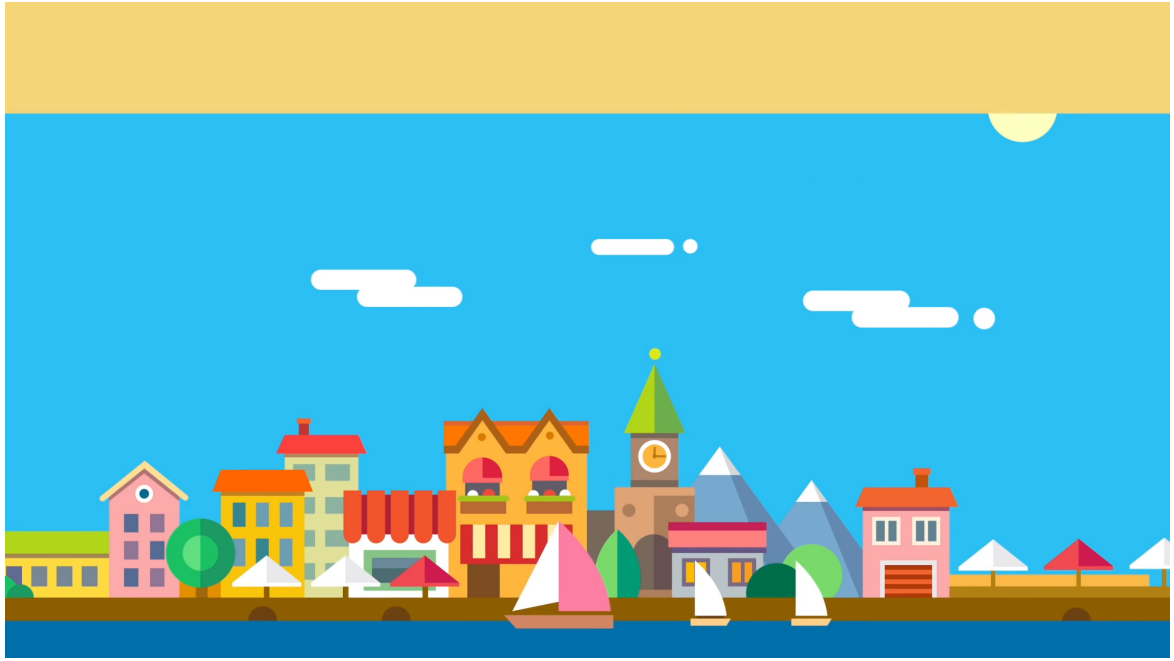
Approximately *80* percent of the effects generated by any large system are caused by *20* percent of the variables

- observed in all large systems, including those in economics, management, user interface design, quality control, and engineering

- A few examples:
  - ***80%*** of a product's **usage** involves ***20%*** of its **features**
  - ***80***% of **errors** are caused by ***20%*** of the **components**

80/20 Rule

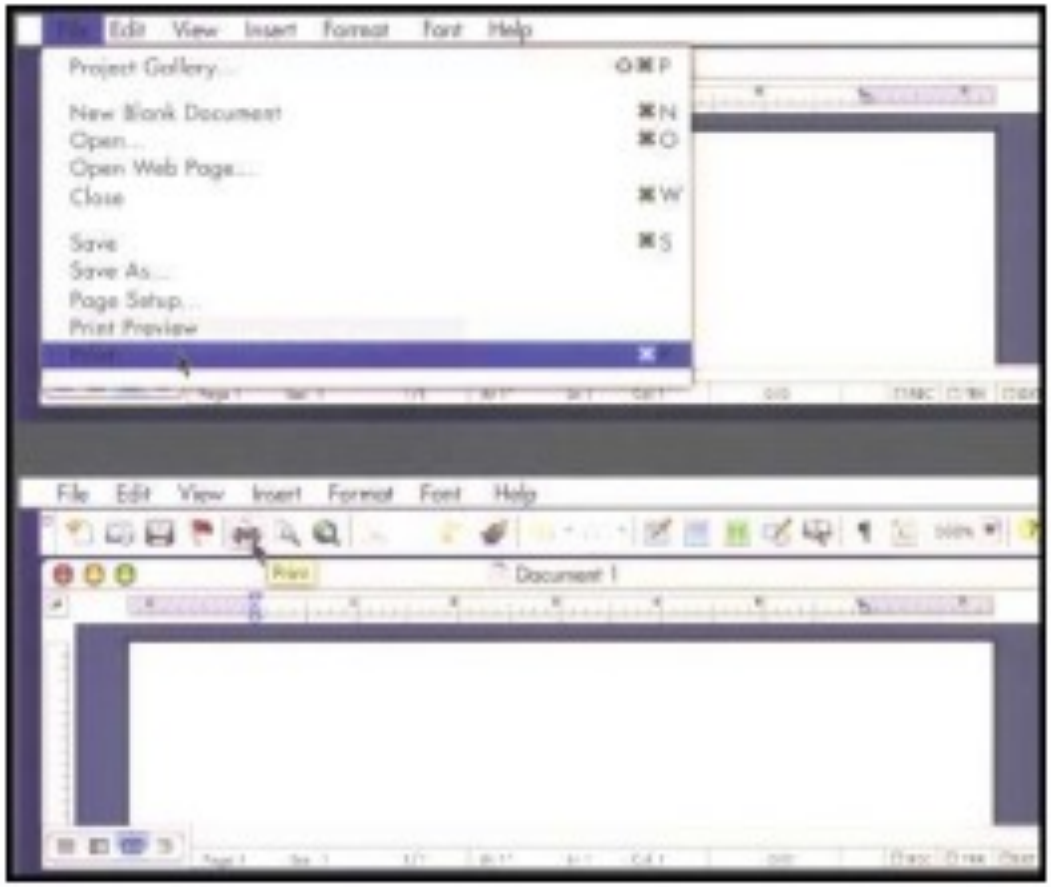
- Useful for focusing on resources
  - Design evaluation
- Use the 80/20 rule to:
  - Assess the value of elements of your system
  - Decide what aspects of your system need redesign



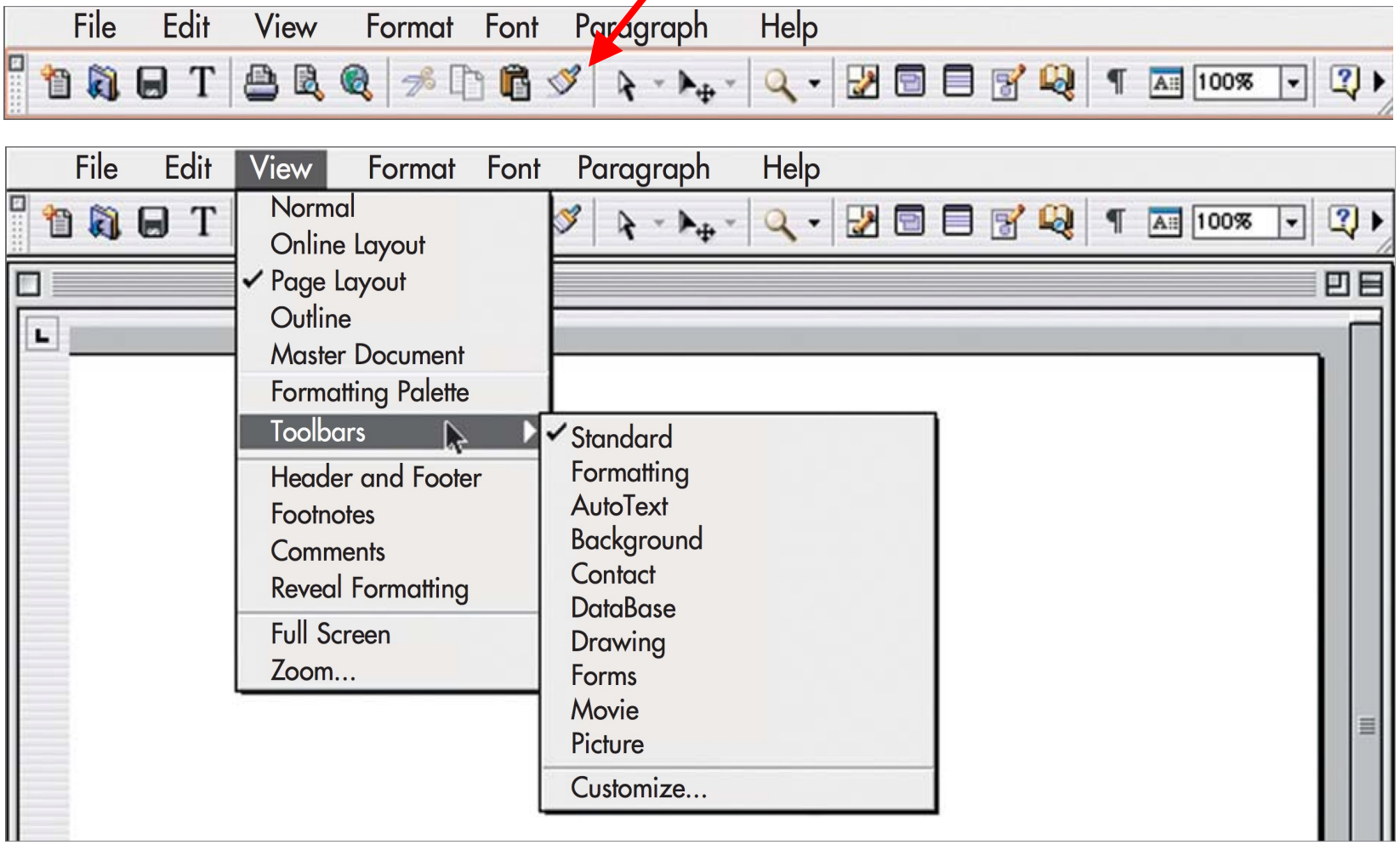
https://www.youtube.com/watch?v=zPoA6dzKmtg

80/20 rule

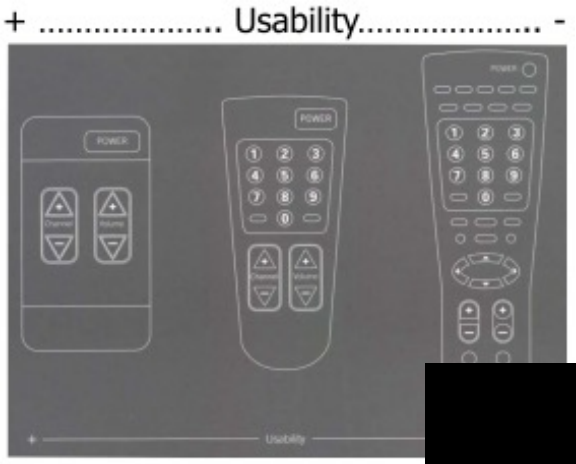
- E.g.,
  - Keep 20% visible, make 80% layered or in menus



critical 20% of functions made readily available



Flexibility-usability tradeoff



The less functionality, the simpler and easier
The more functionality, the more flexible / complex

https://www.youtube.com/watch?v=x6rhqDYzZAo

Knowledge in the world

- Externalization of information so that it is accessible to our senses
- Why?
  - Remembering things is not easy
  - People organize and structure the environment so as to access and recall them easier
  - Knowledge in the world acts like our external memory
    - Remember: Distributed Cognition

Knowledge: Internal vs. External

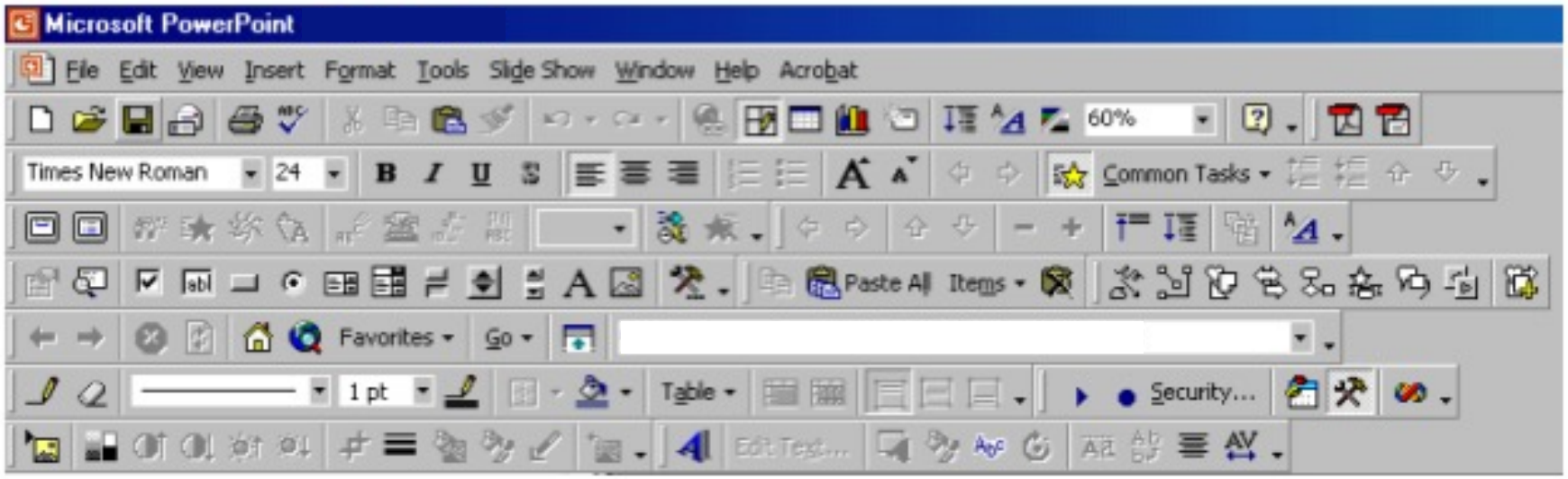
- How do I create a template in PowerPoint?
- How do I find how much disk space I have used in Windows or on the Mac?

Knowledge in the world: Recall

When needed information is readily available in the world (outside the user's head), the need for recall and figuring out what to do diminishes

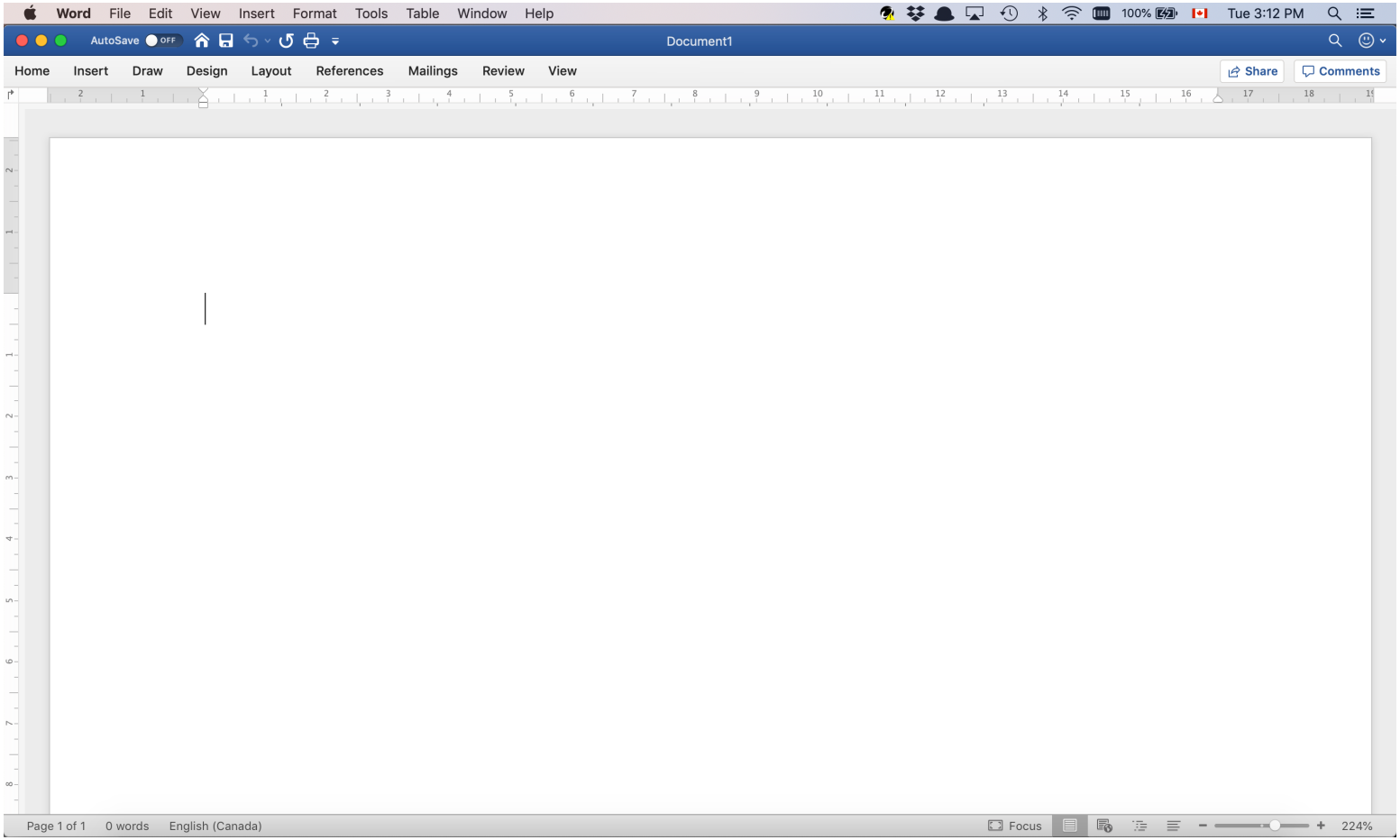
Knowledge in the world: Tradeoff

- All toolbars activated in an old version of PowerPoint
  - Too much externalized information



Knowledge of the world: Tradeoff

- How do I create a template?
  - Information is in the world somewhere, but hard to find



Principle: Externalization of information

Designer must decide what the balance between internalization and externalization of information should be

What can I do with this object?

Visual appearance

- Visual appearance indicates how the object should be used (*could* be used?)
- Chair
  - For sitting
  - For picking up
  - For standing on
  - For….?

Visual appearance

- Table
  - For placing things on
- Knobs
  - For turning
- Buttons
  - For pushing
- HCI artifacts
  - For ??

Visual appearance

- What do these ideas have to do with computers and this course?
- What does the visual appearance of an HCI artifact tell me?

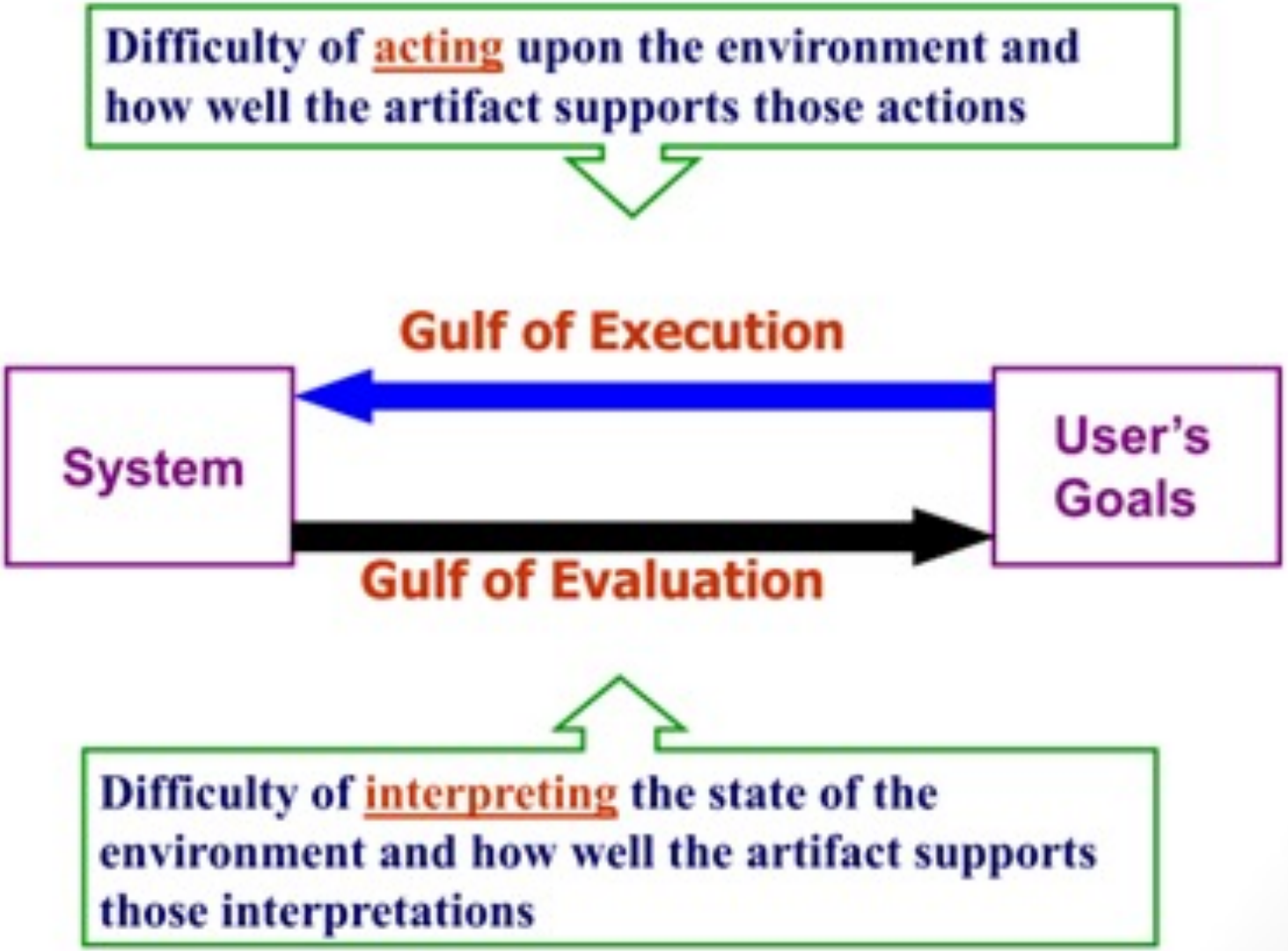
- Computers are representational systems, using complex visual signs to represent real- and artificial-world objects and situations, letting people interact with them
  - It is all about helping users
    - Do/act
    - Think/know
    - Feel/experience

Visibility…

Systems are more usable when they **clearly indicate and make perceptible** their ***status***, the possible ***actions*** that can be performed, ***how*** those actions can be used, and the ***consequences*** of the actions once performed

- An interface feature is visible if
  - It is accessible to human sense organs (usually the eyes)
  - Or accessible to other sensory modalities
- Vision, in particular, plays a central role in human thought and understanding
  - So make important objects visible

Visibility…


Difficulty of acting upon the environment and how well the artifact supports those actions
Gulf of Execution
System
User's Goals
Gulf of Evaluation
Difficulty of interpreting the state of the environment and how well the artifact supports those interpretations

Visibility: a ‘must’

- Relation between user’s
  - Intentions
  - Required actions
  - Results

  - Must be:
    - Sensible
    - Non-arbitrary
    - Meaningful

Non-visibility

- If an interface forces you to do the following, that feature or method is NOT VISIBLE (in an HCI sense)
  - Memorize the fact that the feature exists
  - Search around the interface until, by luck and perseverance, you reach a sequence of actions that activates the feature
  - Seek help to discover how to perform an operation (e.g., use manual or help system)

Visibility

- Is more than detectable existence
- An object may exist but be
  - too small to be noticed
  - indistinguishable from other objects
- Concern should be with perceptual and cognitive properties of interfaces

Visibility: Recognition vs recall

- People are better at recognizing objects (or solutions), when selecting from a set of options, than recalling them from memory
  - E.g.,
    - "Is Ottawa the capital of Canada?" vs. "What is the capital of Canada?"
- When it comes to the design of complex systems, the principle of visibility is perhaps the most important and most violated principle of design

Visibility principle

- Good design should make necessary options and information for a given task visible without overwhelming and distracting the user with extraneous or redundant information
- In other words, good design doesn't:
  - Overwhelm users with unnecessary alternatives
  - Confuse them with unneeded information

How to accommodate visibility

- Transparency
  - Each function and method of operating must be apparent by merely looking at it
- Function
  - Appearance of controls should suggest how they are used (pulled, pushed, dragged, etc…)
- Brings us to the concept of affordance

Affordance

- Gibson (1977) – "The Theory of Affordances" & "The Ecological Approach to Visual Perception"
  - "All action possibilities latent in an environment, objectively measureable and independent of the individual's ability to recognize them, but always in relation to the actor and therefore dependent on their capabilities."
  - For example, a set of steps which rises four feet high does not afford the act of climbing if the actor is a crawling infant

Affordance

- Norman (1988) – "The Psychology of Everyday Things"
  - "the term affordance refers to the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used."
  - A chair affords support and, therefore, affords sitting

Affordances

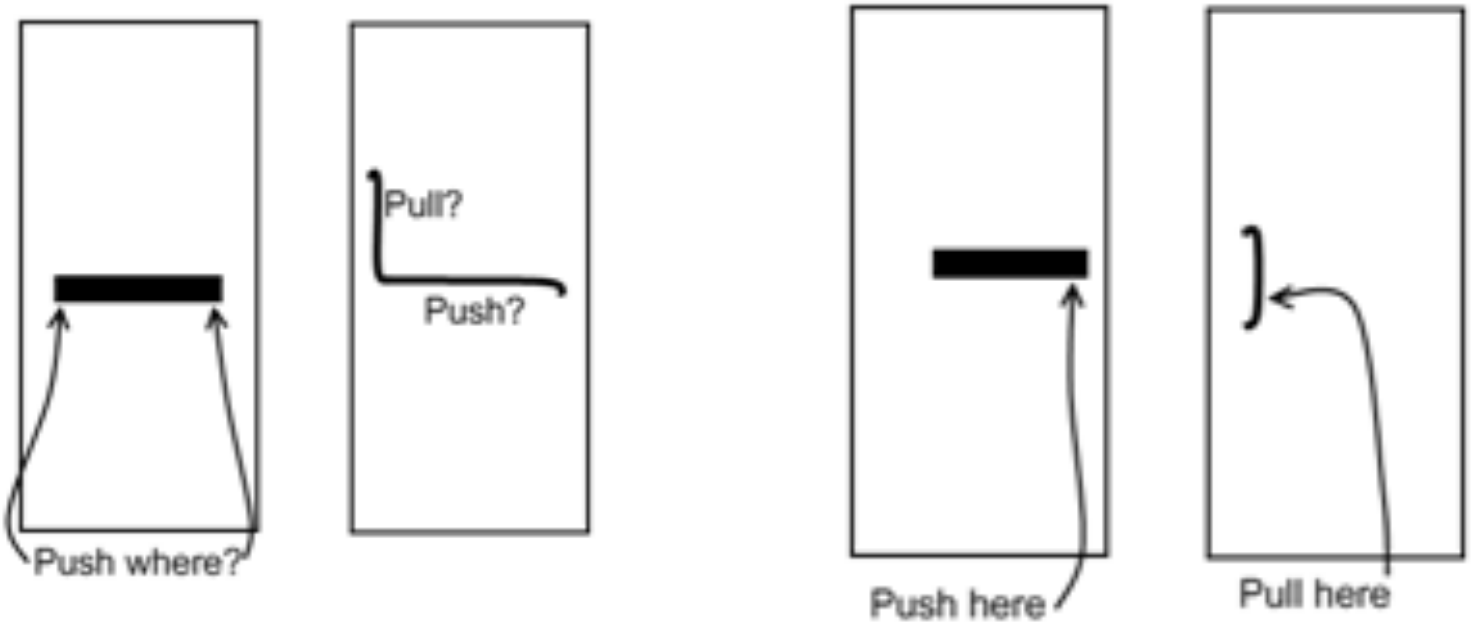
https://www.youtube.com/watch?v=a6F0EYCUjcE

Affordances are *relational properties*—not just of the object, but of the user and object together

Affordances: Dealing with the unknown

- What can you do with these?

Affordances: Good or bad design?

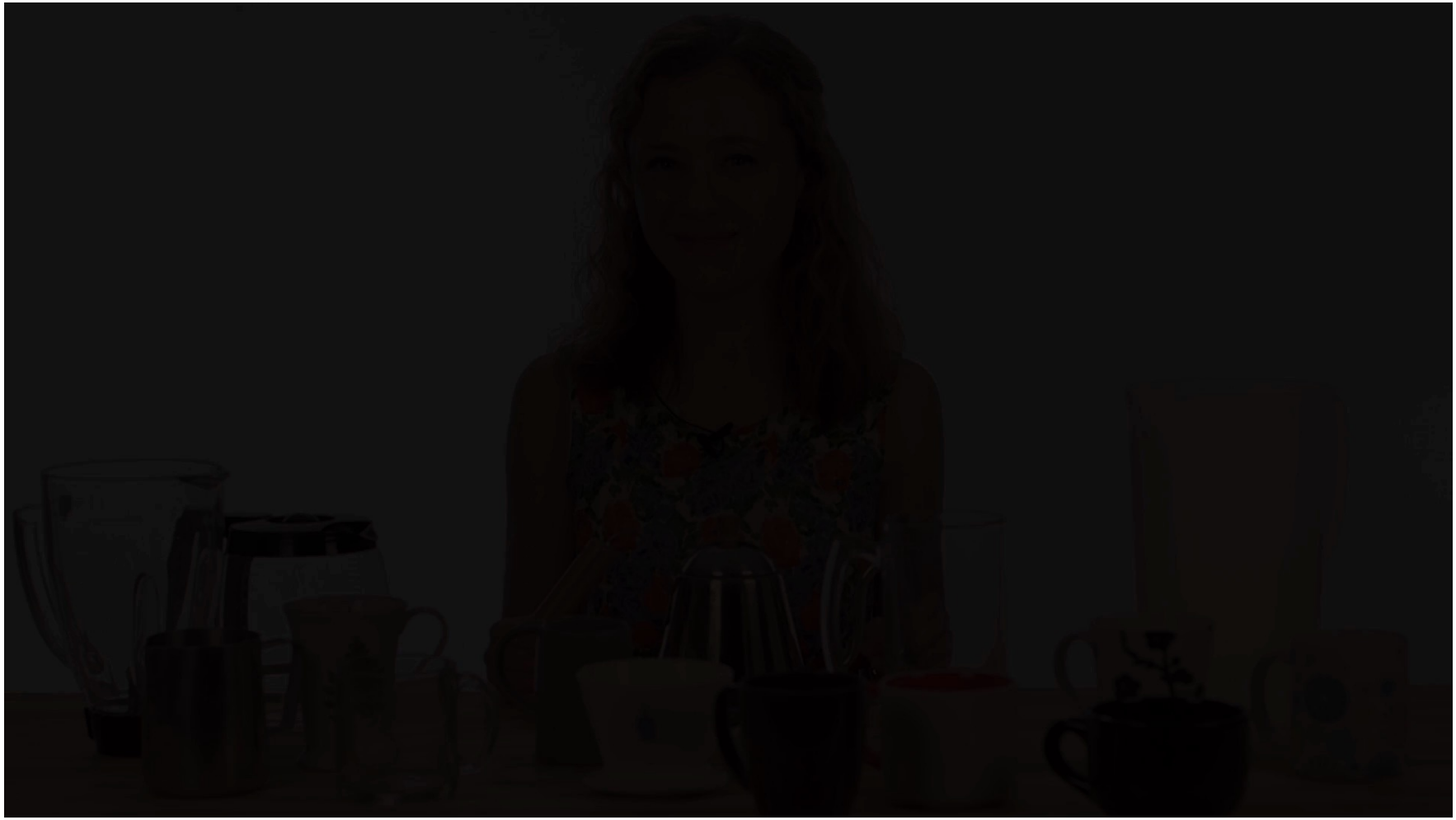


Affordances

- This is poor design!
- Simple things like doors should not need instructions!

Good or bad design?

Affordances



https://www.youtube.com/watch?v=ye5AwaCTdBc

Affordances

- Provide strong cues to the operations of things
- When affordances are taken advantage of, the user knows exactly what to do
  - No need for labels, instruction, etc…

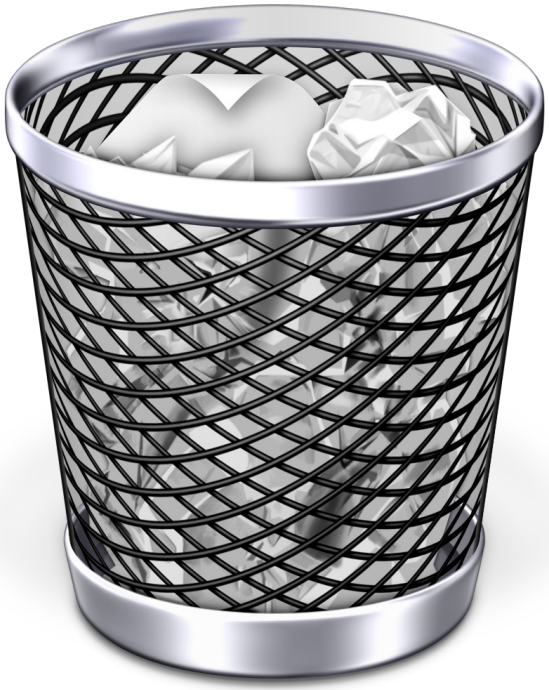
Visibility and affordance

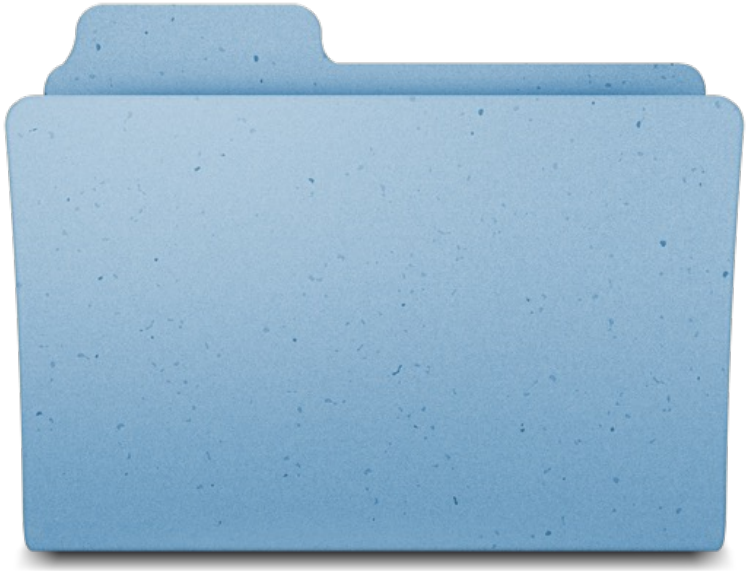
- In constructing or evaluating interfaces, one should always
  - Ask how the user knows that an action is possible
  - Provide each visible feature with a recognizable affordance

➔ Mimic familiar objects and environments in software interfaces to imply the way they can be used.

Principle of familiarity

- Familiarity with forms and behaviours in our environment activates different mental models
- Can suggest function and behaviour of digital things





Visibility, affordance, and skeuomorphism

https://www.youtube.com/watch?v=4ANoLWML8BU

Affordance principle

Complex things may need explaining, but simple things should not.

When simple things need pictures, labels, instructions, *then design has failed*

Affordances

- The perceived and actual properties of
  - An object
  - Situation
  - Environment

  - That determine how people could possibly
    - Use an object
    - Behave in a situation
    - Think in and environment or know what to do

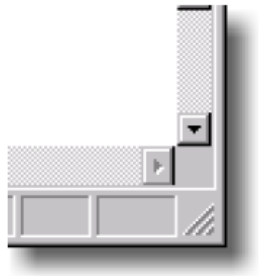
- Generally speaking, we want to make perceived and actual the same

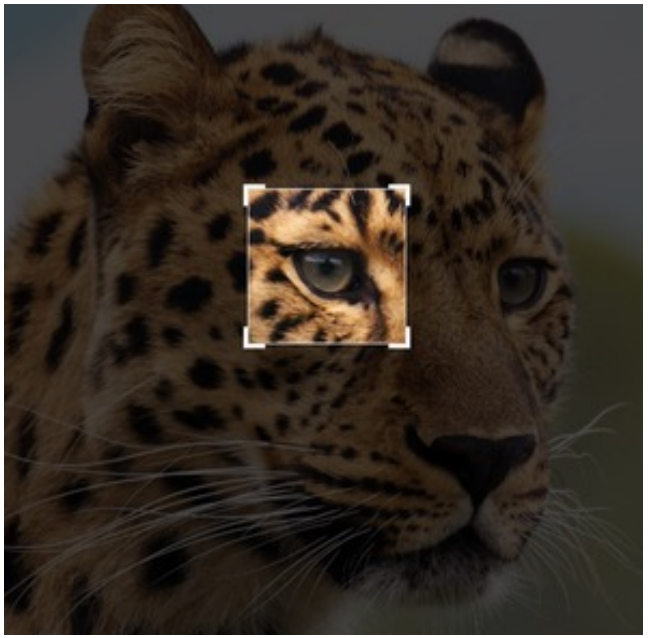
Perceived affordances

- Actual affordance
  - What the user can actually do
- Perceived affordance
  - What the user perceives is possible to do
- What about affordances in digital objects?
- Programmers can make objects do just about anything!
- In digital objects there are no physical affordances, only *perceived ones*
  - In other words, learned conventions

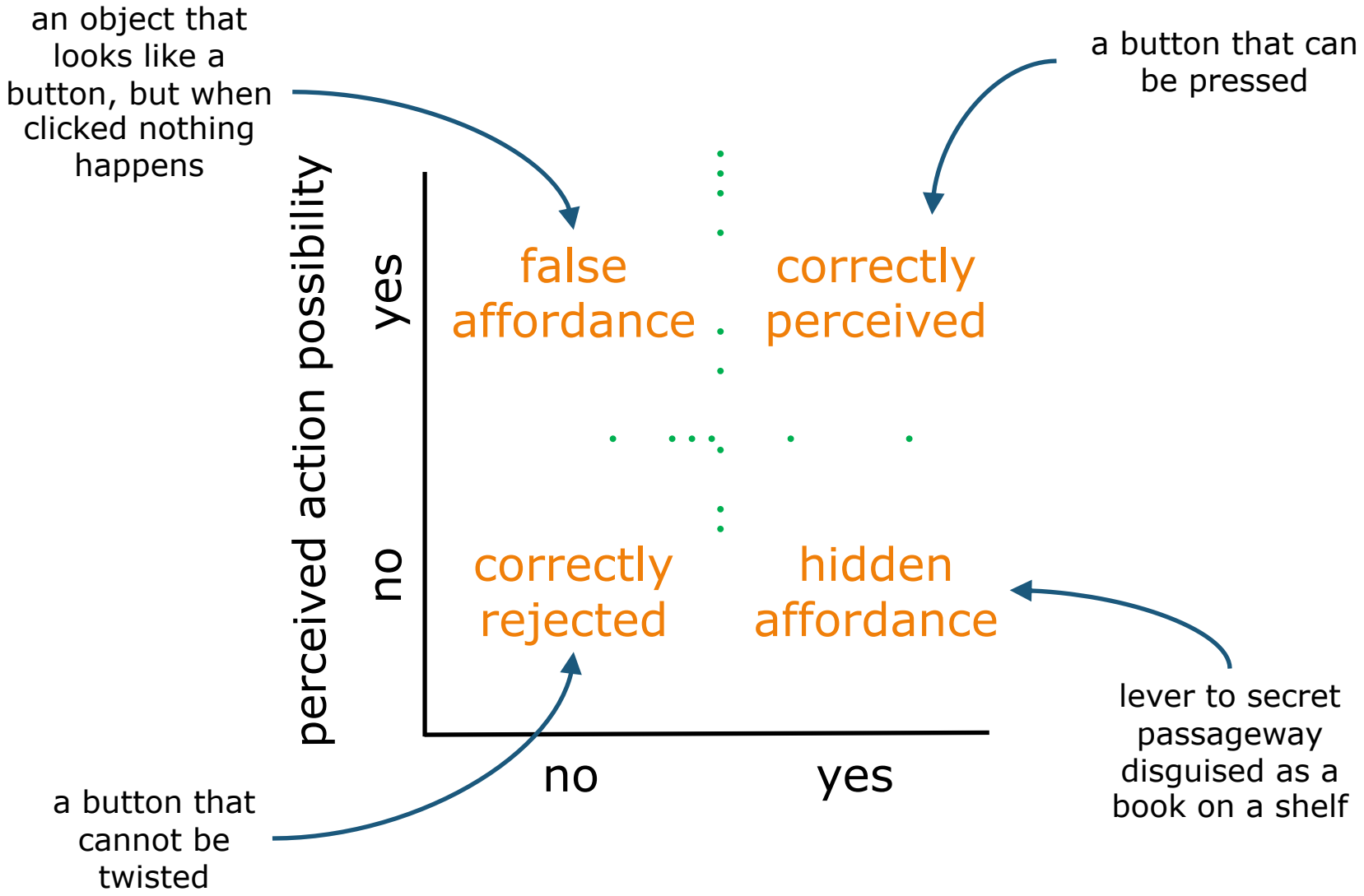
Perceived affordances

- What do these afford? How do you know?
- Is there a reason why they shouldn't be used in some other way?



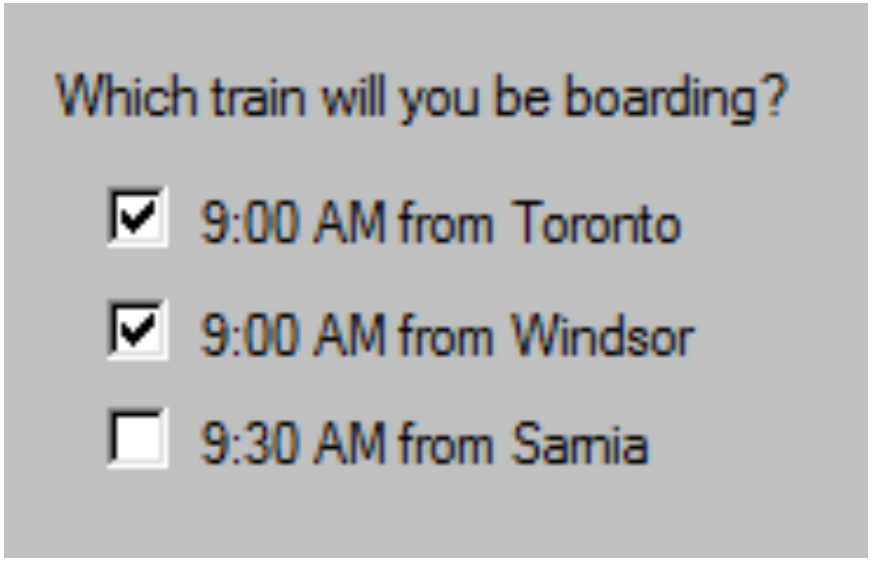


Real vs. perceived affordances



User interface affordance

- Options are mutually exclusive, however interface allows all of them
- Use of checkboxes rather than radio buttons
- Affords what it should not!



User interface affordance

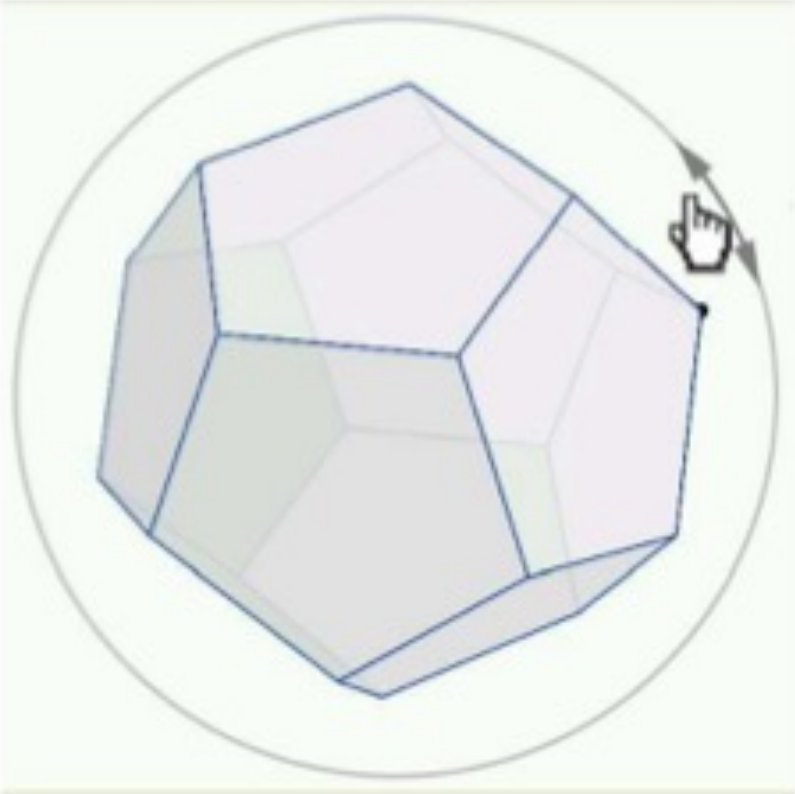
- Setting cache size in an old IE 3.0 – user can only specify a percentage of hard drive
  - Smallest setting is 1%
  - What if HD is 1TB
- Affords what it should not!

Static vs. dynamic affordance

- Static
  - Visual cue is present, but user should interpret it just by looking at it
    - E.g., a button is for pushing
- Dynamic
  - Visual cue is present, but is accompanied by additional 'forward' feedback
    - E.g., a button highlighted by a mouseover

Feedforward

- Cue users about possible actions
- Also sometimes called “dynamic affordances”



Designing affordances

- Remember Mental Models?
  - What is the relationship between mental models and affordances?

- Challenge for designers:

Activate the appropriate mental models so that what the user perceives as the system's affordances is consistent with its actual affordances

Visibility: Make use of affordances

ONCE AGAIN

- To the degree possible, make each function and method of operating the system apparent by merely looking at it
- Appearance of controls should suggest how they are used (pulled, pushed, dragged, etc…)
- Make use of perceived and actual affordances

Affordances in design of online learning

https://www.youtube.com/watch?v=RWJMOqu8L8A

Affordance and flat design

https://www.youtube.com/watch?v=03ANk1Iq1iE

Mini quiz

- Take couple of minutes and jot down everything we have talked about so far today
- What were the key concepts and ideas?

Question

- Do human beings think only about possibilities?
- What else is needed to guide perception and action?

Constraints

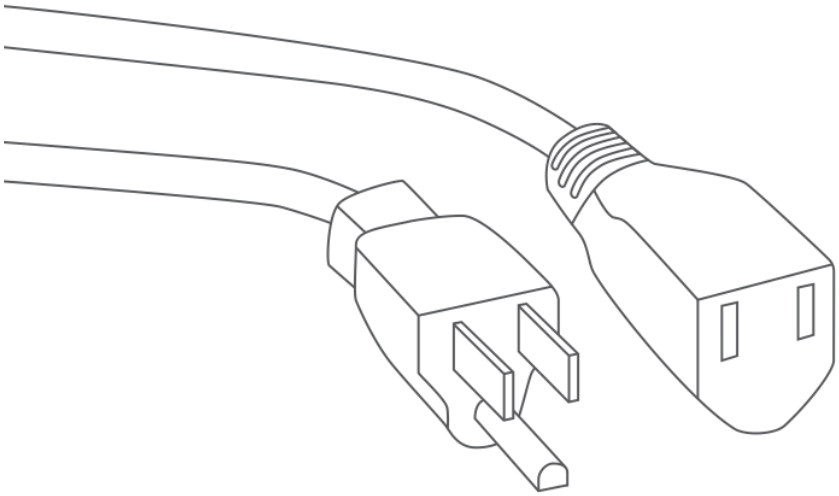
- Limits on action and thoughts
  - Total freedom confuses people
- Constraints and affordance should balance one another
- 2 main types
  - Physical
  - Non-physical

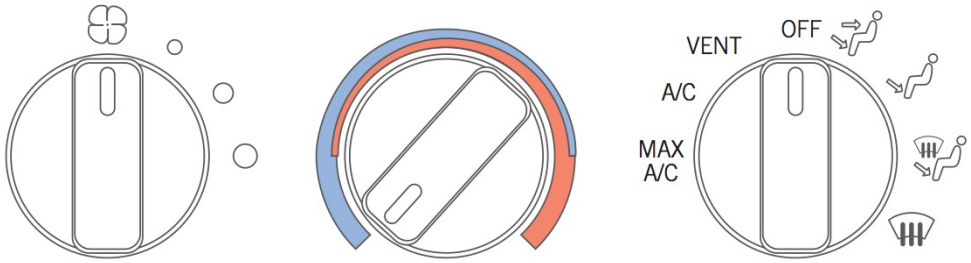
Constraints

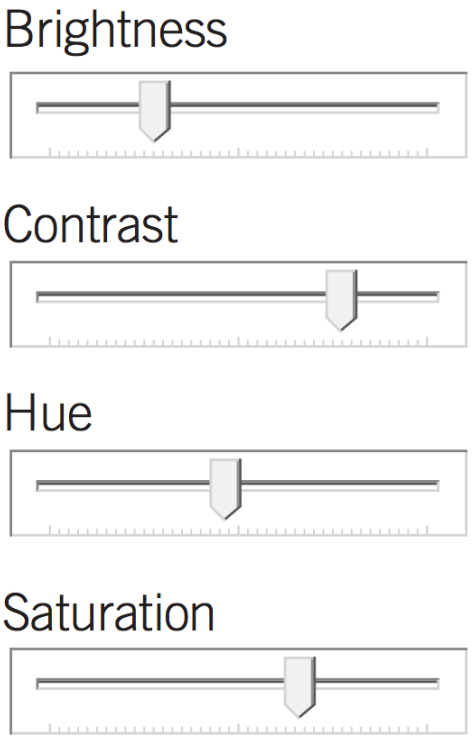
https://www.youtube.com/watch?v=JizQxRpMvUQ

Constraints: Physical

Limitations of the possible actions are caused by the object's form and directing actions in a certain way





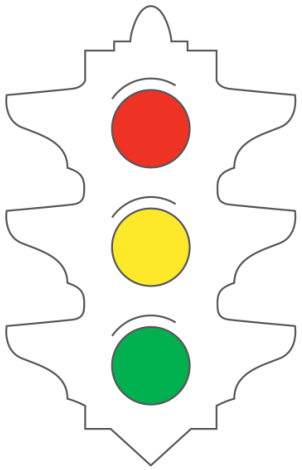


Constraints: Non-Physical

Limit the range of possible actions by leveraging the way people perceive and think about the world

psychological
visible characteristics of something naturally suggest action possibilities

logical
action and behaviour are according to what makes logical sense



semantic/cultural
prior knowledge of a situation and the world determine action

Design Principle: Constraints

Use constraints in design to improve usability and minimize errors. Use physical constraints to minimize unintentional inputs, and prevent or slow dangerous actions. Use non-physical constraints to improve the clarity of a design.

Chunking

- Chunk
  - "a collection of elements having strong associations with one another, but weak associations with elements within other chunks" (Gobet et al., 2001)

- A technique for combining many units of information into a limited number of units (chunks)
  - Chunks are perceived as one unit and processed as one unit

Chunking

Chunk information when users are required to **recall and retain** information, or when information is used for **problem solving**

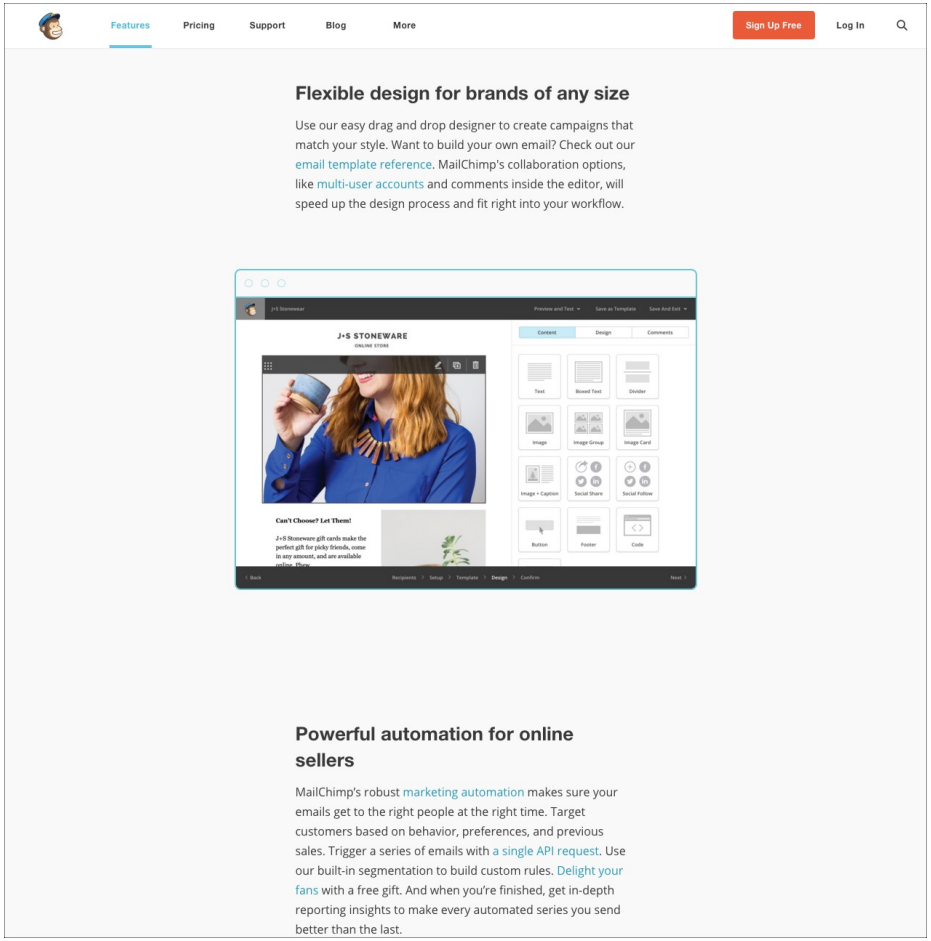
- Use: 4 ±1 chunks

- This principle is often misapplied
- Are the following correct applications of the chunking principle?
  - Constrain the number of items on a menu bar to five or six
  - Place only five or six items in a pull-down menu
  - Apply "The rule of six" to power point presentation or bulleted lists
  - Never have more than five or six radio buttons or check boxes together

Chunk only when tasks are related to memory (recall or working)!

Chunking multimedia content

- See https://www.nngroup.com/articles/chunking/ for more examples:
- "The key to effectively chunking multimedia content (text as well as images, graphics, videos, buttons, and other elements) is to keep related things close together and aligned (in accordance with the Law of Proximity in Gestalt psychology). Using background colors, horizontal rules, and white space can help users visually distinguish between what's related and what isn't."



Mapping

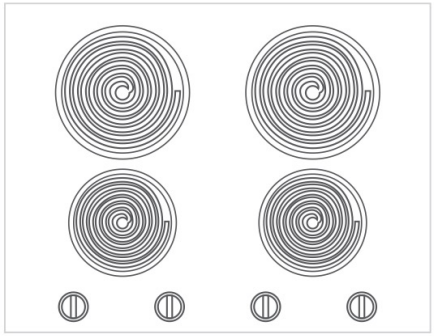
- Concerned with **relationship between controls and their movements or effects**
- Closely related to affordances & constraints
- How actions are translated (mapped) to effects

Mapping

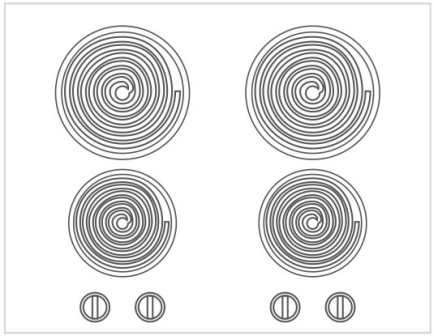
- User should easily perceive the relationship between action and reaction
- Good mappings
  - Feel natural
  - Are usually pictorial
  - Or can be kinesthetic (i.e., correspond with physical motion)
- Layout
  - stove controls and burners
- Behaviour
  - steering wheel and car turning
- Meaning
  - emergency button is red
- Locating
  - Where information is

Mapping: Layout

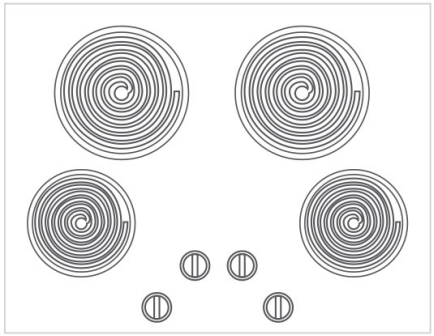
24 possibilities
requires visible labels / memory



4 possibilities
requires visible labels / memory

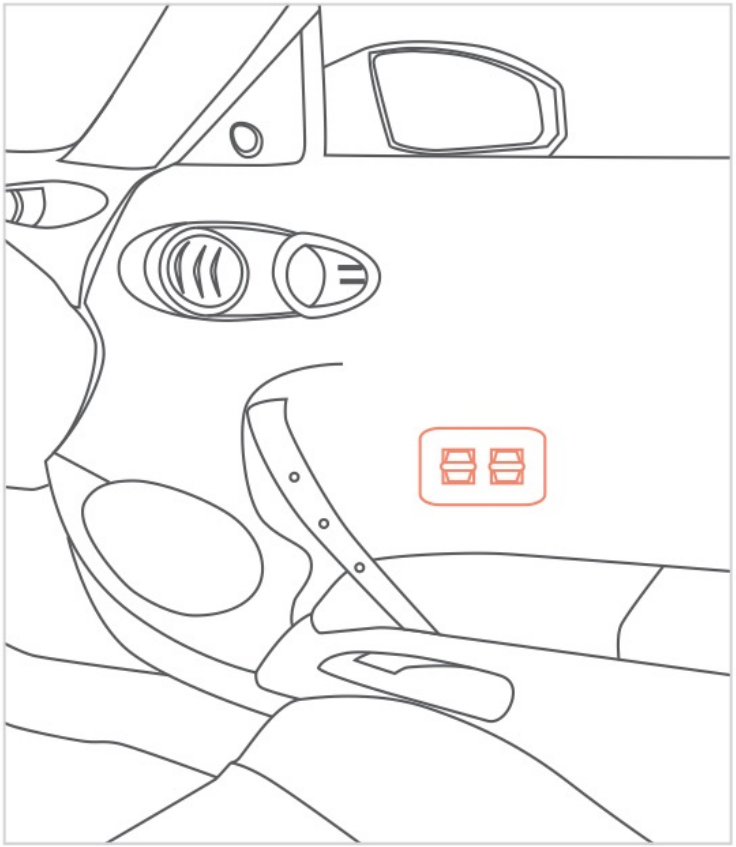


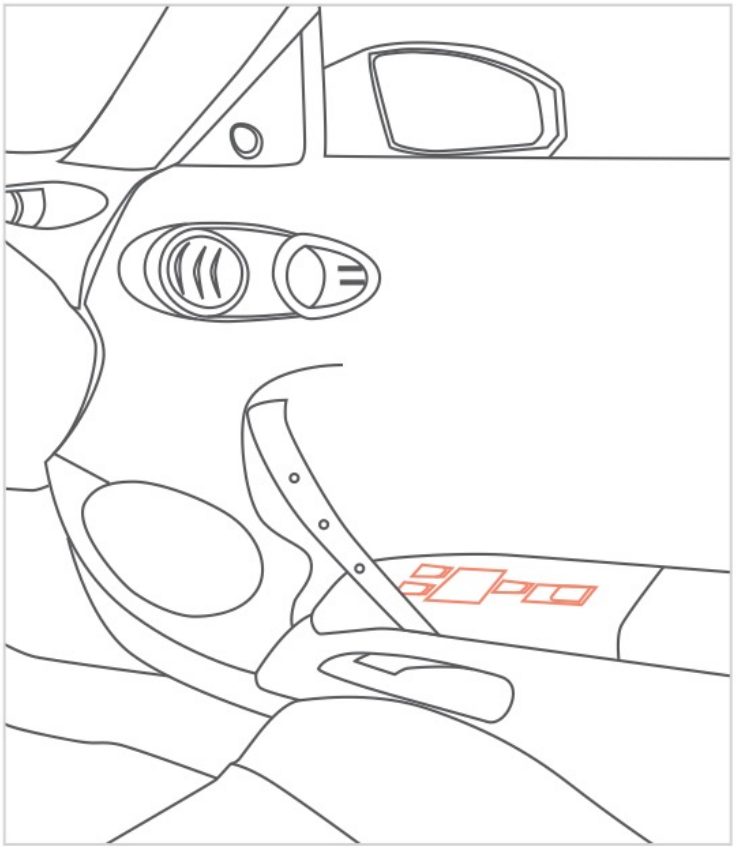
1 possibility
no labels needed



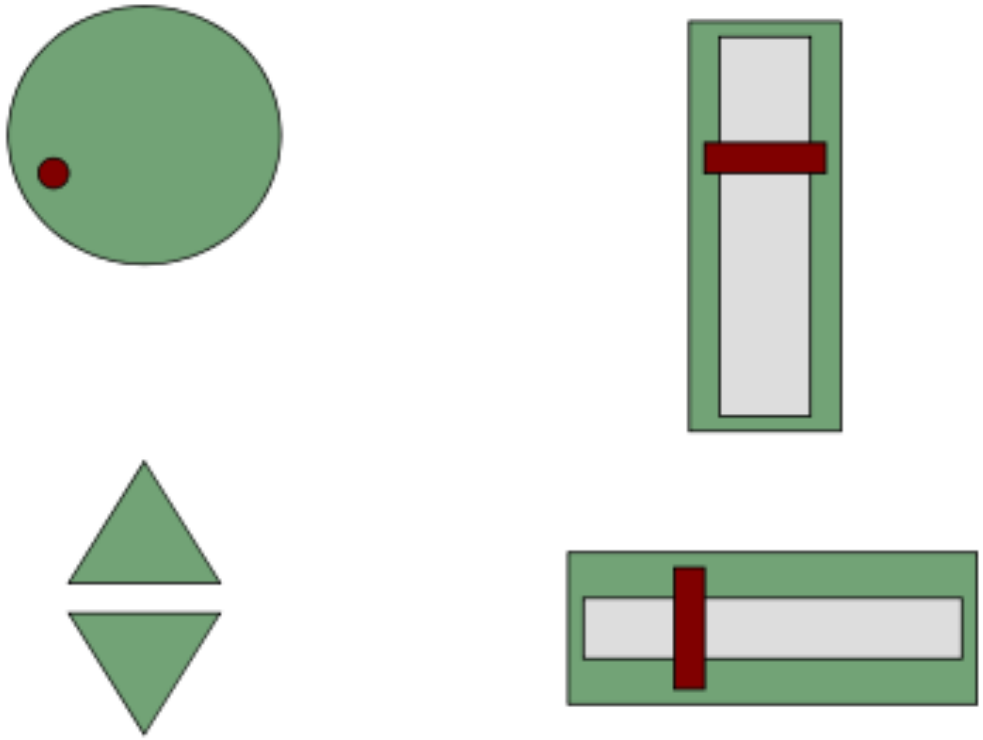
Mapping: Behavior

Which is better?





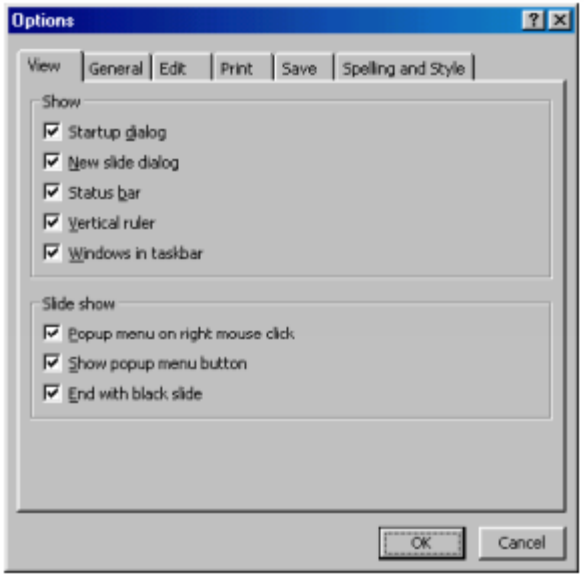
Mappings for increase/decrease

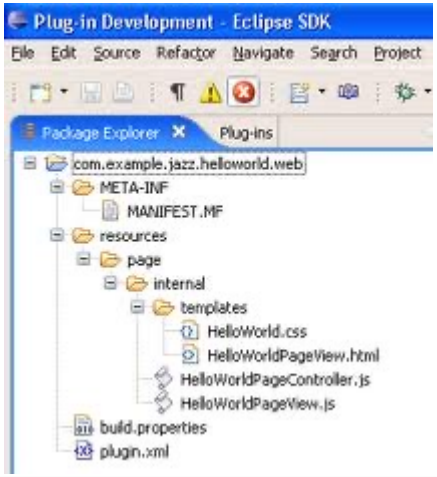


- Which is better?

Mapping and location

- Group information into separate categories
- Label them (must make sense)
- Allow user to select one at a time
- If not many categories, use tabs at the top
- If many, use hierarchies

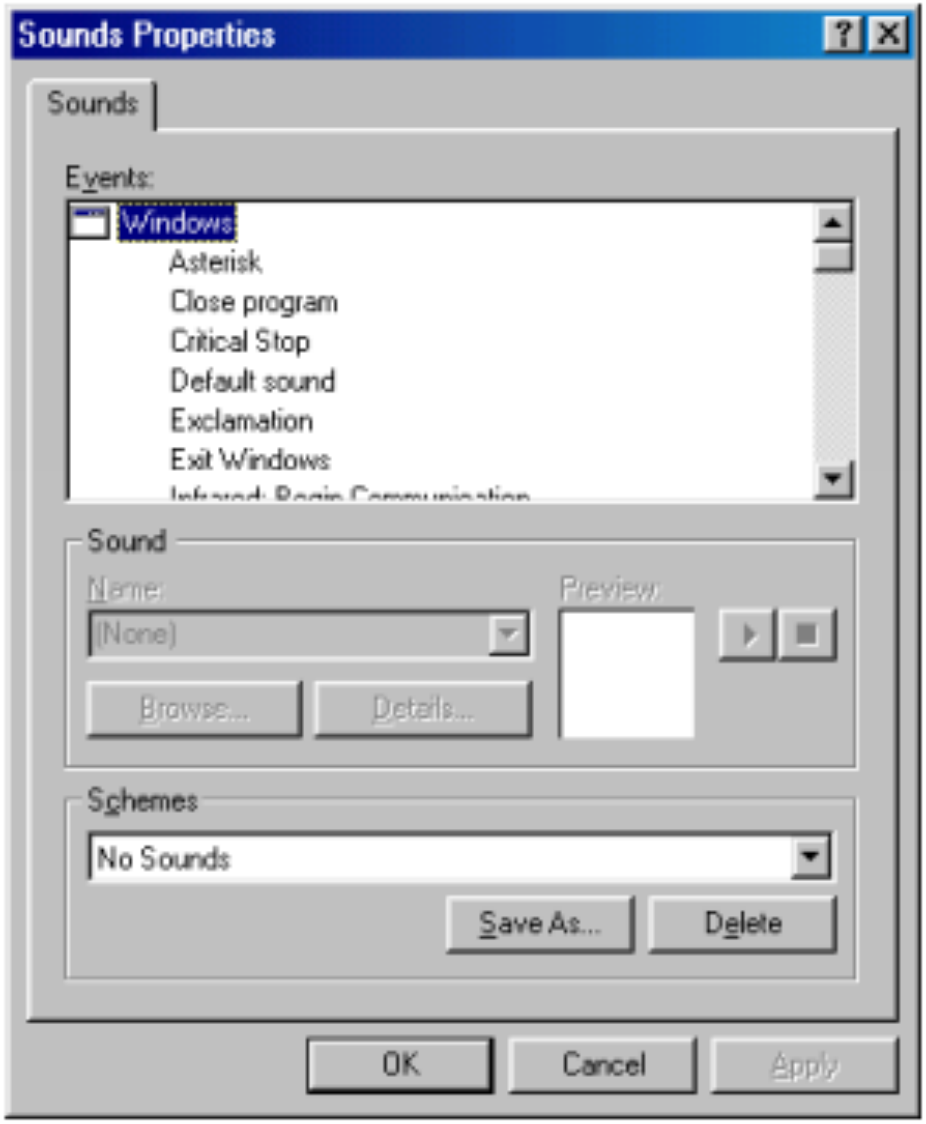




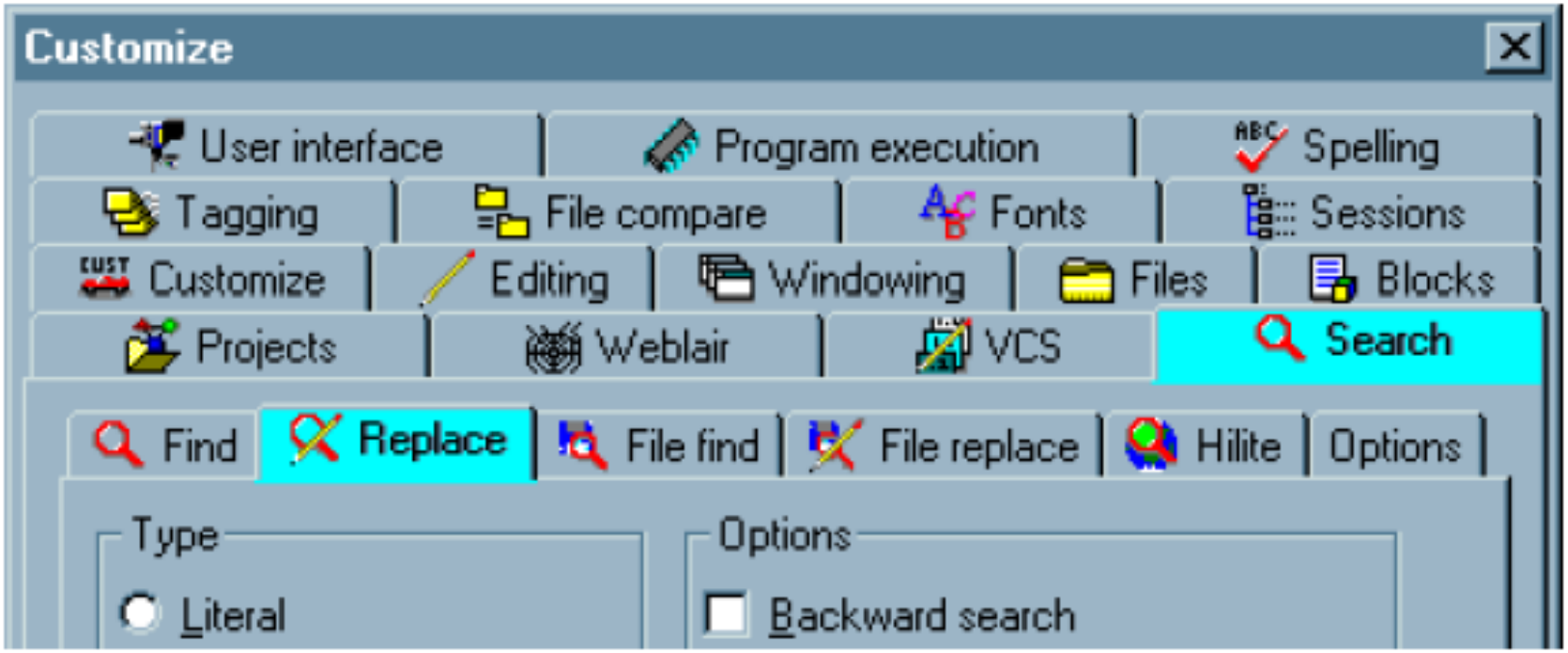
Mapping and tabs

- What is wrong here?

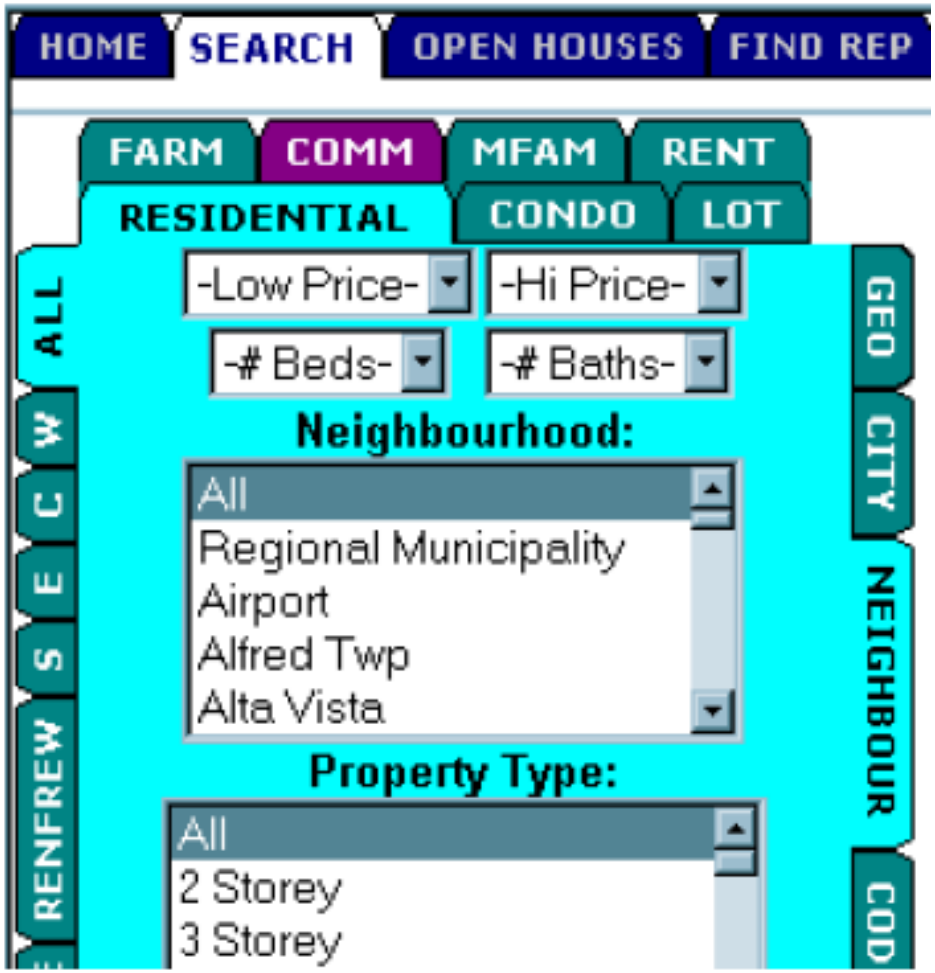
Why just have one tab?



Mapping can get out of hand…

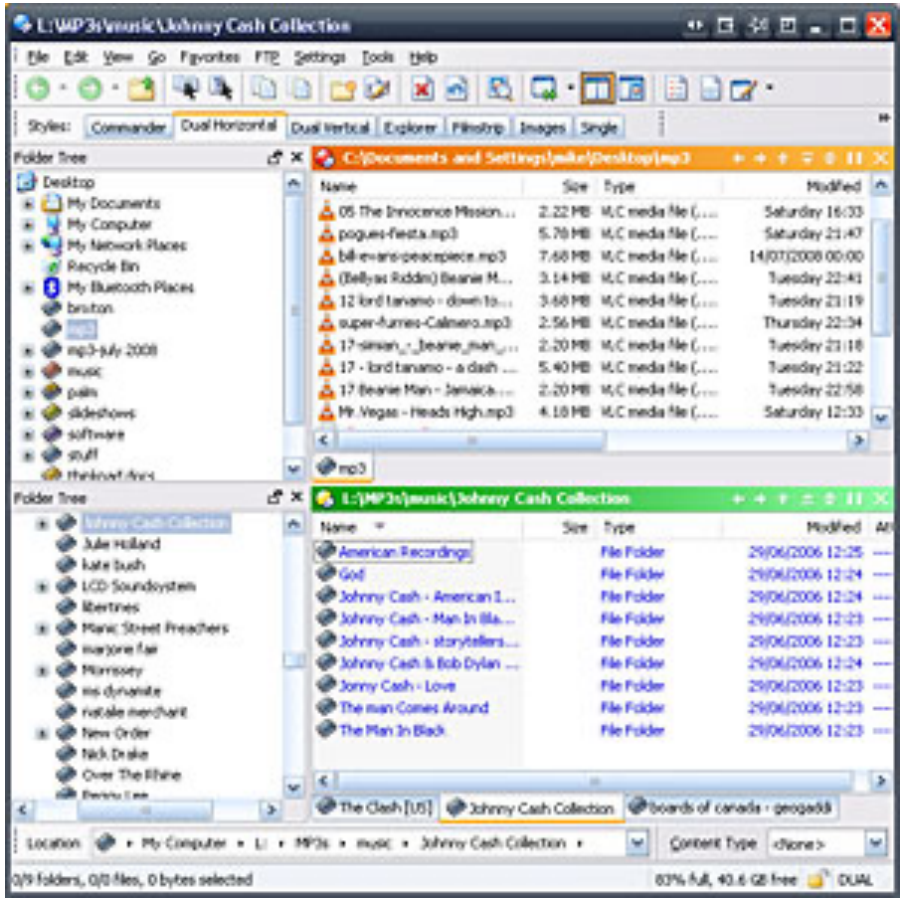


… *really* out of hand…



Mapping and large information spaces

- Split screens into panes
- Each pane shows one container
- Items in one container act as controls and map into another container
- Allows user to navigate in information space



Mapping principle

Whenever the number of possible actions is more than the number of interface controls, users will almost certainly run into problems

In general, try to avoid using a single control for multiple functions

Design principle: Mapping

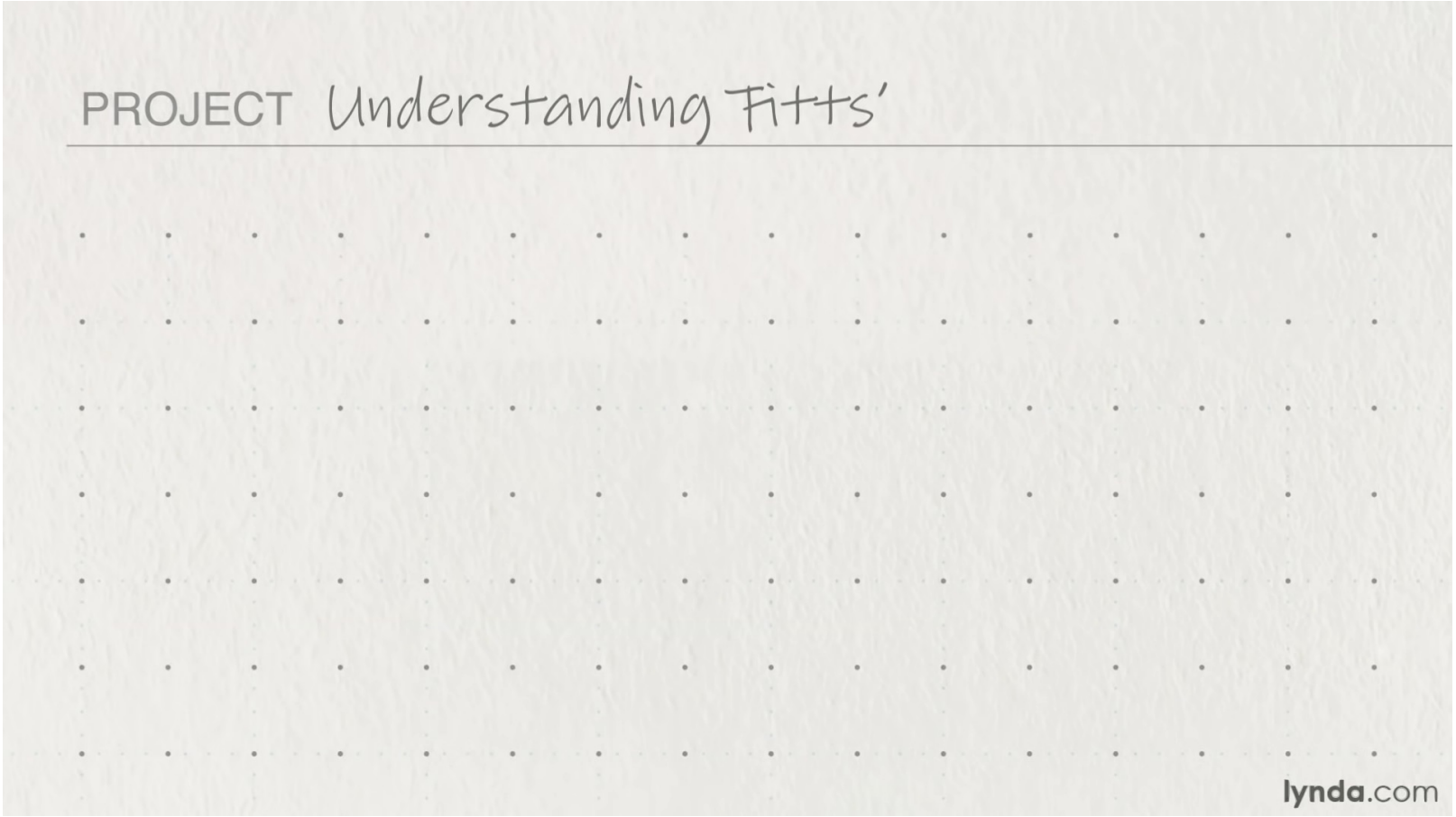
- Design controls so that their locations and behaviours correspond to the layout and behaviour of the device
- Simple control-effect relationships work best
- Avoid using a single control for multiple functions whenever possible; it is difficult to achieve good mappings for a one control-multiple effect relationship
- Be careful when relying on conventions to attach meaning to controls, as different population groups may interpret the conventions differently (e.g., in England, flipping a light switch up turns it off and flipping it down turns it on)

Fitts' Law

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

- The smaller and more distant a target, the longer it will take to move to a **resting position** over the target
- The faster the required movement and the smaller the target, the greater the error rate due to a speed-accuracy tradeoff
- Fitts' Law has implications for the design of controls, control layouts, and any device that facilitates movement to a target

Fitts' law



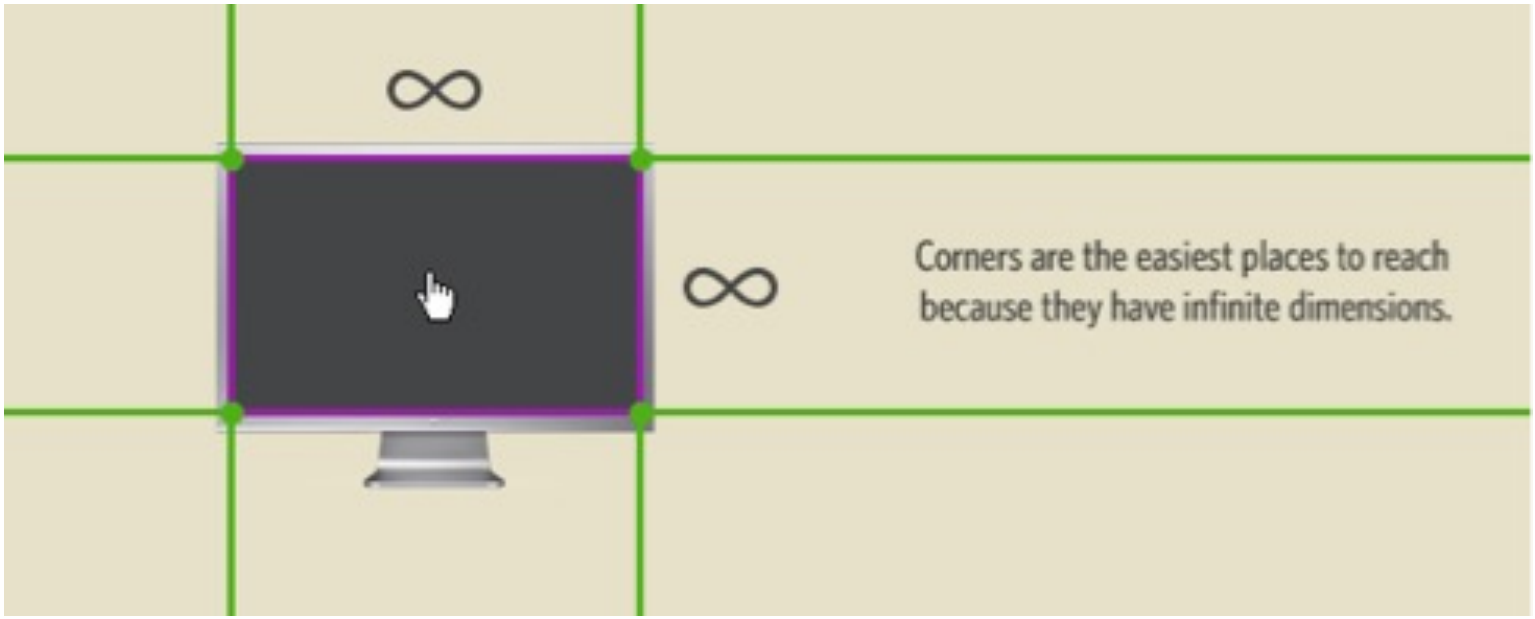
https://www.youtube.com/watch?v=95RoKSFyQ_k

Fitts' Law

- Applicable only for **rapid, pointing movements**, not for more continuous movements, such as writing or drawing
- Pointing movements typically consist of
  - One large, quick movement toward a target
  - Followed by fine-adjustment movements (*homing movements*)
  - To a resting position over the target
- Homing movements are generally responsible for most of the movement time and cause most errors

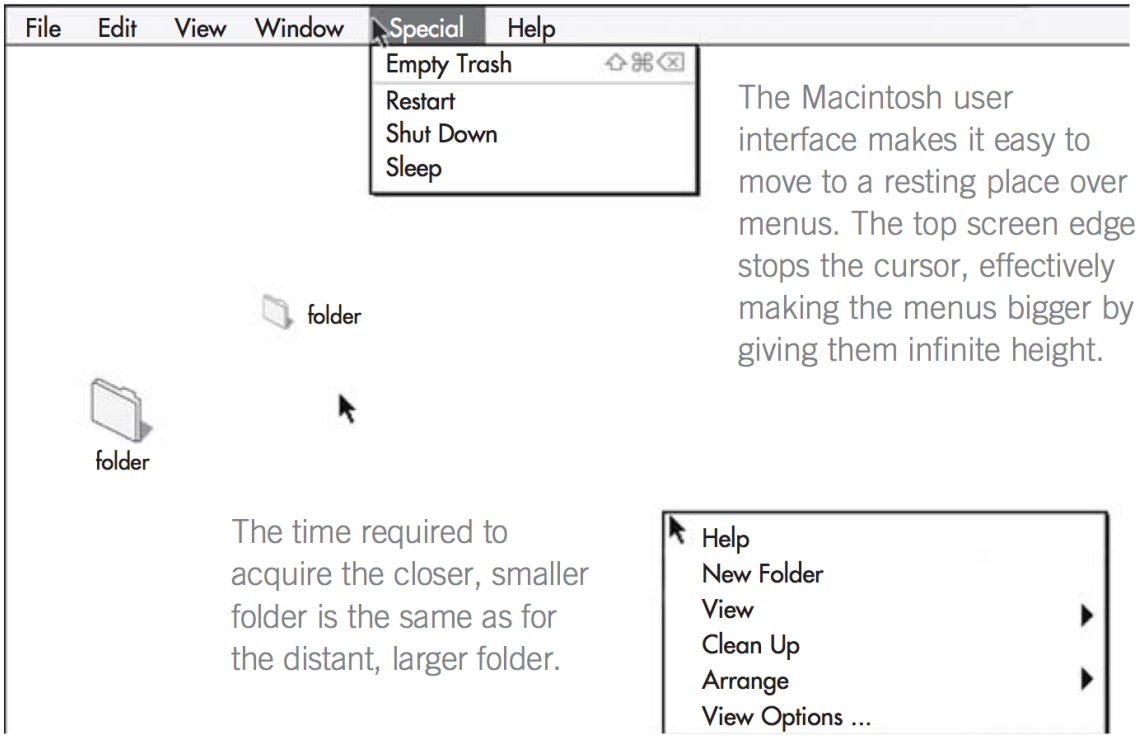
Fitts' Law

- Remember:… "homing movements are generally responsible for most of the movement time and cause most errors"
  - Rule of the infinite edge
  - Often ignored



Fitts' Law

"While at first glance, this law might seem patently obvious, it is one of the most ignored principles in design. Fitts' law dictates the Macintosh pull-down menu acquisition should be approximately five times faster than Windows menu acquisition, and this is proven out." – Bruce Tognazzini



The Macintosh user interface makes it easy to move to a resting place over menus. The top screen edge stops the cursor, effectively making the menus bigger by giving them infinite height.

The time required to acquire the closer, smaller folder is the same as for the distant, larger folder.

The Microsoft Windows user interface presents pop-up menus when you press the right mouse button. Since the distance between the cursor position and the pop-up menu is minimal, menu items are quickly acquired.

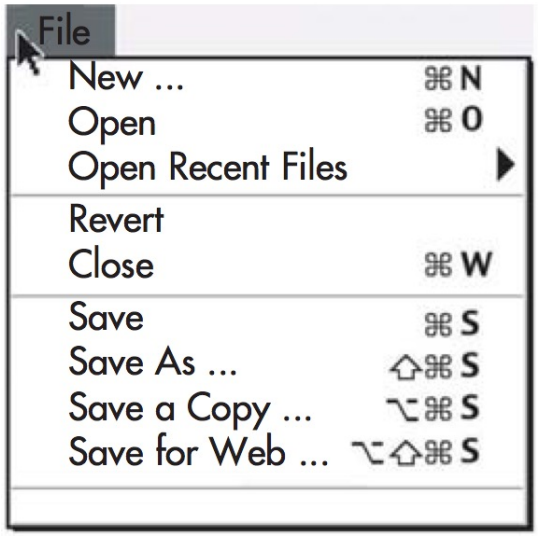
Hick's Law

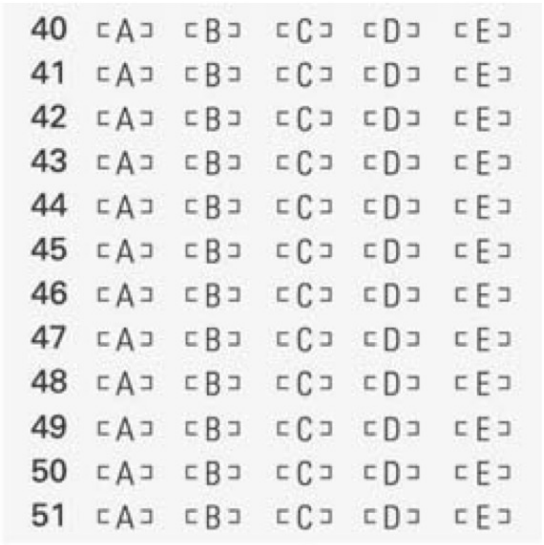
The time it takes to make a decision increases as the number of alternatives increases

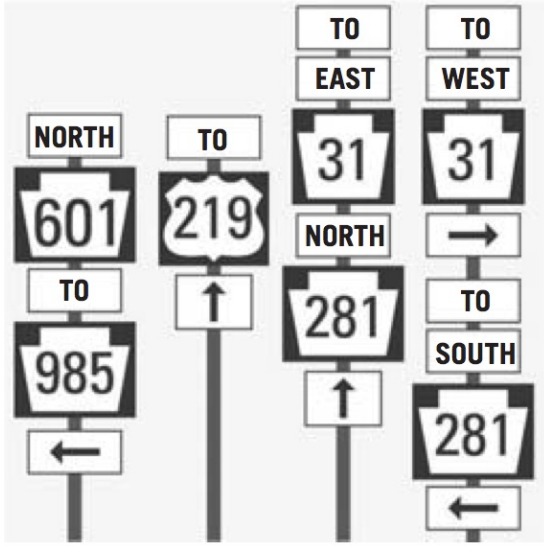
- Hick's Law has implications for the design of any system or process that requires **simple and quick decisions** to be made based on multiple options
  - e.g., control-room operator responds to alarm by pressing correct button
- **Not useful for complex tasks**
  - Decreasingly applicable as complexity increases

Hick's Law

- Does Hick's Law apply here?







Menu: yes, but not necessarily for large, ordered menus and sub-menus

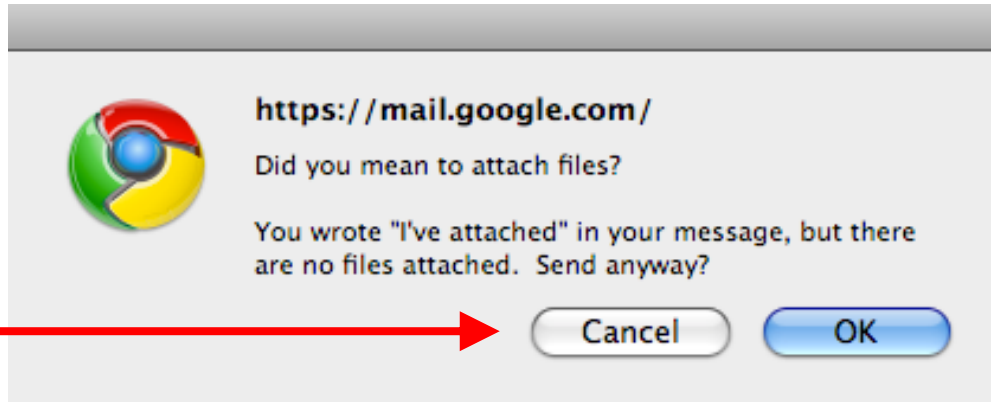
Ordering is important. Scanning time is linear—think about random sorting vs. alphabetical

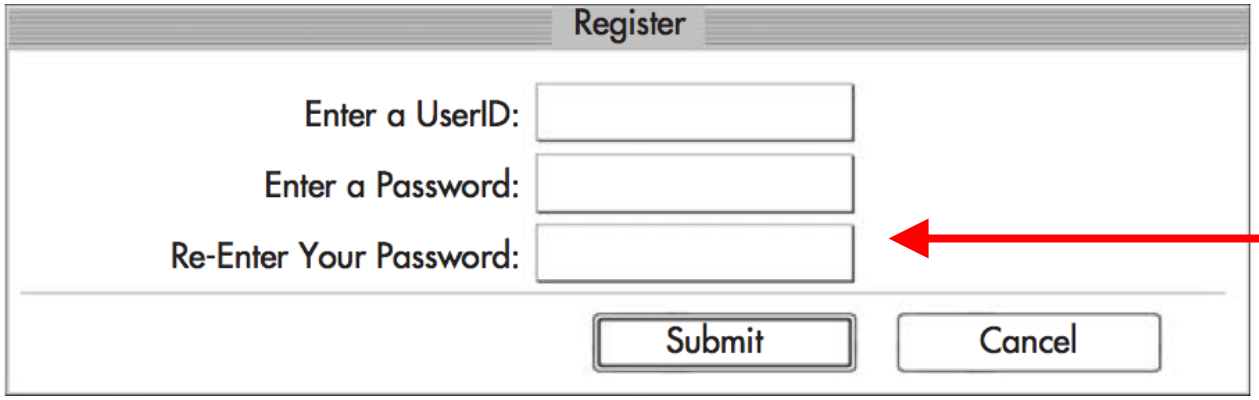
Multiple choice: no

Road signs: yes, assuming the driver's task is simple—e.g., "which way do I turn here?"

Confirmation

confirm intent





prevent slips

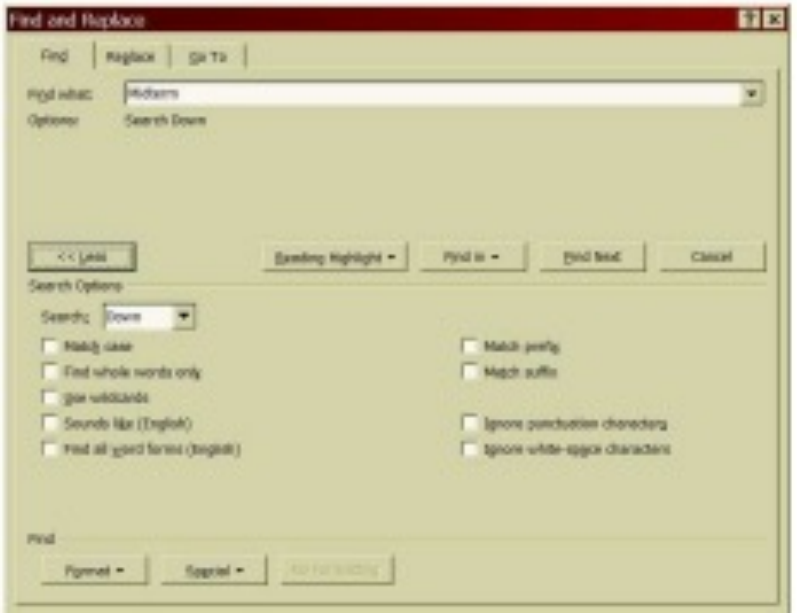
Confirmation

Use confirmations to minimize errors in the performance of critical or irrevocable operations. Avoid overusing confirmations to ensure that they are unexpected and uncommon; otherwise, they may be ignored.

- When would you use this principle?
- When the consequences of an action are not serious, or when actions are completely and easily reversible, confirmations are not needed

Progressive disclosure

- Use this strategy to manage information complexity where only necessary or requested information is displayed at any given time



Progressive disclosure

https://www.youtube.com/watch?v=kmDqFB1fCnk

Consistency

The usability of a system is improved when similar parts are expressed in similar ways

Two basic types:

1. Aesthetic
   - Style and appearance
   - Logo and colours are consistent across a website or brand

2. Functional
   - Action and behaviour
   - Clicking and dragging a window is consistent across an application

Consistency

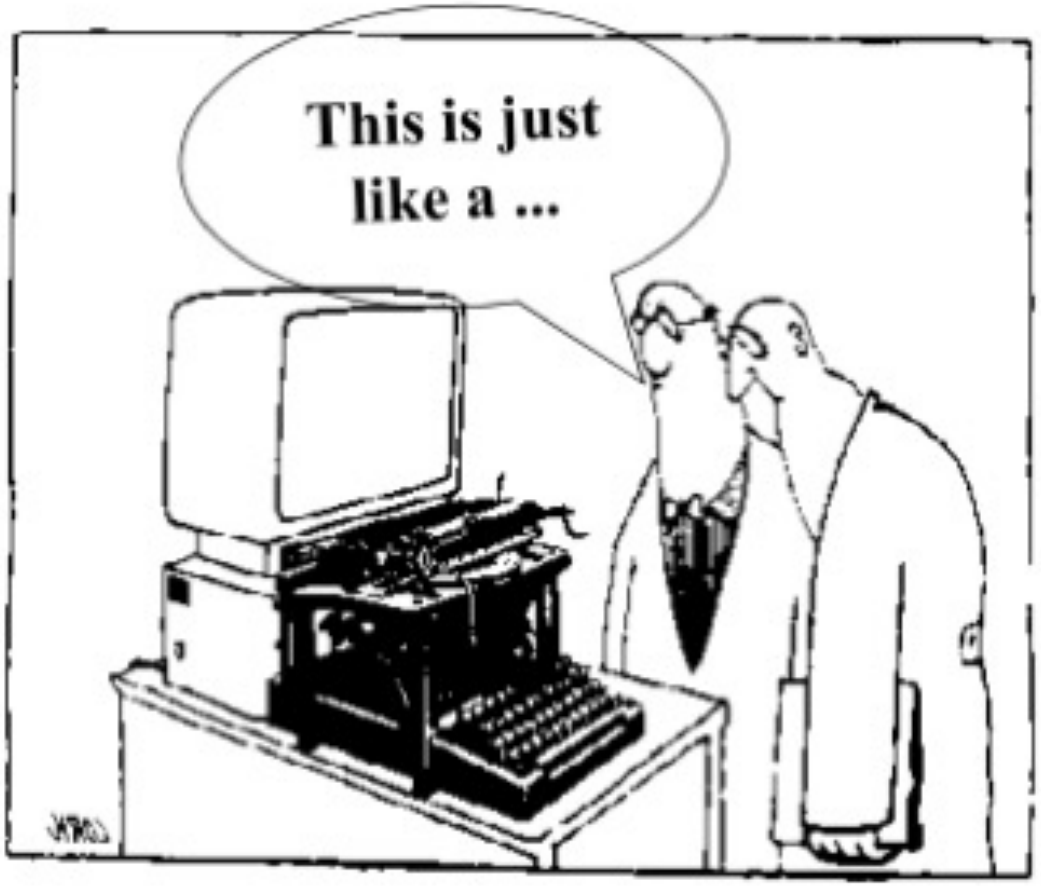
1. Internal
   - consistency with other elements within a system
   - where system can be place, application, tool, etc…
     - e.g., symbols within an application are consistent
     - e.g., traffic symbols within a country are consistent

2. External
   - consistency with other relevant elements in an environment outside the system of focus/concern
     - e.g., elements within an application are consistent with other applications (e.g., Apple iOS)
     - e.g., elements within a control-room software tool are consistent with other elements in the room

Consistency

- Watch this yourself
  - https://www.youtube.com/watch?v=DwQwHpgXRPk

What is happening here?



This is called **transfer**

The process and the effective extent to which past experiences (transfer source) affect learning and performance in a current novel situation (transfer target)

Transfer levels

Transfer in HCI takes place at two levels

1. High, conceptual level (macro level)
   - e.g., how an application works
   - "this word processor is just like the other one"

2. Low, task & implementation level (micro level)
   - e.g., how to carry out a task (e.g., cut-paste)
   - "oh, it's ctrl-x and ctrl-v … like the other one"

Transfer Types

- 3 types of transfer:

1. negative

2. zero

3. positive

Transfer: Negative

- What is learned in one context hinders or delays learning in a different setting
  - i.e., previous knowledge conflicts with new situation
- Can be resolved after some period of making mistakes
  - Can be troublesome, wasteful
  - Can make user feel stupid

Transfer: Zero

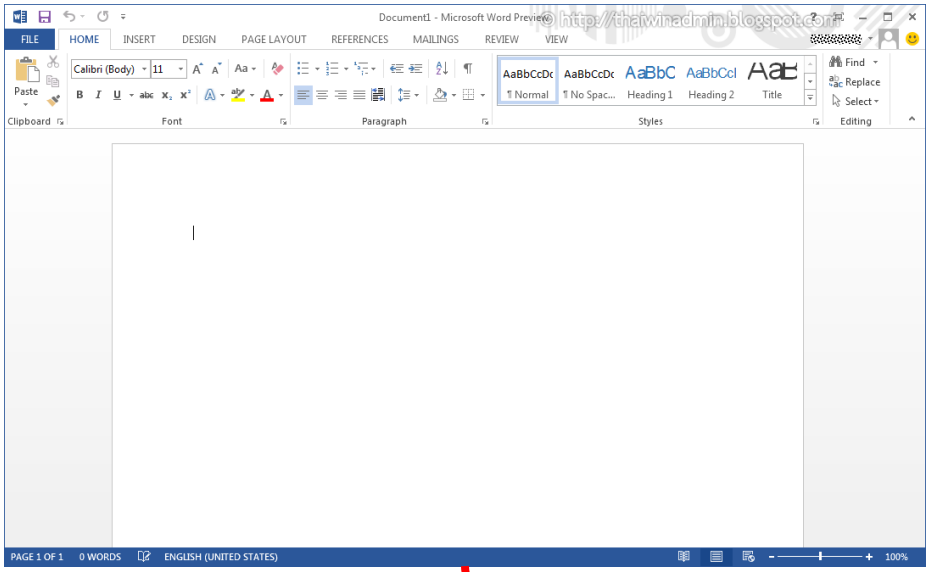
- User does not see any similarity between the current situation or object and anything known from before
  - e.g., knowing SQL does not readily transfer to MS word
- May be desirable when you design something totally novel

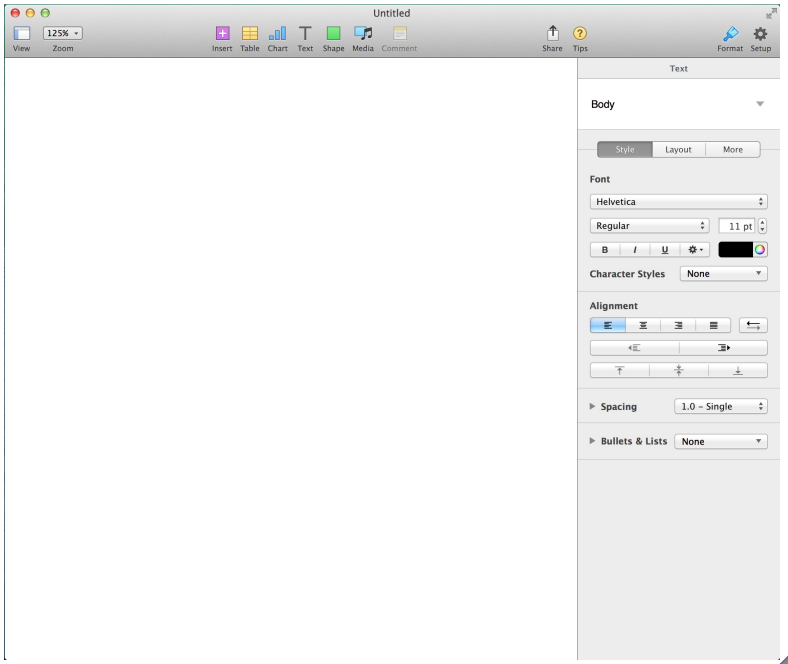
Transfer: Positive

- Previous knowledge transfers to and supports the new situation
  - e.g., how menus are used across applications
- May be desirable if you want users to know how to use an application quickly: transfer of usability across applications and platforms

Transfer Example

- A Windows user finds a Mac system only somewhat familiar
  - Positive transfer at the high, conceptual level
  - Negative transfer at the low, task level





Design principle: Transfer

It is important to think about how your users will transfer their understanding of previous artifacts and situations to what you are designing.

Positive transfer is best

Zero transfer is desirable when you are designing completely novel artifacts

Negative transfer is not desirable

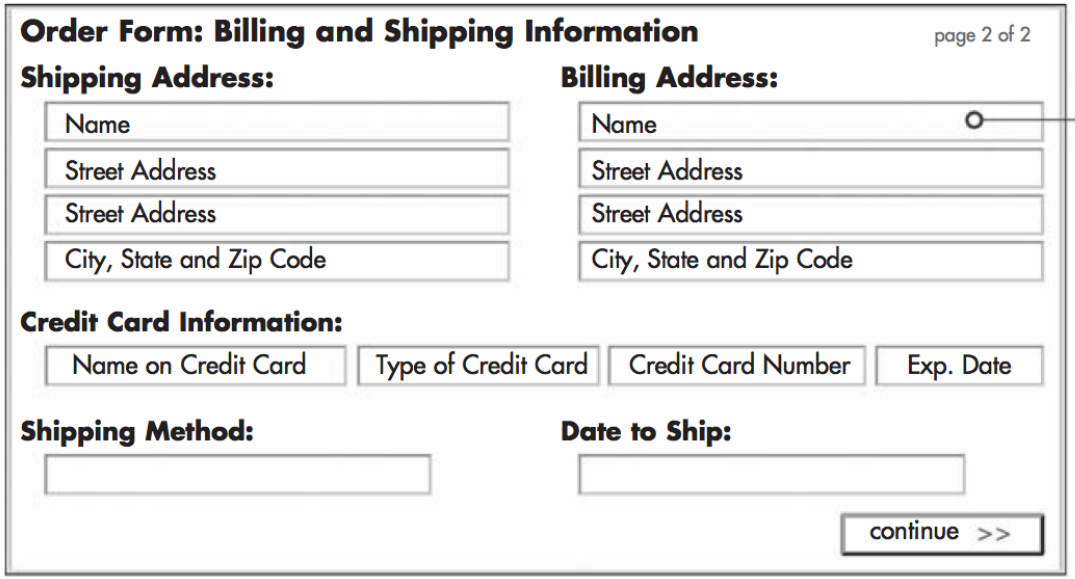
Garbage in – Garbage out

The quality of system output is dependent on the quality of system input

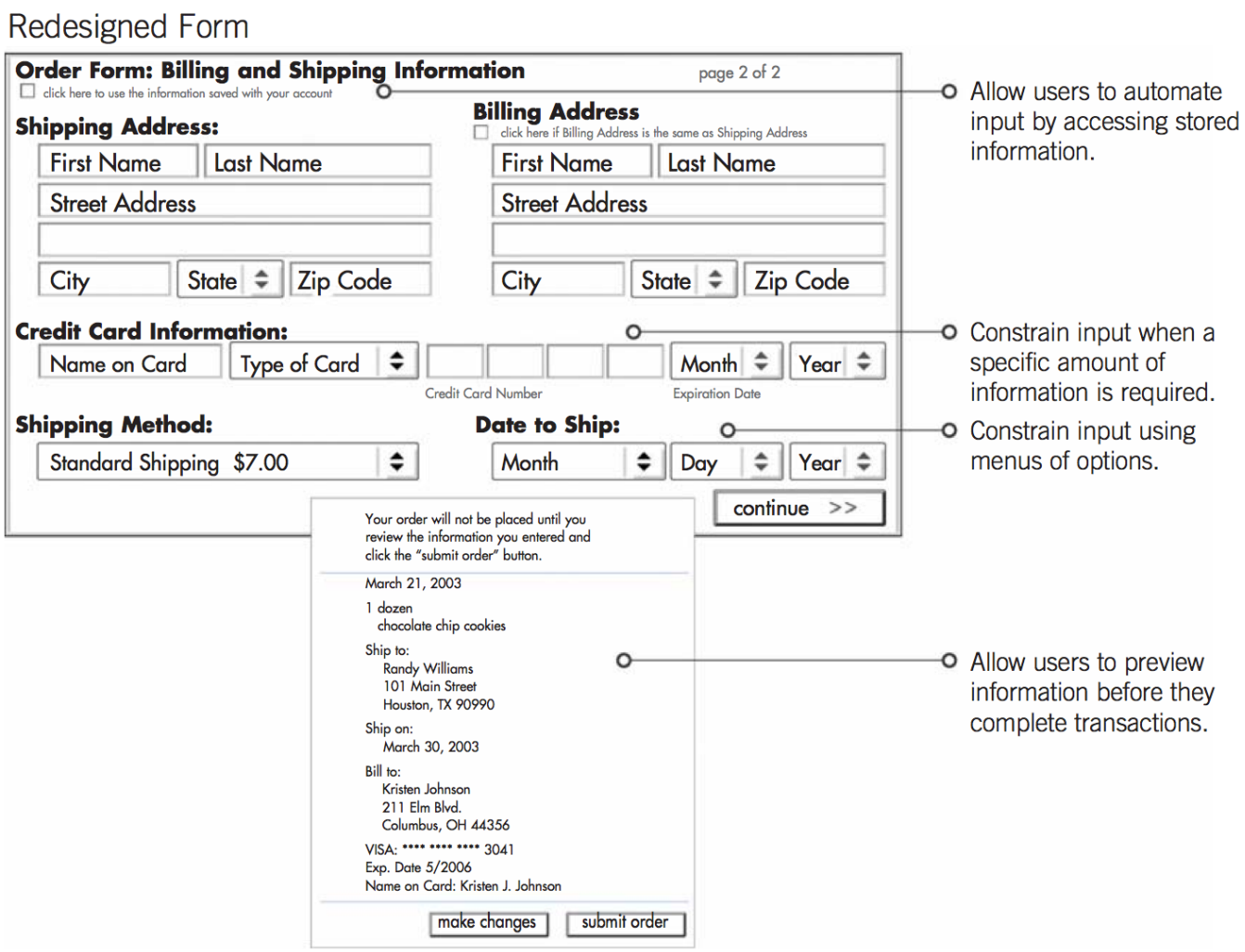
- Rule has been generalized over time to apply to all systems, and is commonly invoked in domains such as business, education, nutrition, and engineering
- Proper use of other principles (affordances, constraints, feedforward, confirmation, …) can minimize garbage input

Garbage in – Garbage out

Original Form

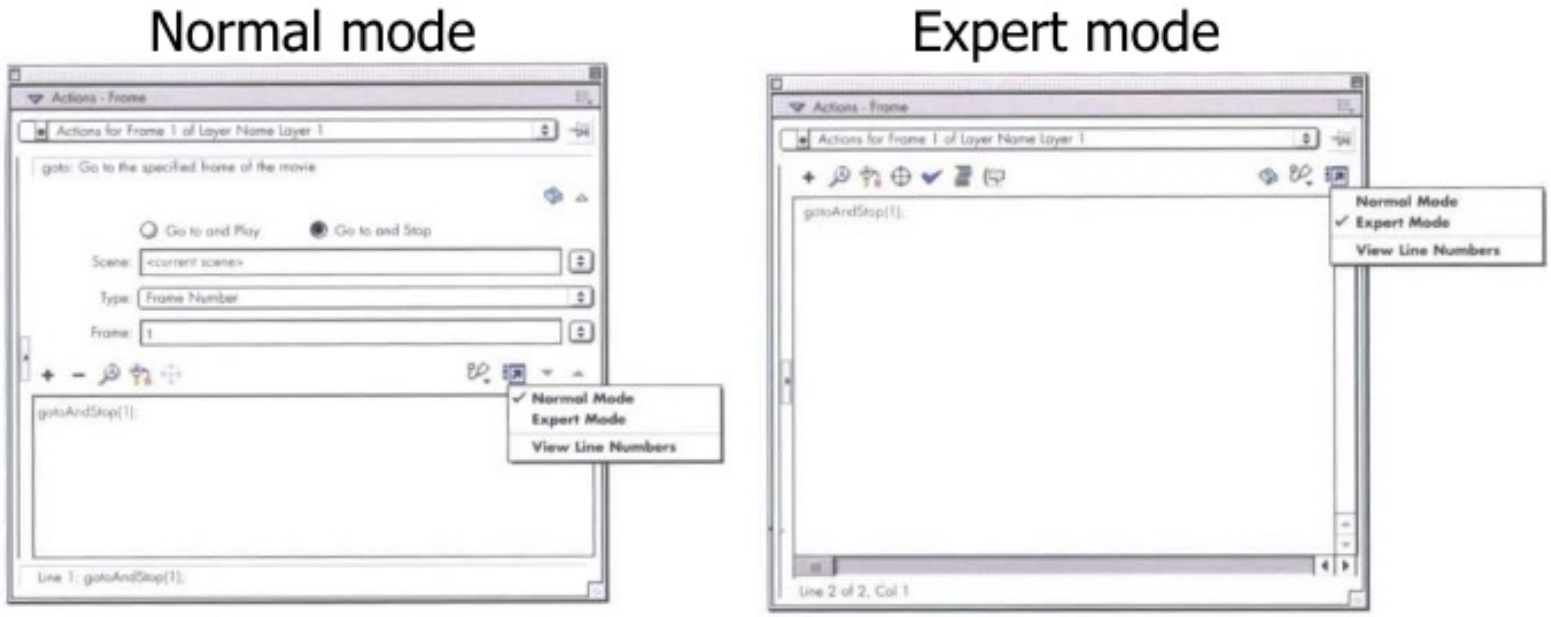


Garbage in – Garbage out



Control

- For complex systems, consider designs that allow users control over the levels of expertise and individual preferences



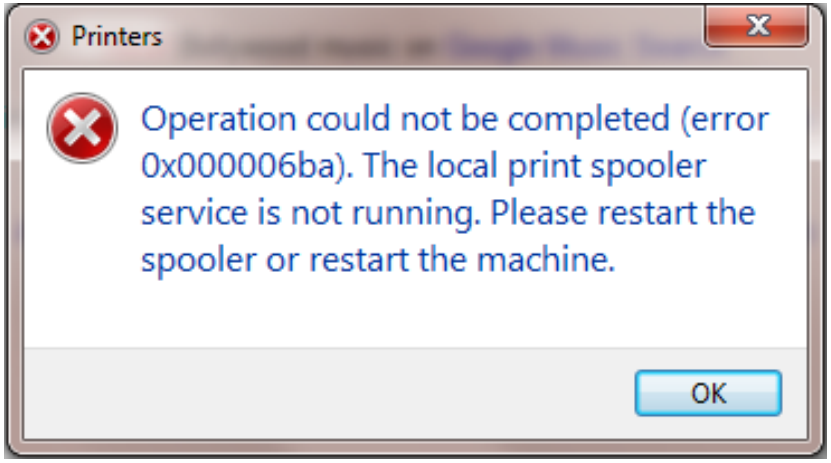
Feedback

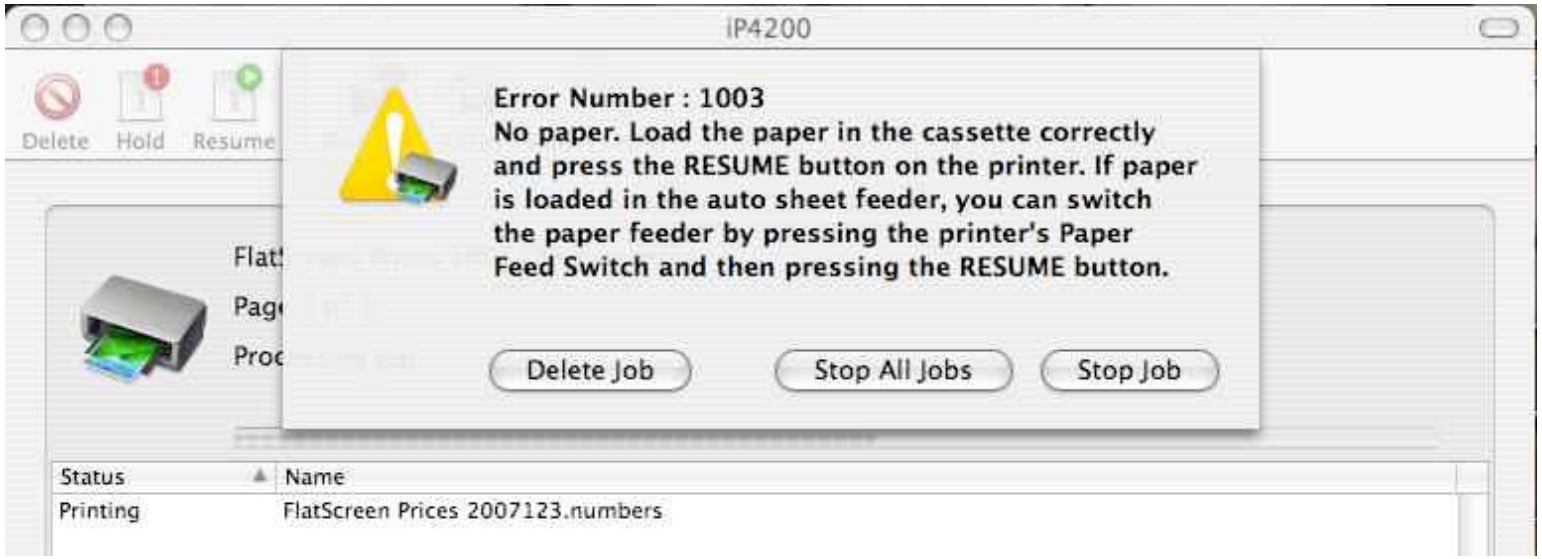
- Providing user with (immediate) information about
  - What action has been done
  - Whether action has desired or undesired result
  - What result has been accomplished
  - What action the user has to perform
  - …
- Imagine:
  - Talking to someone and not hearing your own voice, or the other person not responding
  - Writing with a pencil, but not seeing any marks
  - Typing a command and not knowing what the result is
    - e.g., many command-language interfaces
- These are examples of no feedback

Bad feedback

Bad feedback

Bad vs. better feedback





Types of feedback

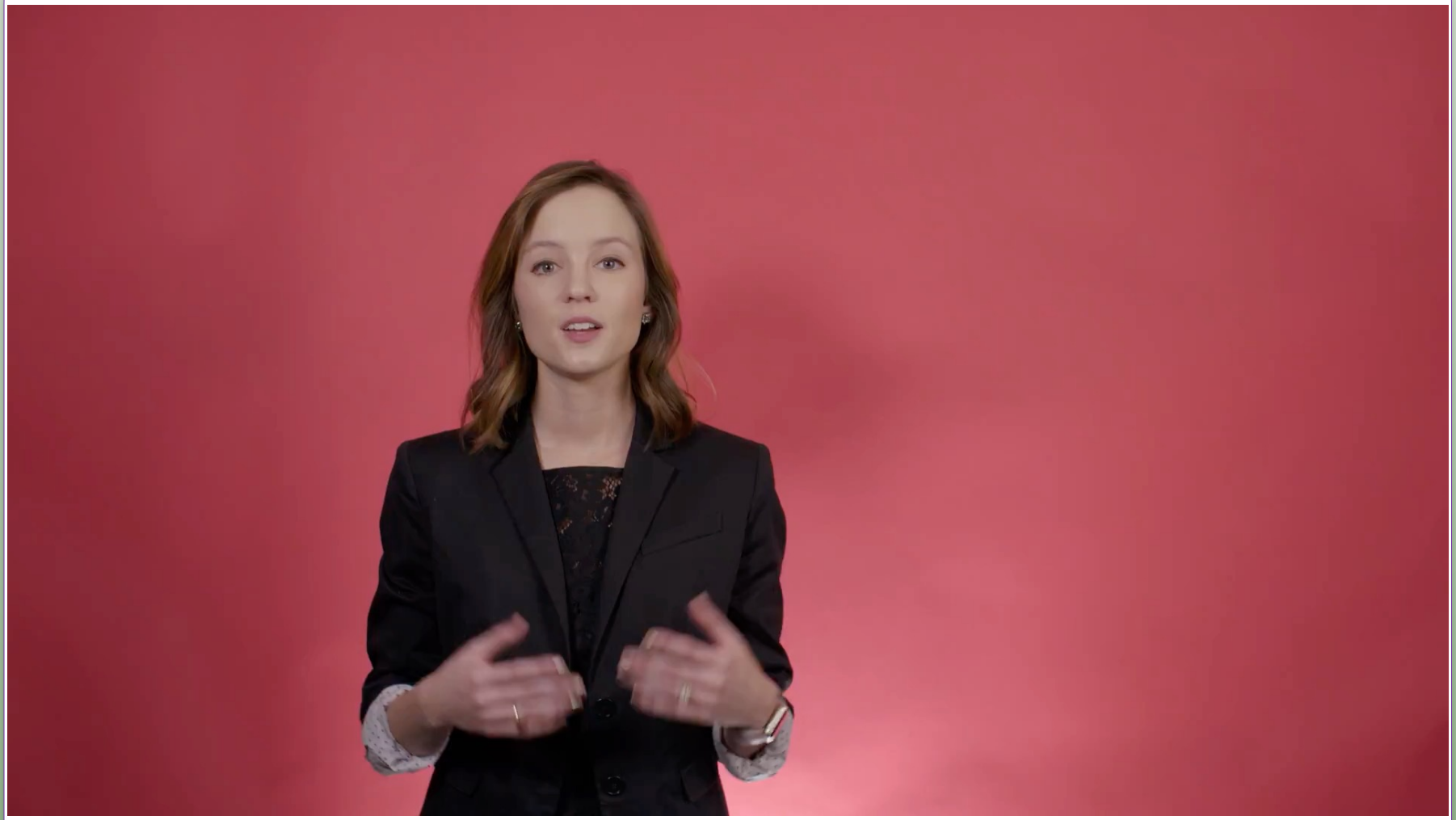
result of action… was it accomplished?

time/progress known

time/progress unknown

help bridge gulf of evaluation

Search suggestions



https://www.youtube.com/watch?v=7XpQVWaTO2s

Causality

The thing that happens right after an action is interpreted by people to be caused by that action

- False causality
  - Simultaneous effect
    - Starting up an unfamiliar application just as computer crashes causes "superstitious" behaviors
  - Invisible effect
    - Command with no apparent result often re-entered repeatedly
- When possible, try to predict and account for false causality in your designs

Conceptual Models Revisited

- People form mental models of:
  - How objects work
  - Why events take place
  - What action causes what effect
  - ….
- Models allow people to *mentally simulate* the operation of a device
- Designers should provide appropriate conceptual models

Conceptual models revisited

- Conceptual models are made from your design language which embodies the following:
  - Gestalt rules
  - Affordances
  - Constraints
  - Mapping
  - Transfer
  - Consistency
  - Control
  - Feedback
  - Visibility
  - Representations
  - Metaphors
  - …

How to think about design principles

- Have to know them thoroughly
  - Good designers have a repertoire in their mind to draw from
- Not enough to just know them in isolation! Need to **carefully think** about how they combine and are related, just like "language" and combining concepts
  - e.g., what principles apply to this HCI artifact, and how should I combine them to meet the needs of the task, context, users, …?
  - e.g., how should I combine transfer and consistency?
  - e.g., what is the interplay between visibility and 80/20?
  - e.g., metaphors, affordances, and constraints
  - e.g., mapping and progressive disclosure
  - e.g., …