

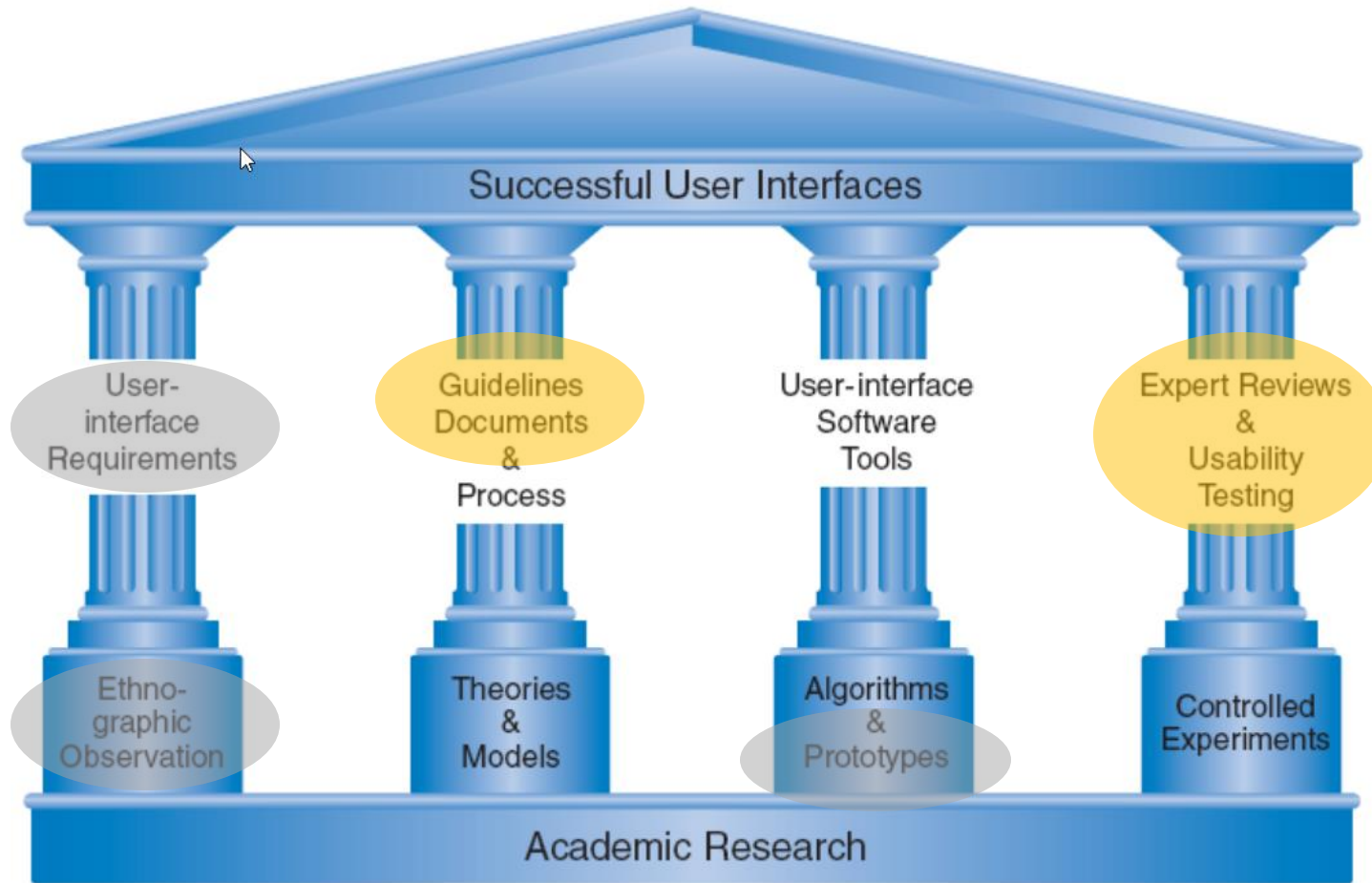
Evaluation: Introduction and Heuristics

Human Computer Interaction

Luigi De Russis, Alberto Monge Roffarello

Academic Year 2024/2025

The Four Pillars of Design



Ben Shneiderman & Catherine Plaisant, Designing the User Interface: Strategies for Effective Human-Computer Interaction

Goals

Generating design solutions

- Guidelines
- Principles
- Theories

Evaluating generated designs

- Expert reviews and heuristics
- Usability testing
- Controlled experiments

Evaluation

Testing the usability, functionality and acceptability of an interactive system

Goal

- Evaluation: «Evaluation tests the usability, functionality and acceptability of an interactive system»
 - According to the design stage (sketch, prototype, final)
 - According to the initial goals
 - Alongside the different usability dimensions
 - Using a range of different techniques
- Identify and correct issues as soon as possible

Usability

- **Usability:** how well users can use the system's functionality.
- Dimensions of usability:
 - **Usefulness:** does it do something people want?
 - **Learnability:** is it easy to learn?
 - **Memorability:** once learned, is it easy to remember?
 - **Effectiveness:** does it allow reaching the goal?
 - **Efficiency:** once learned, is it fast to use?
 - **Visibility:** is the state of the system visible?
 - **Errors:** are errors few and recoverable?
 - **Satisfaction:** is it enjoyable to use?

Functionality

- **Functionality:** the system's functionality must accord with the user's requirements and should enable users to perform their intended tasks.
- Functionality can be tested in different ways:
 - Are the appropriate functionality available within the system?
 - Are they clearly reachable by the user?
 - Do they match the the user's expectations?
- Functionality evaluation may also include measuring the user's performance with the system, to assess the effectiveness of the system in supporting the task.

Acceptability

- Technology **acceptability**: one's perception of a system before use.
Technology **acceptance**: one's perception of the system after use.
- Good User Interface design can make a product easy to understand and use, which results in greater user acceptance.
- Testing **acceptability** means evaluating the enjoyment and emotional response to a system
 - particularly for systems aimed at leisure or entertainment
- This may involve:
 - measuring satisfaction and comfort
 - identifying areas of the design that overload the user

Many Evaluation Approaches

- Evaluation may take place:
 - In the laboratory
 - In the field

Many Evaluation Approaches

- In lab studies, users are taken out of their normal work environment to take part in **controlled** tests. They are typically adopted in early stages of design (e.g., to compare alternatives, you do not need a working implementation).
 - 👍 simulation of dangerous environments
 - 👍 suitable for specific tasks within a system
 - 👎 lack of context
 - 👎 unnatural situations leading to biases
 - 👎 not suitable for all the tasks

Many Evaluation Approaches

- Field studies takes the designer or evaluator out into the **user's work environment** to observe the system in action.
 - 👍 open nature: the “real” context
 - 👍 users are in their natural environment
 - 👎 low degree of control
 - 👎 higher costs (you need a working implementation)
 - 👎 longer duration

Many Evaluation Approaches

- Evaluation may be based on **expert evaluation**:
 - Analytic methods
 - Review methods
 - Model-based methods
 - Heuristics
- It is useful to identify any areas that are likely to cause difficulties because they violate known cognitive principles, or ignore accepted empirical results
 - 👍 it can be used at any stage in the development process
 - 👍 it is relatively cheap, since it does not require user involvement
 - 👎 it does not assess actual use of the system

Many Evaluation Approaches

- Evaluation may involve users:
 - Experimental methods
 - Observational methods
 - Query methods
 - Formal or semi-formal or informal
- In experimental and observational methods, the evaluator chooses a hypothesis to test, which can be determined by measuring some attribute of participant behavior.
 - 👍 they provide empirical evidence
 - 👎 they require more time to be designed and analyzed
- Query techniques (e.g., interviews) relies on asking the user about the interface directly
 - 👍 they are simple and cheap
 - 👎 you get subjective results

Many Evaluation Approaches

- We can also adopt automated evaluation:
 - Simulation and software measures
 - Formal evaluation with models and formulas
 - Especially for low-level issues

Heuristic Evaluation

Experts check potential issues on your design, by referring to a set of heuristic criteria

When Is Design Critique Useful?

- Before user testing
 - To save effort
 - Solving easy-to-solve problems
 - Leaving user testing for bigger issues
- Before redesigning
 - Identify the good parts (to be kept) and the bad ones (to be redesigned)
- To generate evidence for problems that are known (or suspected)
 - From ‘murmurs’ or ‘impressions’ to hard evidence
- Before release
 - Smoothing and polishing

Heuristic Evaluation

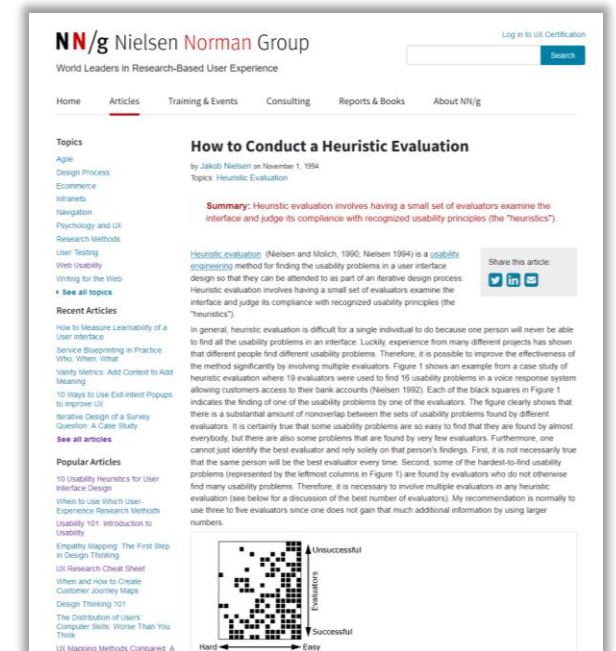


- A method developed by Jakob Nielsen (1994)
 - Structured design critique
 - Using a set of simple and general heuristics
 - Executed by a small group of experts (3-5)
 - Suitable for any stage of the design (sketches, UI, ...)
 - Goal: find usability problems in a design
- Also popularized as “Discount Usability”
- A heuristic is a guideline or general principle or rule of thumb that can guide a design decision or be used to critique a decision that has already been made.



Basic Idea

- Define a set of heuristics (or principles)
- Give those heuristics to a group of experts
 - Each expert will use heuristics to look for problems in the design
- Experts work independently
 - Each expert will find different problems
- At the end, experts communicate and share their findings
 - Findings are analyzed, aggregated, ranked
- The discovered *violations* of the heuristics are used to fix problems or to re-design



Heuristics

- Nielsen proposed 10 heuristic rules
 - Good at finding most design problems
 - Inspired and connected to the Design Principles (→Guidelines)
- In a specific context, application domain, or for specific design goals ...
 - ... new heuristics can be defined
 - ... some heuristic can be ignored

Phases of Heuristic Evaluation

1. Pre-evaluation training
 - Give evaluator information about the domain and the scenario to be evaluated
2. Evaluation
 - Individual
3. Severity Rating
 - First, individually
 - Then, aggregate and find consensus
4. Debriefing
 - Review with the design team

Evaluation (I)

- Define a set of tasks, that the evaluators should analyze
- For each task, the evaluator should step through the design several times, and inspect the UI elements
 - On the real design, or on a preliminary prototype
- At each step, check the design according to each of the heuristics
 - 1st step, get a general feeling for the interaction flow and general scope
 - 2nd step (and following), focus on specific UI elements, knowing where they fit in the general picture
- Heuristics are used as a “reminder” of things to look for
 - Other types of problems can also be reported

Evaluation (II)

- Comments from each evaluator should be recorded or written
 - There may be an observer, taking notes
 - The observer may provide clarifications, especially if the evaluator is not a domain expert
- Session duration is normally 1h – 2h
- Each evaluator should provide a list of usability problems
 - Which heuristic (or other usability rule) has been violated, and why
 - Not a subjective comment, but a reference to a known principle
 - Each problem reported separately, in detail

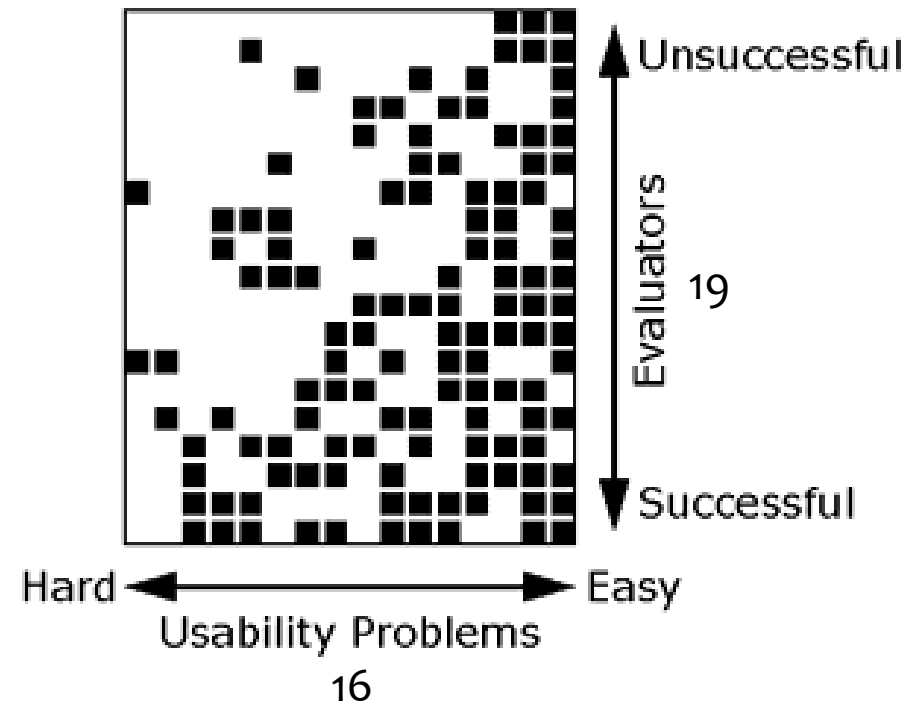


Evaluation (III)

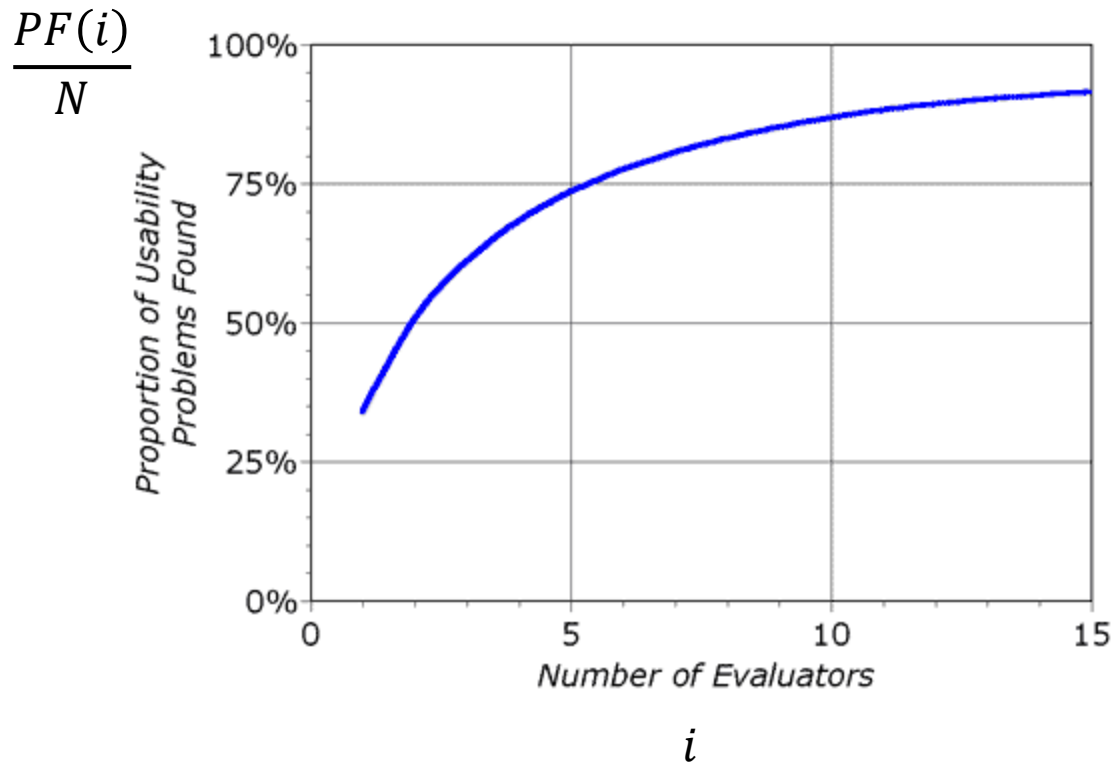
- Where problems may be found
 - A single location in the UI
 - Two or more locations that need to be compared
 - Problem with the overall UI structure
 - Something is missing
 - May be due to prototype approximation
 - May still be unimplemented

Multiple Evaluators

- No evaluator finds all problems
 - Even the best one finds only $\sim 1/3$
- Different evaluators find different problems
 - Substantial amount of nonoverlap
- Some evaluators find more problems than others



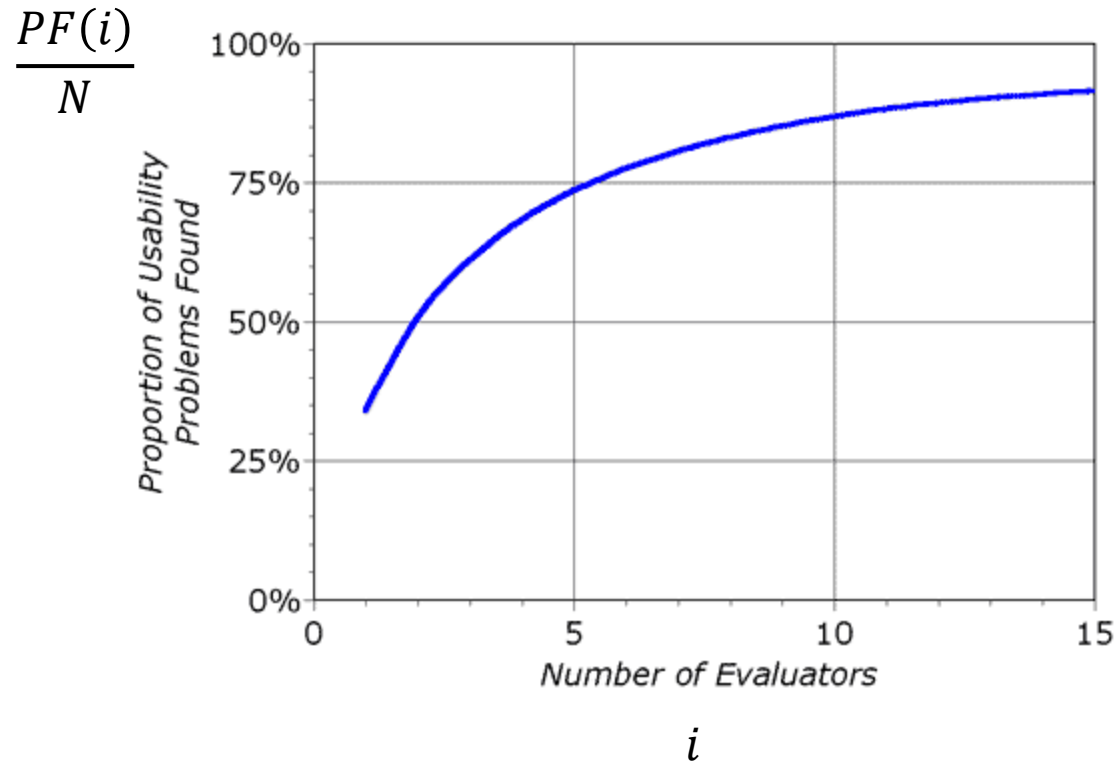
How Many Evaluators?



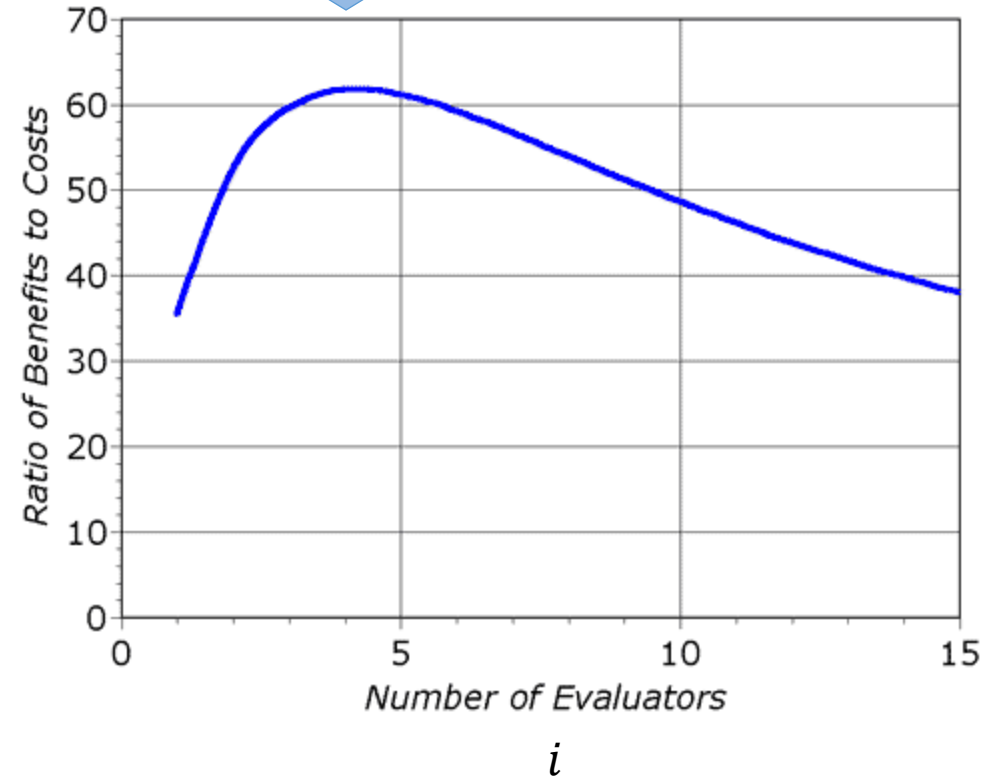
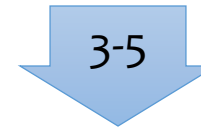
- $PF(i) = N(1 - (1 - l)^i)$
- $PF(i)$: problems found
- i : number of *independent* evaluators
- N : number of existing (but unknown) usability problems
- l : ratio of usability problems found by a single evaluator

How Many Evaluators?

$$Cost(i) = Fixed + Fee \times i$$



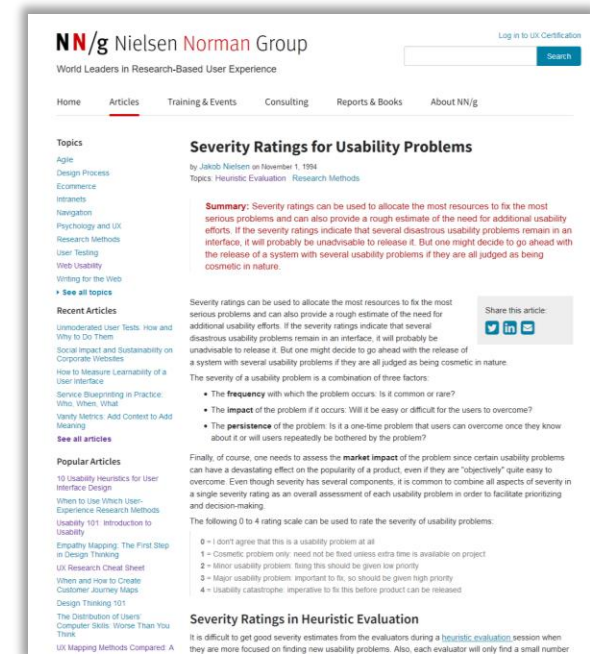
$\frac{PF(i)/N}{Cost(i)}$





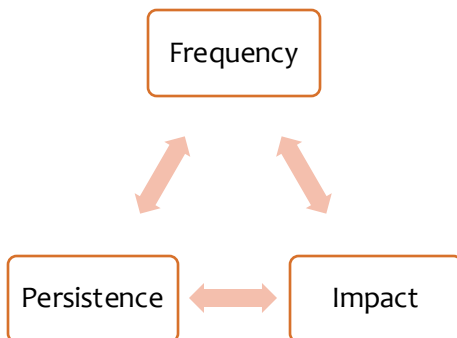
Severity Rating

- We need to allocate the most resources to fix the most serious problems
- We need to understand if additional usability efforts are required
- **Severity** is a combination of:
 - **Frequency** with which the problem occurs: common or rare?
 - **Impact** of the problem if it occurs: easy to overcome or difficult?
 - **Persistence**, is it one-time or will it occur many times to users?
- Define a *combined severity rating*
 - Individually, for each evaluator



Severity Ratings scale

0	No problem	I don't agree that this is a usability problem at all
1	Cosmetic problem only	need not be fixed unless extra time is available on project
2	Minor usability problem	fixing this should be given low priority
3	Major usability problem	important to fix, so should be given high priority
4	Usability catastrophe	imperative to fix this before product can be released



Combined Severity Ratings

- Severity ratings from *one* evaluator have been found *unreliable*, they should not be used
- After all evaluators completed their rankings
 - Either let them discuss, and agree on a consensus ranking
 - Or just compute the average of the 3-5 ratings

Debriefing

- Meeting of all evaluators, with observers, and members of the *development* team
- Line-by-line analysis of the problems identified
 - Discussion: how can we fix it?
 - Discussion: how much will it cost to fix it?
- Can also be used to brainstorm general design ideas

Heuristic Evaluation vs. User Testing

Heuristic Evaluation

- Faster (1-2h per evaluator)
- Results are pre-interpreted (thanks to the evaluators)
- Could generate *false positives*
- Might *miss* some problems

User Testing

- Need to develop software, and prepare the set-up
- More accurate (by definition!)
 - Actual users and tasks
- ... *more on this later in the course!*

Heuristic Evaluation vs. User Testing

Heuristic Evaluation

- Faster (1-2h per evaluator)
- Results are pre-interpreted (thanks to the evaluators)
- Could generate false positives
- Might miss some

- Alternate the methods!
 - Find different problems
 - Do not waste participants

User Testing

- Need to develop software, and prepare the set-up
- More accurate (by definition!)
 - Actual users and tasks

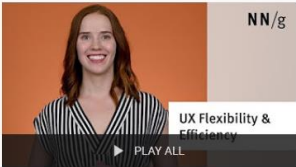
s later in the course!

<https://www.nngroup.com/articles/usability-problems-found-by-heuristic-evaluation/>

Nielsen's Usability Heuristics

10 Usability Principles to be used in Heuristic Evaluation

10 Nielsen's Usability Heuristics



The 10 Usability Heuristics

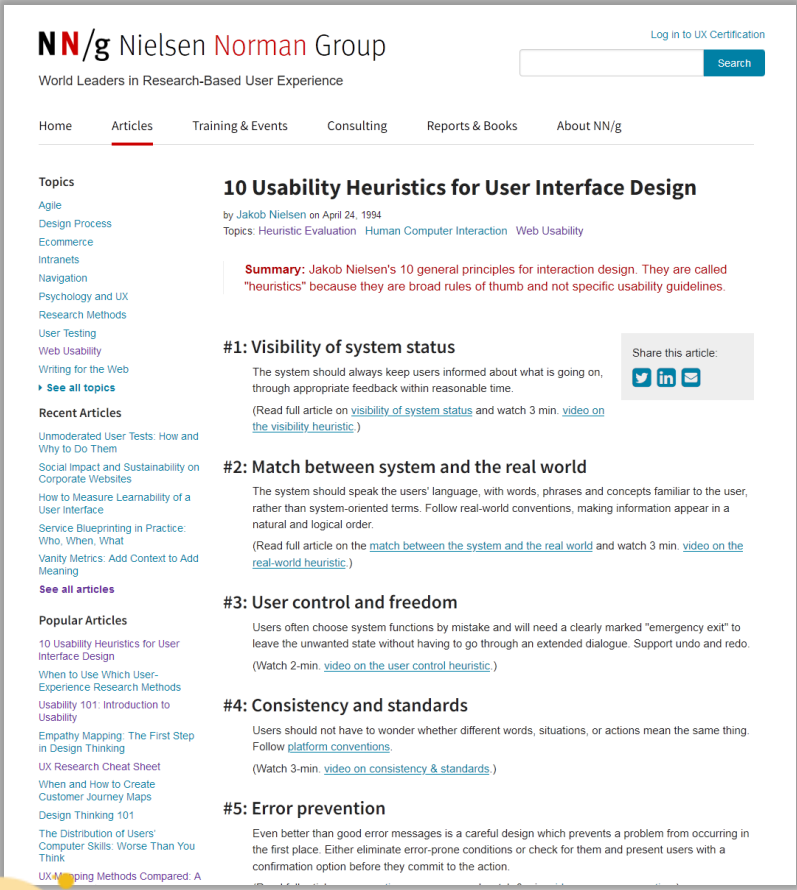
11 videos • 9,192 views • Last updated on Oct 6, 2019

The 10 basic principles for designing a good user experience: these have remained true for decades, since they were introduced for heuristic evaluation of user interfaces. More info: <https://www.nngroup.com/articles/ten-...>

#UX #HeuristicEvaluation

Subscribe

- Usability Heuristic 1: Visibility of System Status (2:37)
- Usability Heuristic 2: Match Between the System and the Real World (3:09)
- Usability Heuristic 3: User Control & Freedom (2:16)
- Usability Heuristic 4: Consistency and Standards (2:38)
- Usability Heuristic 5: Error Prevention (2:53)
- Usability Heuristic 6: Recognition vs. Recall in User Interfaces (2:49)
- Usability Heuristic 7: Flexibility and Efficiency of Use (2:55)
- Usability Heuristic 8: Aesthetic and Minimalist Design (1:58)
- Usability Heuristic 9: Help Users Recognize, Diagnose and Recover from Errors (2:20)
- Usability Heuristic 10: Help & Documentation (2:47)



Log in to UX Certification

Search

World Leaders in Research-Based User Experience

Home Articles Training & Events Consulting Reports & Books About NN/g

10 Usability Heuristics for User Interface Design

by Jakob Nielsen on April 24, 1994

Topics: Heuristic Evaluation Human Computer Interaction Web Usability

Summary: Jakob Nielsen's 10 general principles for interaction design. They are called "heuristics" because they are broad rules of thumb and not specific usability guidelines.

Share this article: [Twitter](#) [LinkedIn](#) [Email](#)

#1: Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

(Read full article on [visibility of system status](#) and watch 3 min. [video on the visibility heuristic](#).)

#2: Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

(Read full article on the [match between the system and the real world](#) and watch 3 min. [video on the real-world heuristic](#).)

#3: User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

(Watch 2-min. [video on the user control heuristic](#).)

#4: Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow [platform conventions](#).

(Watch 3-min. [video on consistency & standards](#).)

#5: Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.



https://www.youtube.com/playlist?list=P_LJOEJ3Ok_idtb2YeifXIG1-TYoMBLoG6I



<https://www.nngroup.com/articles/ten-usability-heuristics/>



10 Nielsen's Usability Heuristics

#1: Visibility of system status

#2: Match between system and the real world

#3: User control and freedom

#4: Consistency and standards

#5: Error prevention

#6: Recognition rather than recall

#7: Flexibility and efficiency of use

#8: Aesthetic and minimalist design

#9: Help users recognize, diagnose, and recover from errors

#10: Help and documentation

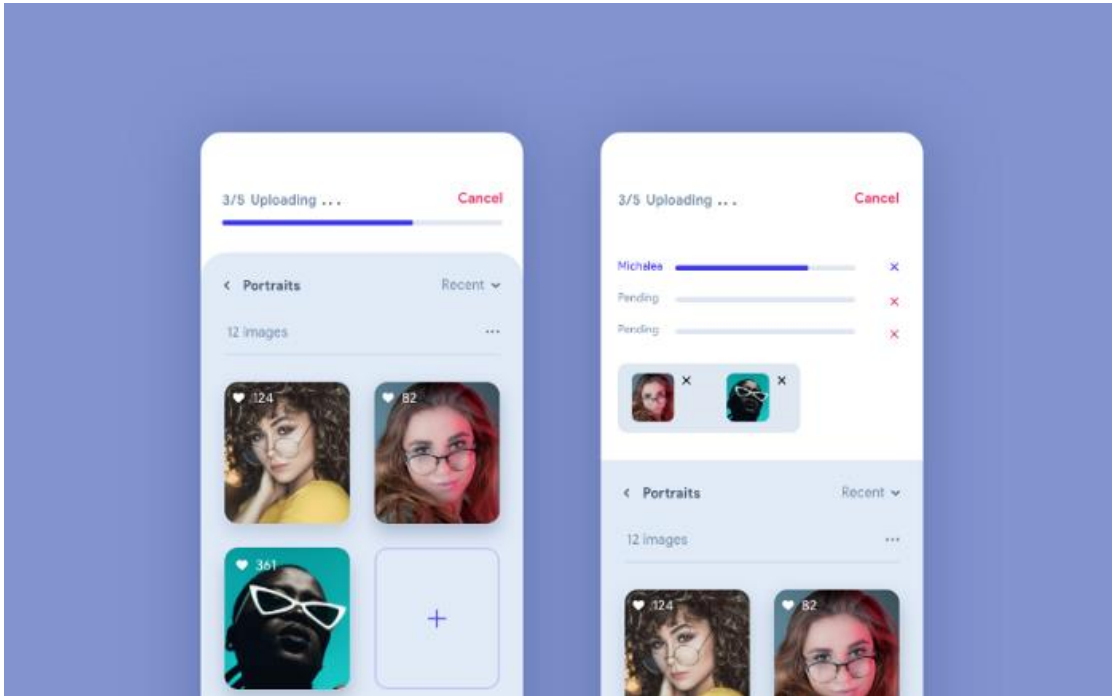
#1: Visibility of system status

- The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.



<https://www.nngroup.com/articles/visibility-system-status/>

#1: Visibility of system status



Some examples from: <http://designingwebinterfaces.com/6-tips-for-a-great-flex-ux-part-5>

Which Feedback?

- Time
 - Execution time for tasks
- Space
 - E.g., occupation of cloud storage
- Change
 - Ensure that the user is aware of changes that he requested (e.g., save, delete, send, ...)
- Action
 - What is happening (running, stopped, ...), in a redundant way
- Next steps
 - What will happen because of your action, and your possible next actions at this point
- Completion
 - Clarify when a task has been finalized

Rule of Thumb (Time)

- If the execution time is...
- ... Less than 1 second \Rightarrow just show the outcome of the action
- ... Around 1-2 seconds \Rightarrow show feedback that the action is underway
- ... More 2-3 seconds \Rightarrow show progress (percentage, estimated time, ...)

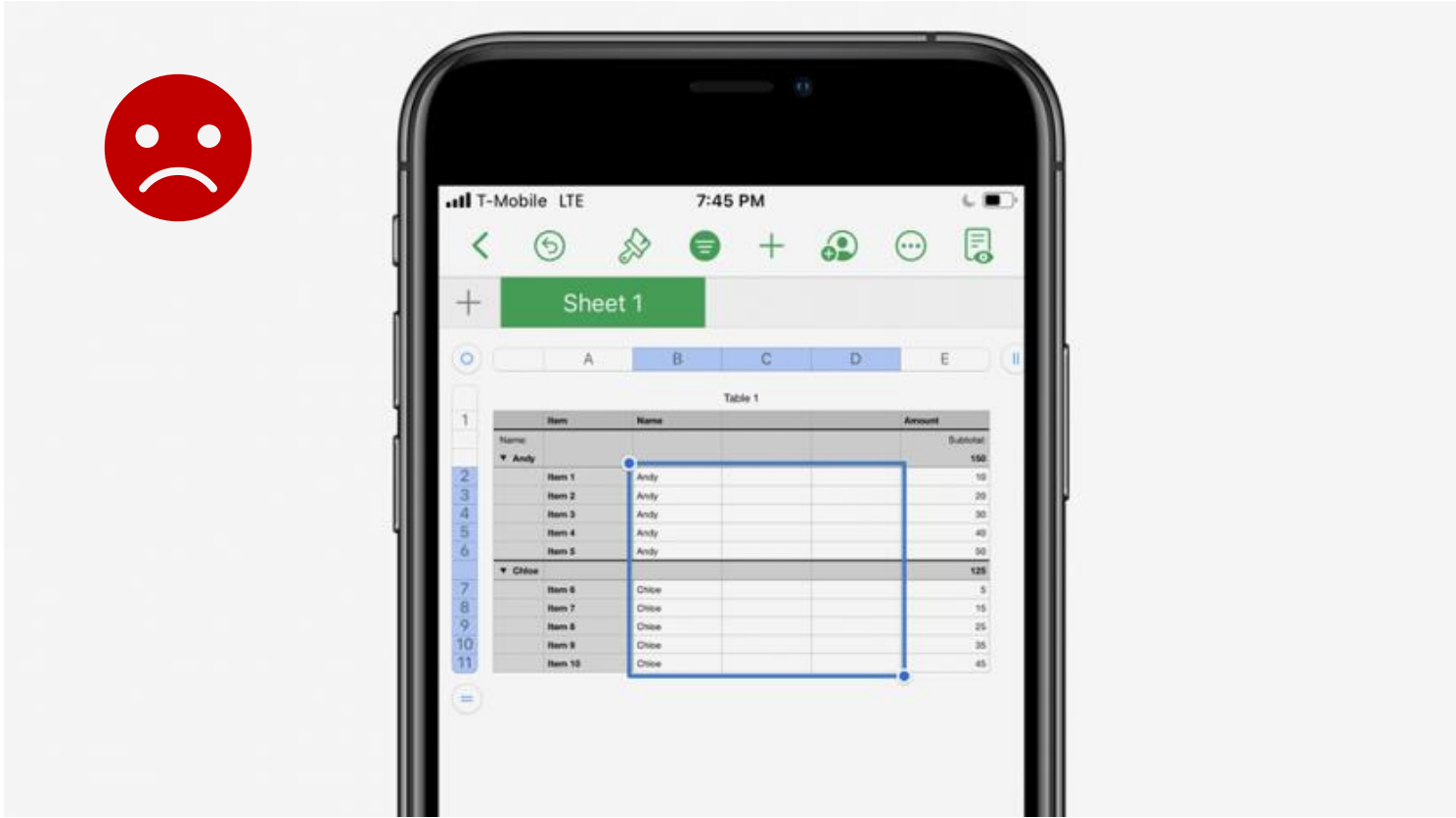
#2: Match between system and the real world

- The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
- Use familiar metaphors and language

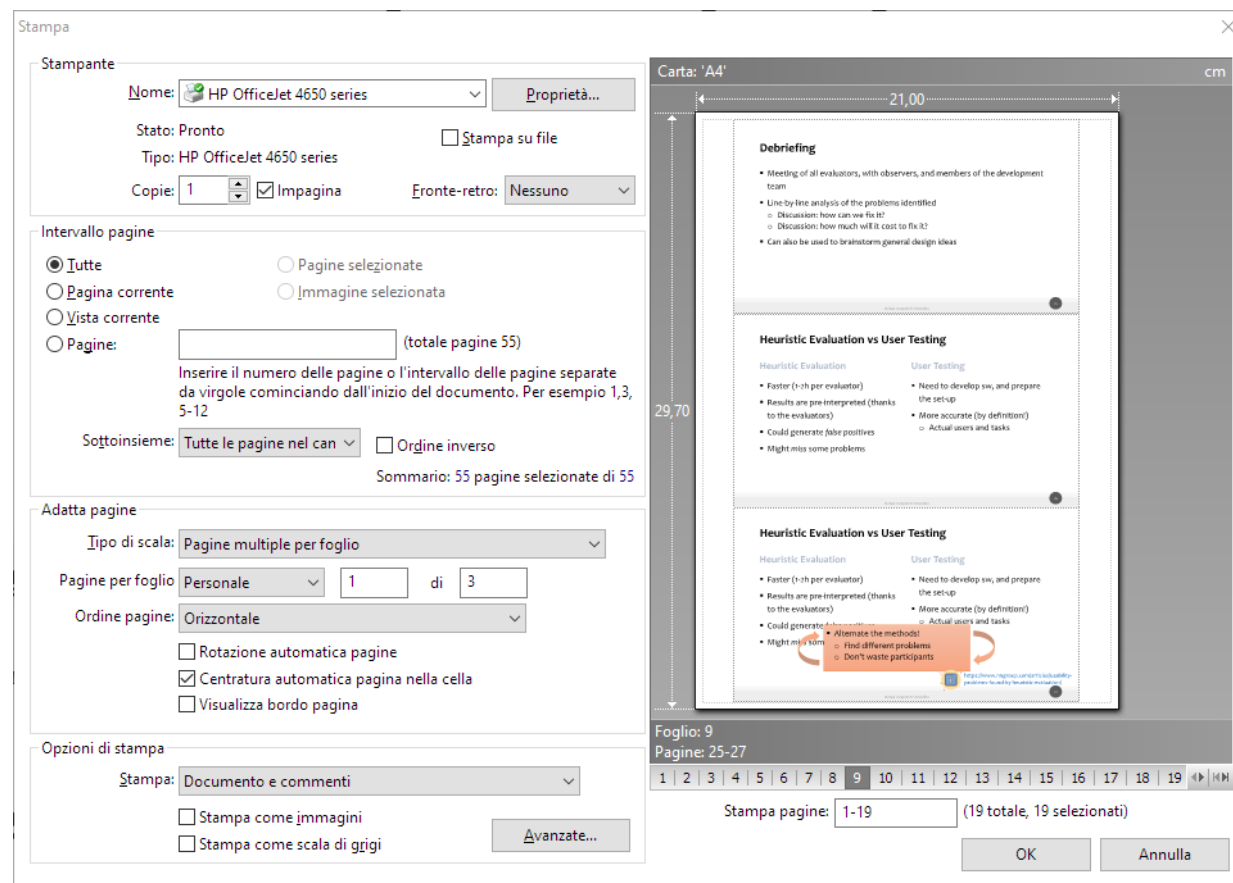
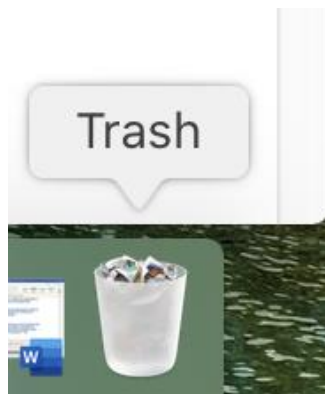
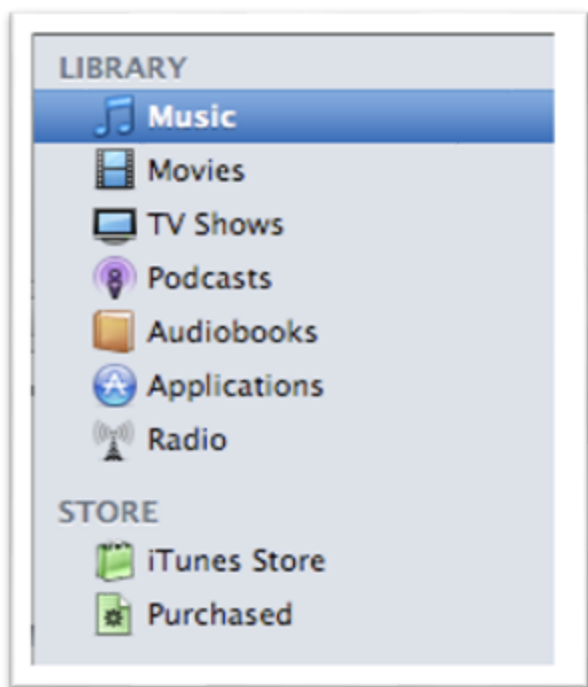


<https://www.nngroup.com/articles/match-system-real-world/>

#2: Match between system and the real world



#2: Match between system and the real world



Exploit Familiarity

- Familiar Metaphors
 - Files, paper, folders, highlighters, ...
- Familiar Language
 - Avoid jargon, acronyms, etc. that could be unknown to your users
- Familiar Categories
- Familiar Choices
 - E.g., explain the meaning of the error message (what happened, what are the consequences, what are the available options) in a simple way

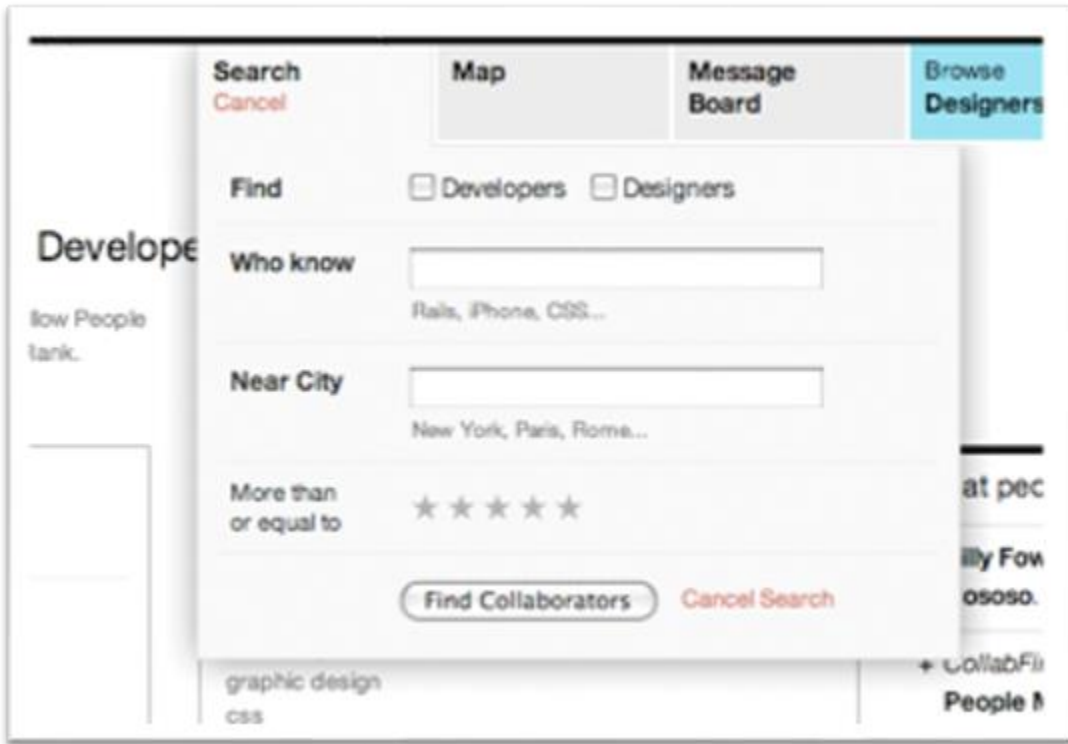
#3: User control and freedom

- Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

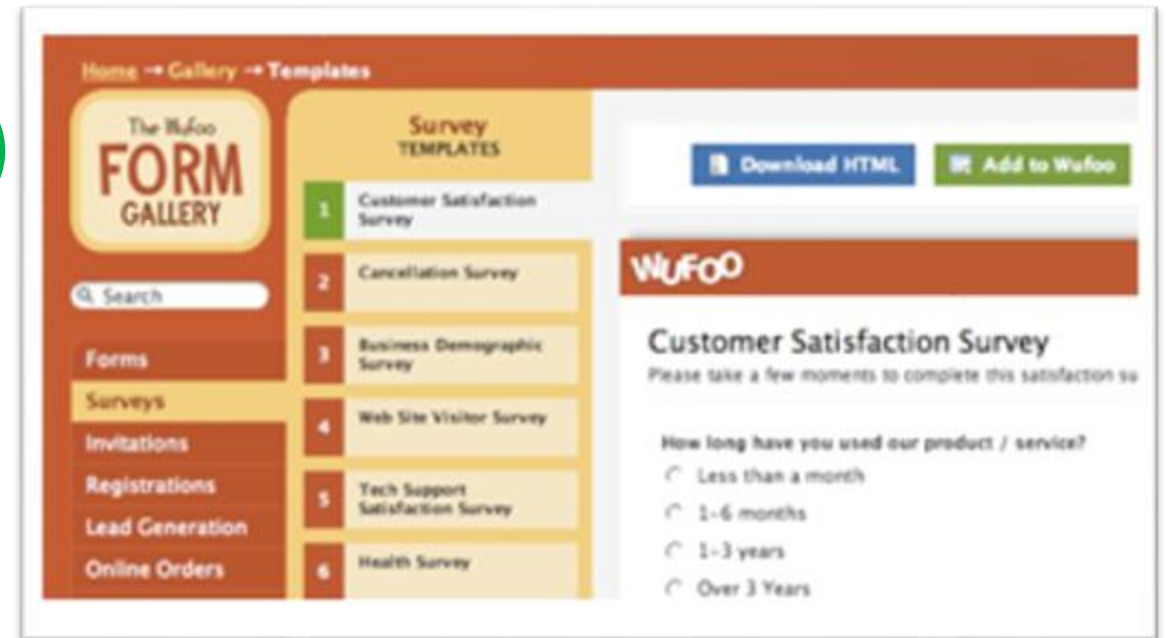
#3: User control and freedom



#3: User control and freedom



	A	B	C	D
1	Item	Quantity	Price	Total
2	Tacos	40	\$5.00	= B2 * C2
3				



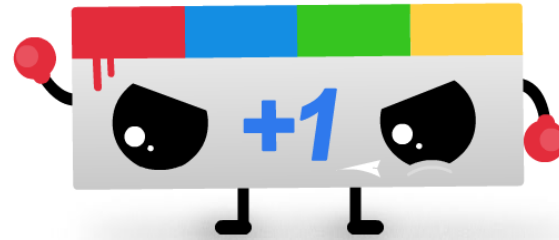
Suggestions

- Always provide a “back” (or equivalent) button
- Allow users to “explore” different alternative paths
 - Except for one-shot wizard-like paths, aimed at novices or first-time users

#4: Consistency and standards

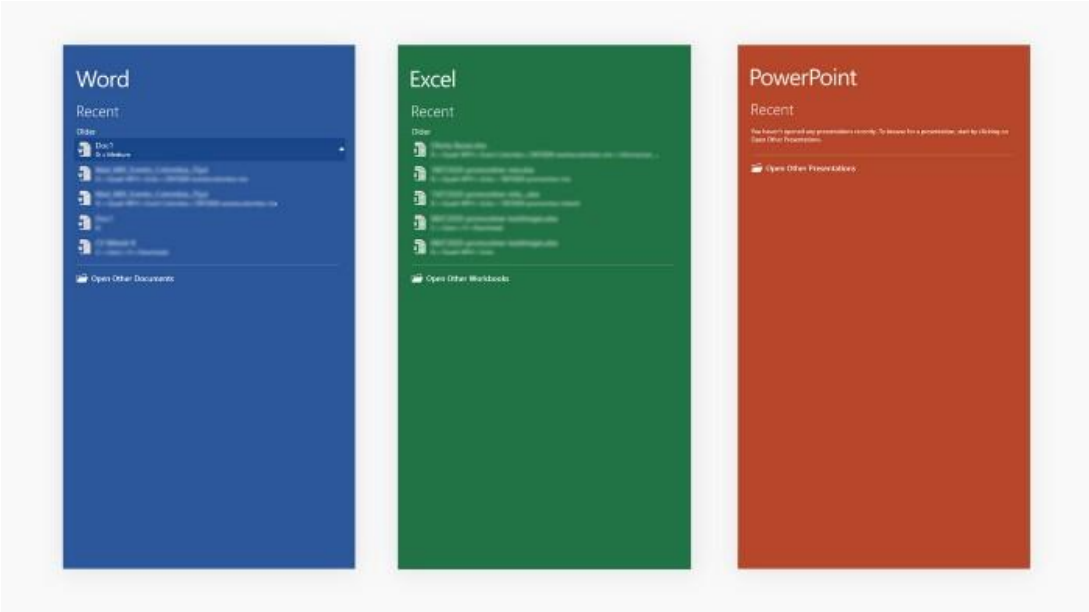
- Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

#4: Consistency and standards



BrandFlakesforBreakfast's Illustration

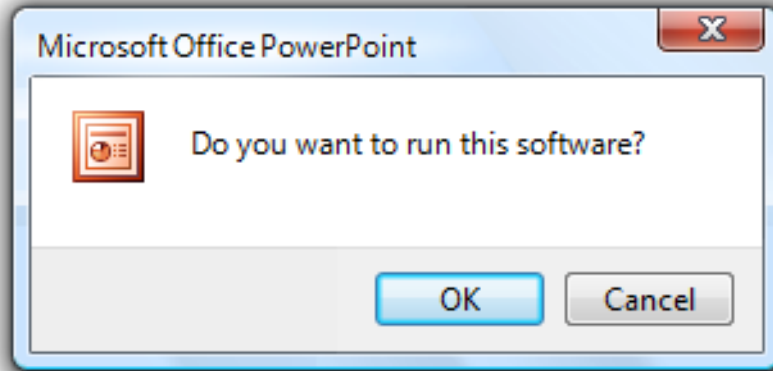
#4: Consistency and standards



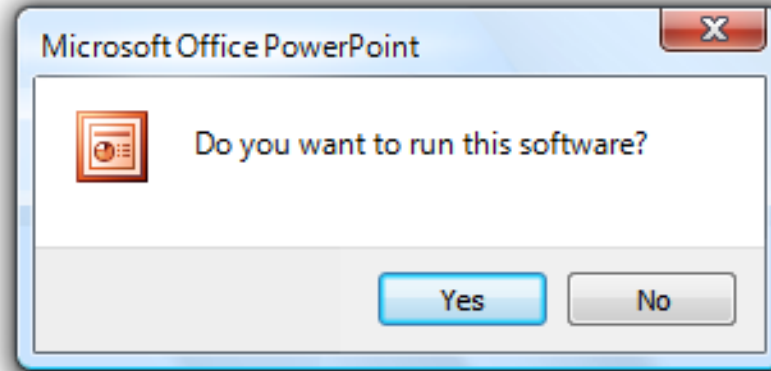
Suggestions

- Consistent layout for dialogs and forms
 - E.g., position of the navigation elements
 - E.g., position of the confirmation buttons
- Consistent meaning for Ok/Cancel, Yes/No choices
 - E.g., avoid: “Do you want to interrupt task?”
 - Still better, label buttons with the actual effect “Insert”, “Interrupt”, ...
- Categories, lists of names, geographical regions, etc, should be taken from “standard” vocabularies

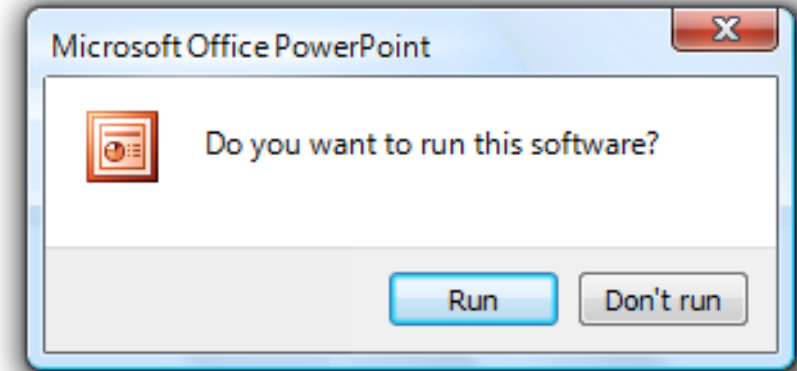
Examples



Bad



Acceptable



Better

source: <https://docs.microsoft.com/en-us/windows/win32/uxguide/win-dialog-box>

#5: Error prevention

- Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.



<https://www.nngroup.com/articles/slips/>

Suggestions

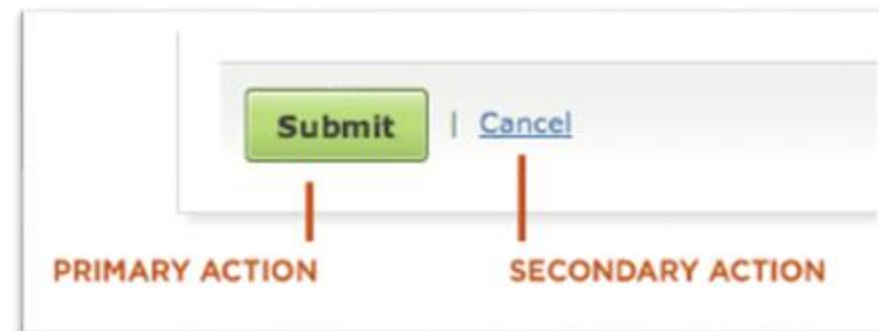
- Preventing data loss
- Prevent clutter
- Prevent confusing flow
- Prevent bad input
- Prevent unnecessary constraints (e.g., provide defaults for missing data)

#5: Error prevention



A screenshot of an email client interface. At the top, there are navigation icons: a back arrow, a reply arrow, a reply all arrow, and a forward arrow. To the right are action buttons: Archive, Move, Delete, Spam, and a three-dot menu. Below this is a header for the email, showing a 'test' subject and 'Yahoo/Sent' status. The main body of the email shows a profile picture of a person with a blue 'T' on a circular background, followed by a large black rectangular redaction box. Below the redaction, the text reads 'Please see the attached files.' To the right of the text is a timestamp 'Dec 11 at 1:04 AM' and a star icon. Below the email body is a row of reply and forward icons. At the bottom, there is a 'Send' button and a row of icons for attachments (paperclip), GIFs, stickers, emojis, links, bold text, italic text, background color, text color, and a trash icon.

#5: Error prevention



#5: Error prevention

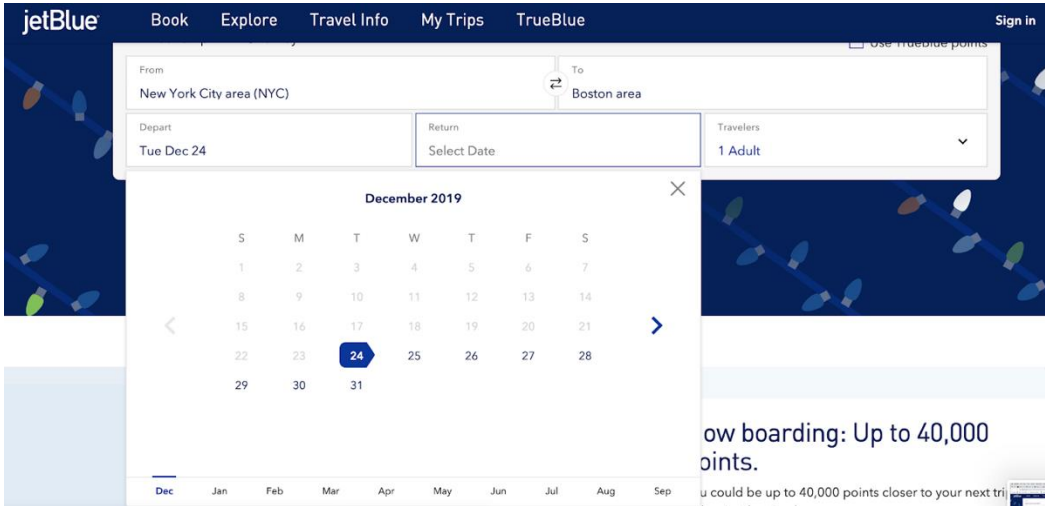


Google

olyimpic

- olympics
- olympics 2016
- olympic trials
- olympics schedule

Press Enter to search.



jetBlue

Book Explore Travel Info My Trips TrueBlue Sign in

From: New York City area (NYC) To: Boston area

Depart: Tue Dec 24 Return: Select Date Travelers: 1 Adult

December 2019

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Now boarding: Up to 40,000 points.

You could be up to 40,000 points closer to your next trip.

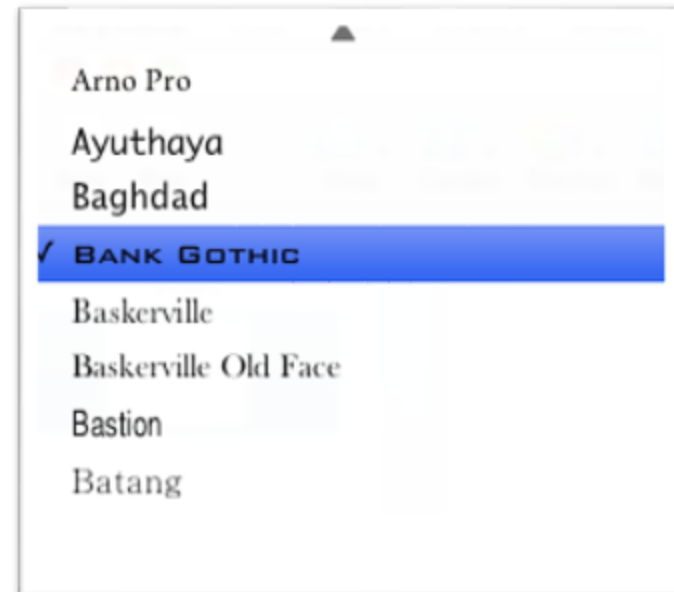
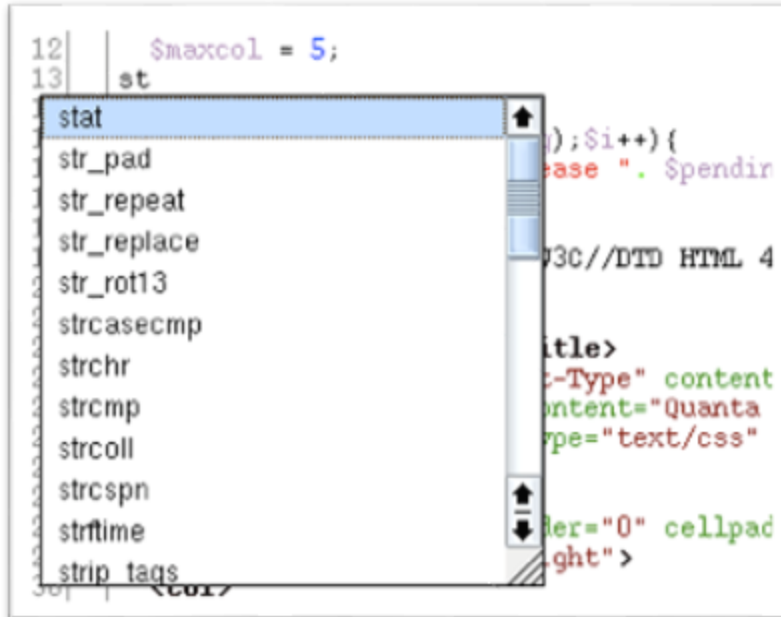
#6: Recognition rather than recall

- Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the interface to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.



<https://www.nngroup.com/articles/recognition-and-recall/>

#6: Recognition rather than recall



Example



g domain and is intended to provide

implementation of a computational notebook as a tool to support its development. Through the analysis of the use case and the landscape of the current computational notebooks, we determined that besides the features of the current computational notebooks an IoT notebook must enable (i) multiple programming languages in the same notebook; (ii) the capability to execute code in the documents in external devices; (iii) keep some code snippets on background execution; (iv) support the specification and installation of mandatory dependencies; and (v) support the visualization of data coming from the sensing devices or external services and platforms. By implementing a prototypical system of the IoT notebook and by validating it against the use case, we could conclude that special



```
\section{Related Work}
\label{sec:related-work}

This work lies in the software engineering domain and is intended to provide insights about the suitability of a computational narrative approach to document, execute, and share the steps involved in IoT prototyping, especially for novice programmers.

%To the best of our knowledge, \highlight{no other authors}\footnote{it's a strong statement... are we absolutely sure?}

have explored this strategy. In the following, we addressed the related work from the perspective of (i) exploring and analyzing the current use of notebooks, and (ii) customizing them to fit into a particular context.

In~\cite{Corno:2019} we propose a first approach to an IoT-tailored literate computing tool in the form of a computational notebook. In this article we presented a use case of a typical IoT system involving several interconnected components and described the implementation of a computational notebook as a tool to support its development. Through the analysis of the use case and the landscape of the current computational notebooks, we determined that besides the features of the current computational notebooks an IoT notebook must enable (i) multiple programming languages in the same notebook; (ii) the capability to execute code in the documents in external devices; (iii) keep some code snippets on background execution; (iv) support the specification and installation of mandatory dependencies; and (v) support the visualization of data coming from the sensing devices or external services and platforms. By implementing a prototypical system of the IoT notebook and by validating it against the use case, we could conclude that special attention should be paid on how to execute the code snippets on external devices, and a more in-depth assessment of the benefits and limitations of a computational narrative in the context of IoT software development and prototyping is needed.

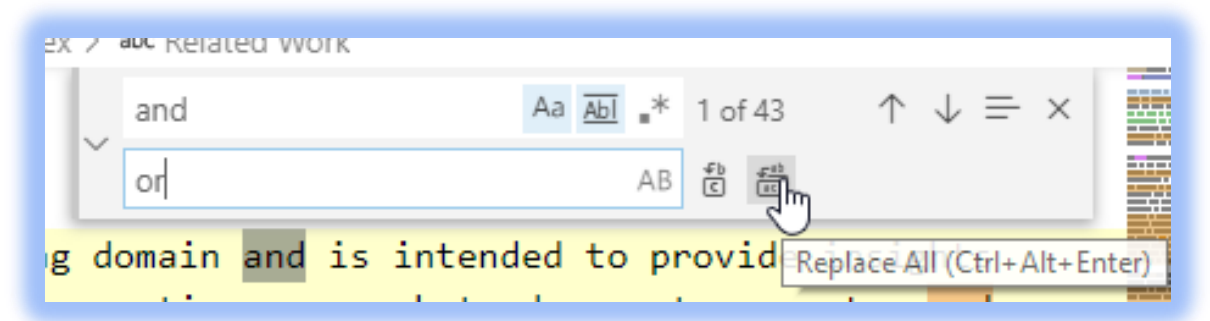
Rule~\textit{et al.}~\cite{Rule:2018} assessed the current use of computational notebooks through quantitative analysis of over 1 million notebooks shared online, qualitative analysis of over 200 academic computational notebooks, and interviews with 15 academic data analysts. These analyses demonstrated a tension between exploration and explanation that computational notebooks should address.

:g/\<and\>/s//or/g
```

:g/\<and\>/s//or/g

Suggestions

- Avoid codes (use explicit names)
 - e.g., L, VL, EL, EA, ... ???
- Avoid extra hurdles
 - e.g., asking for unnecessary (or premature) information
- Provide previews
 - Code completion
 - Page preview
 - Order summary
 - Itinerary
 - ...



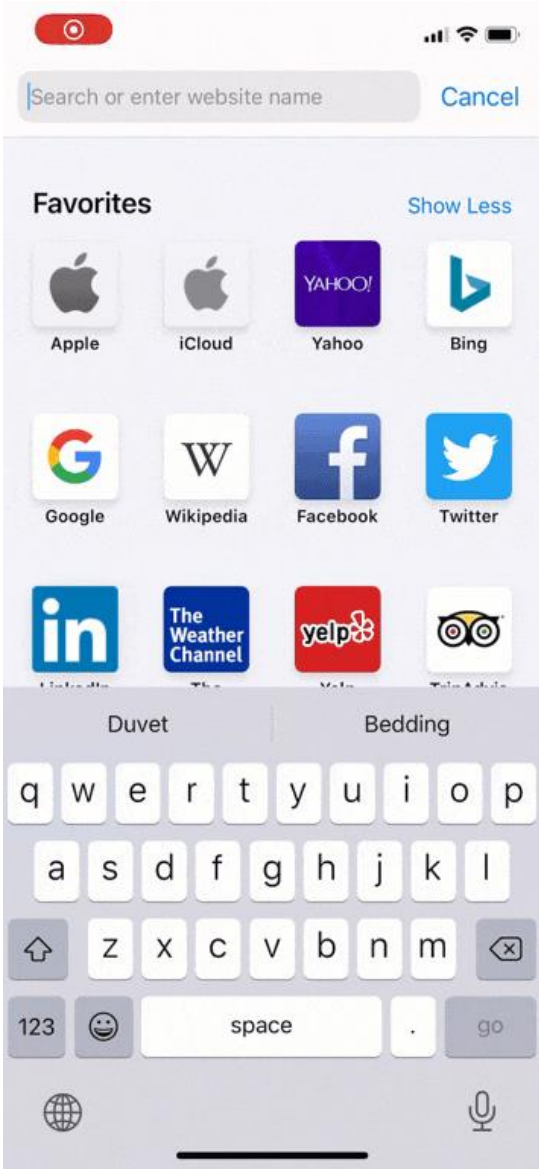
#7: Flexibility and efficiency of use

- Accelerators — unseen by the novice user — may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

#7: Flexibility and efficiency of use

Common Shortcuts	
Add Action	Return
New Window	⌘N
Synchronize with Server	⌘S
Clean Up	⌘K
Planning Mode	⌘1
Context Mode	⌘2
Inbox	⇧⌘1
Quick Entry	⇧⇧Space

Quick Entry's shortcut can be customized in Preferences



Suggestions

- Flexibility = Default + Options
 - E.g., present some popular choices, but let the user enter a custom one (train ticket machines)
- Exploit background information for providing more information
 - E.g., weather forecasts in a calendar interface
- Novice and Expert Users Have Different Needs
 - Support proactivity, personalization, and different interaction techniques!
- Recommendations
- Provide relevant information, only

#8: Aesthetic and minimalist design

- Interfaces should not contain information which is irrelevant or rarely needed. Every extra unit of information in an interface competes with the relevant units of information and diminishes their relative visibility.

#8: Aesthetic and minimalist design

rand of Gate Openers and Operators, Elite, Viking, DoorKing, Power Master, Ramset, Allstar, FAAC, Apollo, SEA. We are also manufacturer of Custom gates in Aluminum, Iron, Steel our Ornamental gates are second to none. For your Driveway automatic electric gates entrance we offer a full line of Access ent, Telephone entry system, Intercom, keypad and gate accessories and safety devices, loop detector, safety loop, photo cell. Offering complete Custom decorative fencing matching designs and style, 100's of fence pictures to choose, picket fence, deck, pool, garden, estate, modern, we have it all at gre

Welcome To: **Gates N Fences**

L.A. Ornamental Corp
3708 N.W. 82nd Street
Miami, Florida 33147
Phone: 305-696-0419
LAOrnamental@aol.com

Designed to Enhance the Entry of your home with Custom Ornamental Decorative Driveway Gates while bringing Safety, Security and convenience.

Home

Driveway Gates

Modern Driveway Gates

Custom Driveway Gates

Privacy Driveway Gates

Garden Gates

Modern Garden Gates

Privacy Garden Gates

Fencing

Fencing_2

Railings

Modern Balcony Railings

Openers - Operators

Ramset

FAAC



Top Brands of Gate Openers and Operators, Commercial, Residential, Industrial, Swing, Slide, Rack & Pinion, Barrier

- BFT Gate Openers
- PowerMaster Gate Openers
- FAAC Gate Openers



Search

All of our Aluminum or [Wrought Iron Gates](#), or Fences are designed and manufactured to withstand a range of outdoor conditions. Our commitment to our customers and dedication to produce quality gates has earned us thousands of satisfied customers.

Although we offer a wide selection of Ornamental Designs or Decorative Designs, we can design and manufacture any style in aluminum or wrought iron metals. L. A. Ornamental & Rack Corp also offers Fences, Garden or Walk Thru Gates to match your driveway gates. With over thirty five years of experience in manufacturing and designing elegant, custom, or exotic [Aluminum Driveway Gates](#) and Fences, our past and future customers can have peace of mind that they are receiving quality workmanship. We are a Fence Company that gives our customers 110% of dedication to manufacture quality driveway gates and fences.

For a quote please send an e-mail to LAOrnamental@aol.com

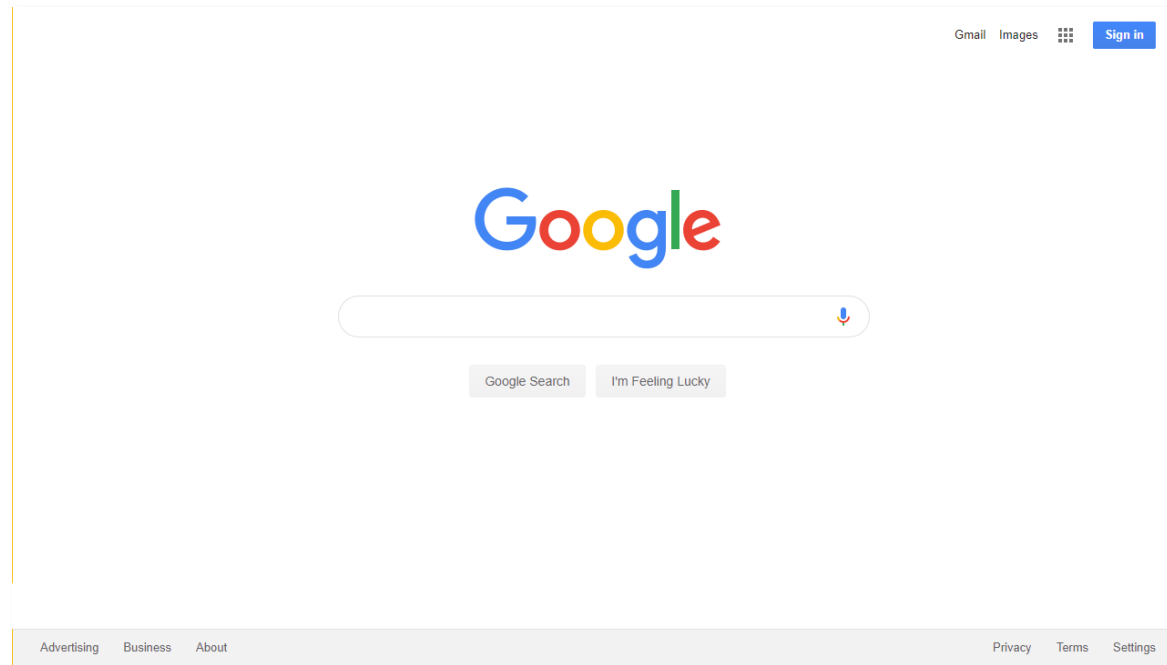
If your looking For Privacy with your Driveway Gates ,Garden Gates, or Walk Thru Gates, we offer a Solid Backing with your choice of Aluminium, Steel, Plexiglas or Plastic. All solid backing are offered in many different colors to choose from. [Privacy Gates](#)

We offer a large selection of Gate Openers and Gate Operators for Residential Driveway Gates, Light or Heavy Commercial Gates, or industrial locations. If your not sure the style or size of the Gate opener / gate operator you need, please e-mail or contact us so we can gladly help guide you to the correct choice. We offer all type of Gate Openers / Gate Operator, Sliding Gate Openers / Gate Operator, Swing Gate Openers / Gate Operator, Hydraulic Gate Openers / Gate Operator. We also have a wide selection of replacement [Main Circuit Boards](#) for all brands, and [Remote Controls](#) for Visors or Keychains.

[Railings](#) - L. A. Ornamental Rack Corp offers top quality Balcony Railings, Front Porch Railings, Deck Railings in Metal, Aluminum, or Wrought



#8: Aesthetic and minimalist design



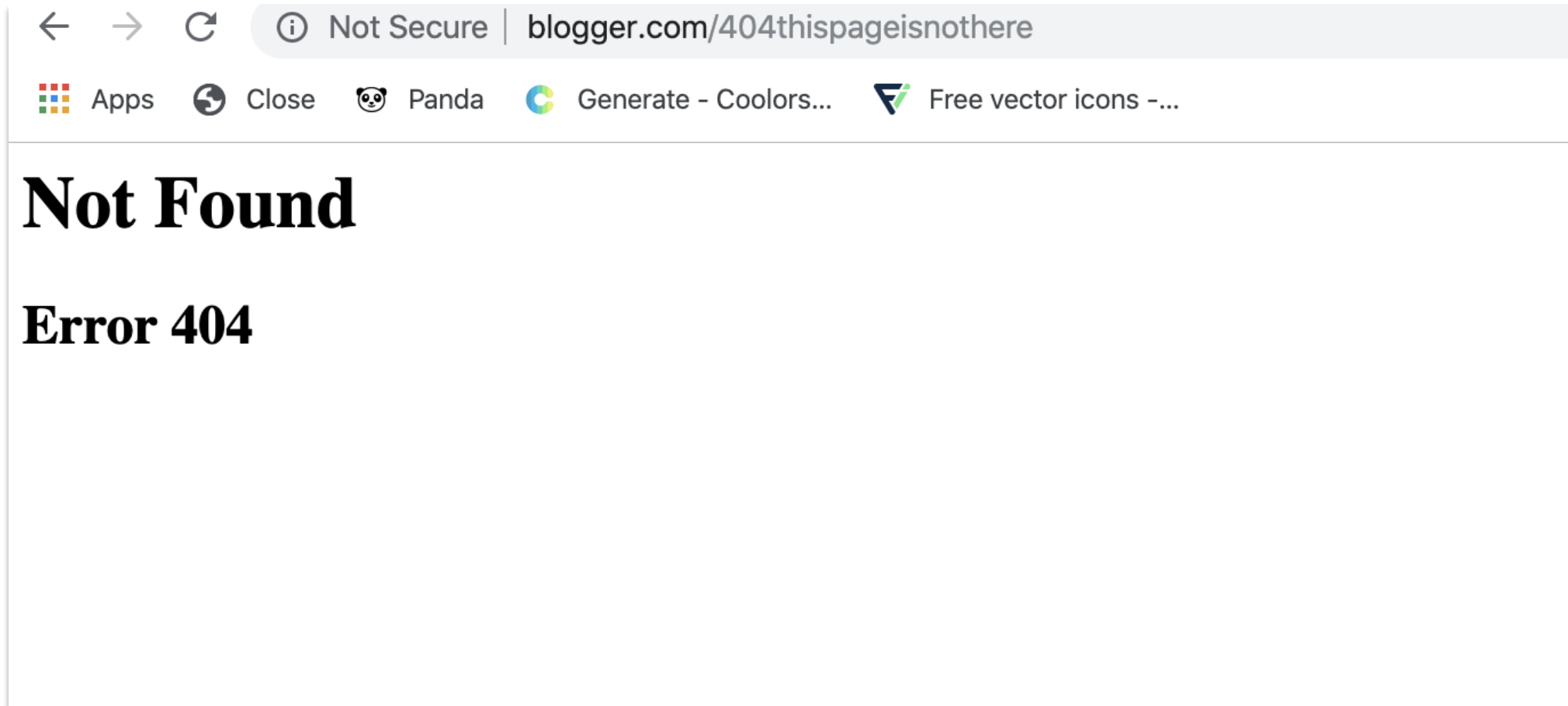
Suggestions

- Key information must be “above the fold”
 - Especially on low-resolution devices
- Keep high signal-to-noise ratio
 - Colors, fonts, backgrounds, animations, ...
 - Borders, dividers, ...
- Minimalistic login experience
- Accept redundant ways of entering information
- Prune features that are outside the “core” functionality

#9: Help users recognize, diagnose, and recover from errors

- Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

#9: Help users recognize, diagnose, and recover from errors



#9: Help users recognize, diagnose, and recover from errors

Or start a new account


Choose a username (no spaces)
bert ⚠ bert is already taken. Please choose a different username.

Choose a password
*** ⚠ Passwords must be at least 6 characters and can only contain letters and numbers.

Retype password

Email address (must be real)
not an email ⚠ The email provided does not appear to be valid

Send me occasional Digg updates.



Oh no!

It seems the page you were trying to find on my site isn't around anymore (or at least around here).

[Report it missing using my contact form](#) and I'll see what I can do about it.

Whilst your here why not check out my [articles listing](#) or [browse my blog](#)? You never know - you may just

Suggestions

- Make errors easy to identify
 - Colors, fonts, ...
- Make problem clear
 - Problem cause
 - Problem location
- Provide a solution
 - Give a suggestion
 - Show a path forward
 - Propose an alternative

#10: Help and documentation

- Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

#10: Help and documentation




#10: Help and documentation



Today








[Redacted]

 **Slackbot** 3:51 AM

I searched for that on our Help Center. Perhaps these articles will help:

- [An introduction to Slackbot](#)
- [Improve company culture with Slack](#)
- [Getting started for workspace creators](#)

Message Slackbot

 **B** *I*  `</>`     **Aa** @ 

Suggestions

- Provide examples
 - In documentation
 - In complex choices
- Help the user understanding the error gravity
 - E.g., printing outside margins
- Provide ‘tips’ for showing new actions or steps
- Use pop-overs to point to changes in UI (or for first usage)
- Avoid too-opaque “terms and conditions” (summarize, if possible)

Example

- Target website: <https://trenitalia.com/>
 - Trenitalia is the primary train operator in Italy
 - It offers national rail transport with regional trains and high-speed trains (“Frecciarossa”)
- Useful tasks to spot several problems:
 - Explore the offers proposed by the website and buy a discounted ticket
 - Buy a “Frecciarossa” round trip from Turin to Rome for the winter holidays
 - Chat with an operator for receiving support
- In performing the tasks, you can register/login to the platform and change the language of the website, if you want



Example - Template

[Issue #]. [Heuristic #] [Heuristic Title]

- *Where: [Where the issue occurred – task, step, page]*
- *What: [Description of the problem]*
- *Why: [Reason why it violates the heuristic]*
- *Severity: [0-4, according to Nielsen's severity rating]*

1. H4 Consistency and standards

- *Where: Specify your language.*
- *What: The app uses “Save” for saving information, except here where it uses “Store”.*
- *Why: It is an inconsistent terminology for the same function in the application, which can create confusion.*
- *Severity: 3*

References and Acknowledgment

- Alan Dix, Janet Finlay, Gregory Abowd, Russell Beale: Human Computer Interaction, 3rd Edition
 - Chapter 9: Evaluation Techniques
- Ben Shneiderman, Catherine Plaisant, Maxine S. Cohen, Steven M. Jacobs, and Niklas Elmqvist, Designing the User Interface: Strategies for Effective Human-Computer Interaction
 - Chapter 5: Evaluation and the User Experience
- COGS120/CSE170: Human-Computer Interaction Design, videos by Scott Klemmer, https://www.youtube.com/playlist?list=PLLsT5z_DsK_nusHL_Mjt87THSTlgrsyJ
- Thanks to Fulvio Corno, past teacher of the course, for his work on some of these slides



License

- These slides are distributed under a Creative Commons license “**Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)**”
- **You are free to:**
 - **Share** — copy and redistribute the material in any medium or format
 - **Adapt** — remix, transform, and build upon the material
 - The licensor cannot revoke these freedoms as long as you follow the license terms.
- **Under the following terms:**
 - **Attribution** — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
 - **NonCommercial** — You may not use the material for [commercial purposes](#).
 - **ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the [same license](#) as the original.
 - **No additional restrictions** — You may not apply legal terms or [technological measures](#) that legally restrict others from doing anything the license permits.
- <https://creativecommons.org/licenses/by-nc-sa/4.0/>

