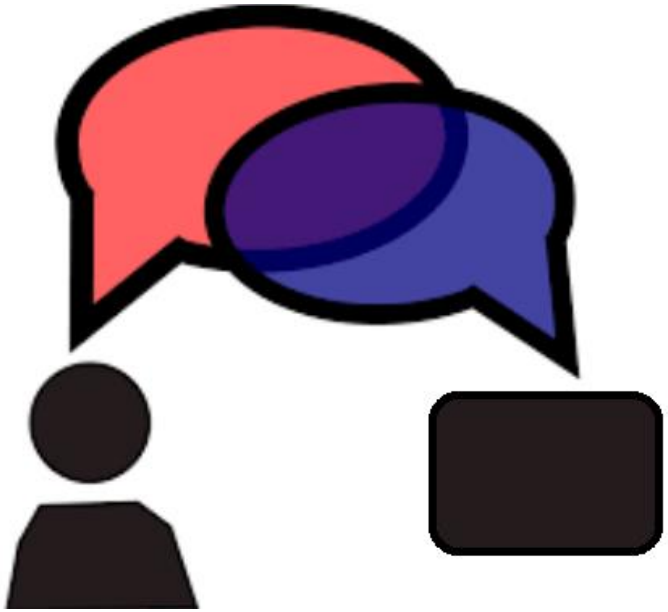
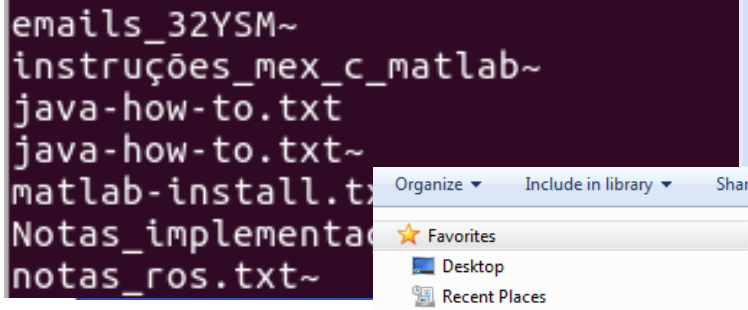
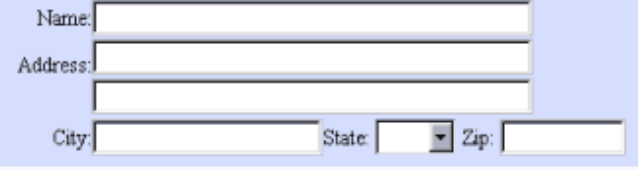
Universidade de Aveiro
Departamento de Electrónica,
Telecomunicações e Informática

Other Interaction Styles



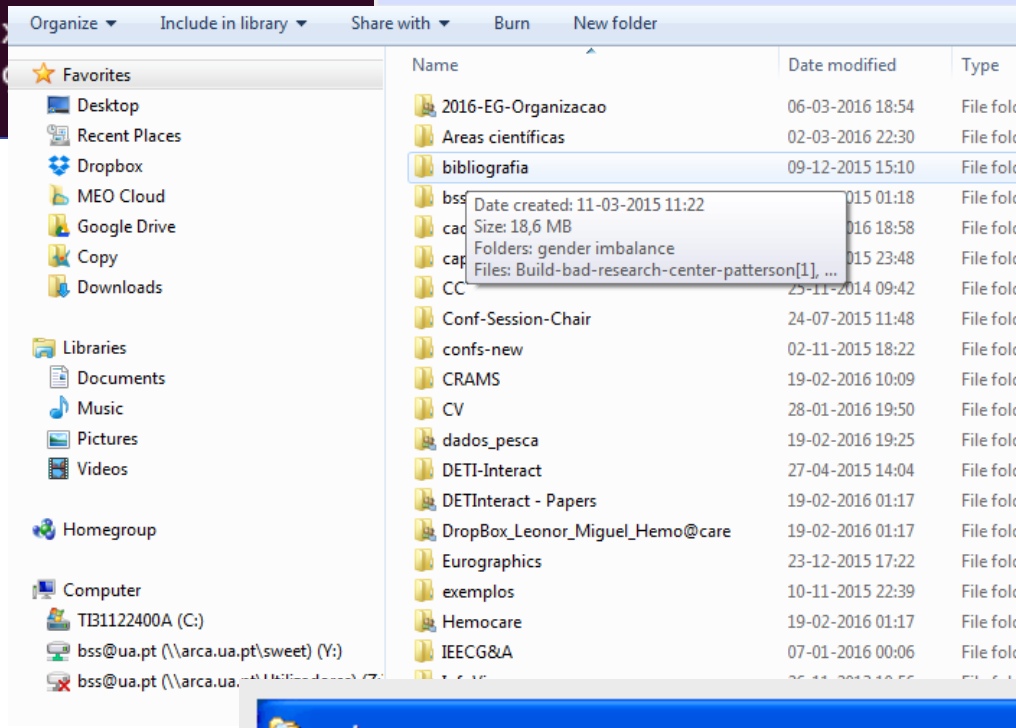
Interaction/ Dialog styles

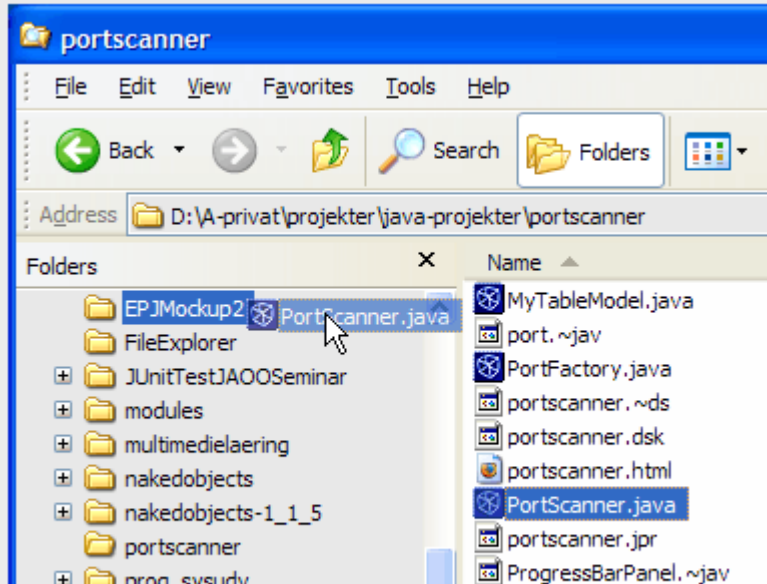




A possible classification:

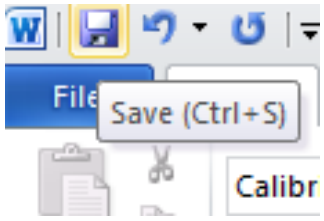
- Menus ✓
- Direct manipulation ✓
- Fill-in-forms
- Function keys
- Question and answer
- Command languages
- Natural languages
- …



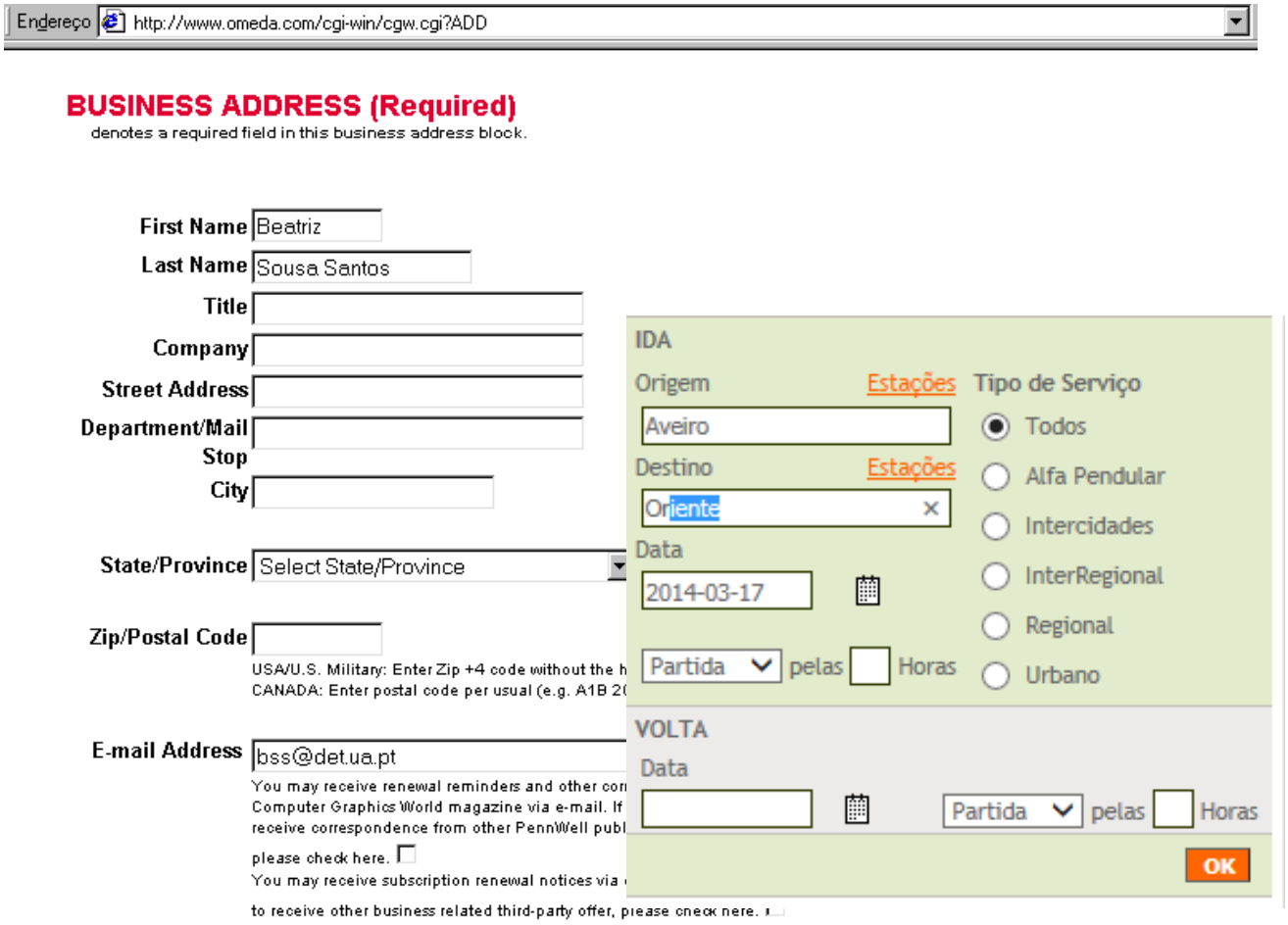


Often two or more styles are used simultaneously;

Why?



Fill in forms



- Fill in forms are particularly useful for routine, clerical work or for tasks that require much data entry

- The concept already existed long ago

- Currently they are often used with other styles

```
  PINE 3.96   ADDRESS BOOK (Edit)

Nickname  : NBA
Fullname  : Players in the NBA
Fcc       :
Comment   :
Addresses : mjordan@nba.com,
            kmalone@nba.com,
            drobinson@aol.com

^G Get Help  ^X eXit/Save ^R RichView  ^Y PrvPg/Top
^C Cancel                              ^V NxtPg/End
```

Main advantages and disadvantages

Advantages (potential)

- Self-explanatory
- Recognition instead of recall
- Allow many different inputs (unlike menus)
- Give context and guide the user
- New functionality is visible (unlike command languages)

Disadvantages

- Imply knowledge of valid inputs
- Error prone
- Not very flexible

Fill in form design: relevant aspects in design

- Organization and layout
- Titles and fields
- Input formats
- Instructions and help
- Navigation
- Error handling

Fill in form design: guidelines

Which is preferable?

Example:

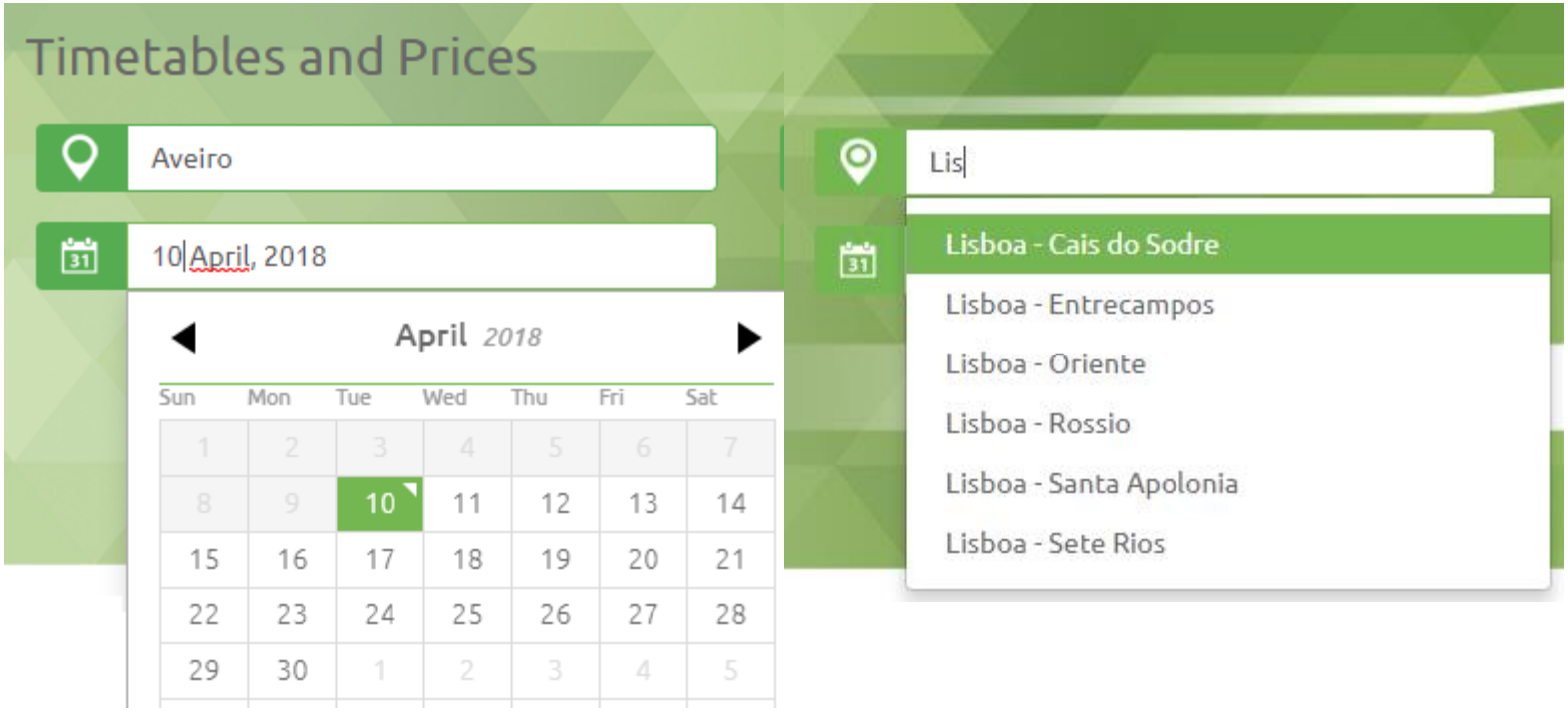
Zip code:
Name:
Country:
Address:
City:

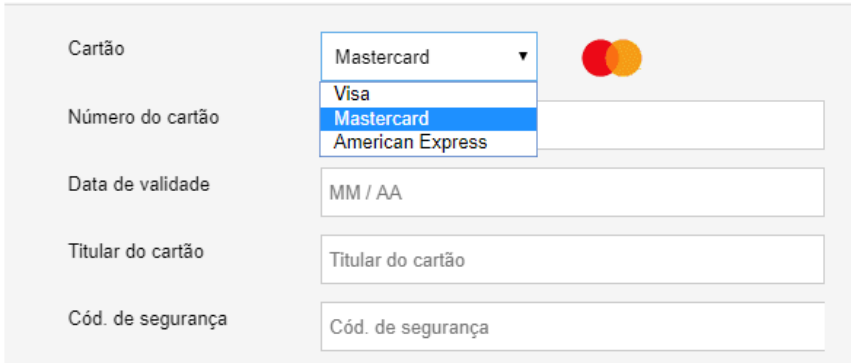
Better:

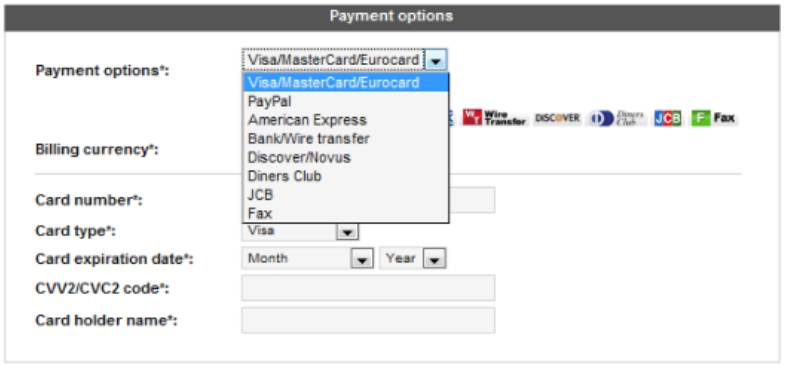
Name:
Address:
Zip code:
City:
Country:

Avoid unfamiliar layouts!

Provide a menu when possible inputs are known
(combining two interaction styles…)

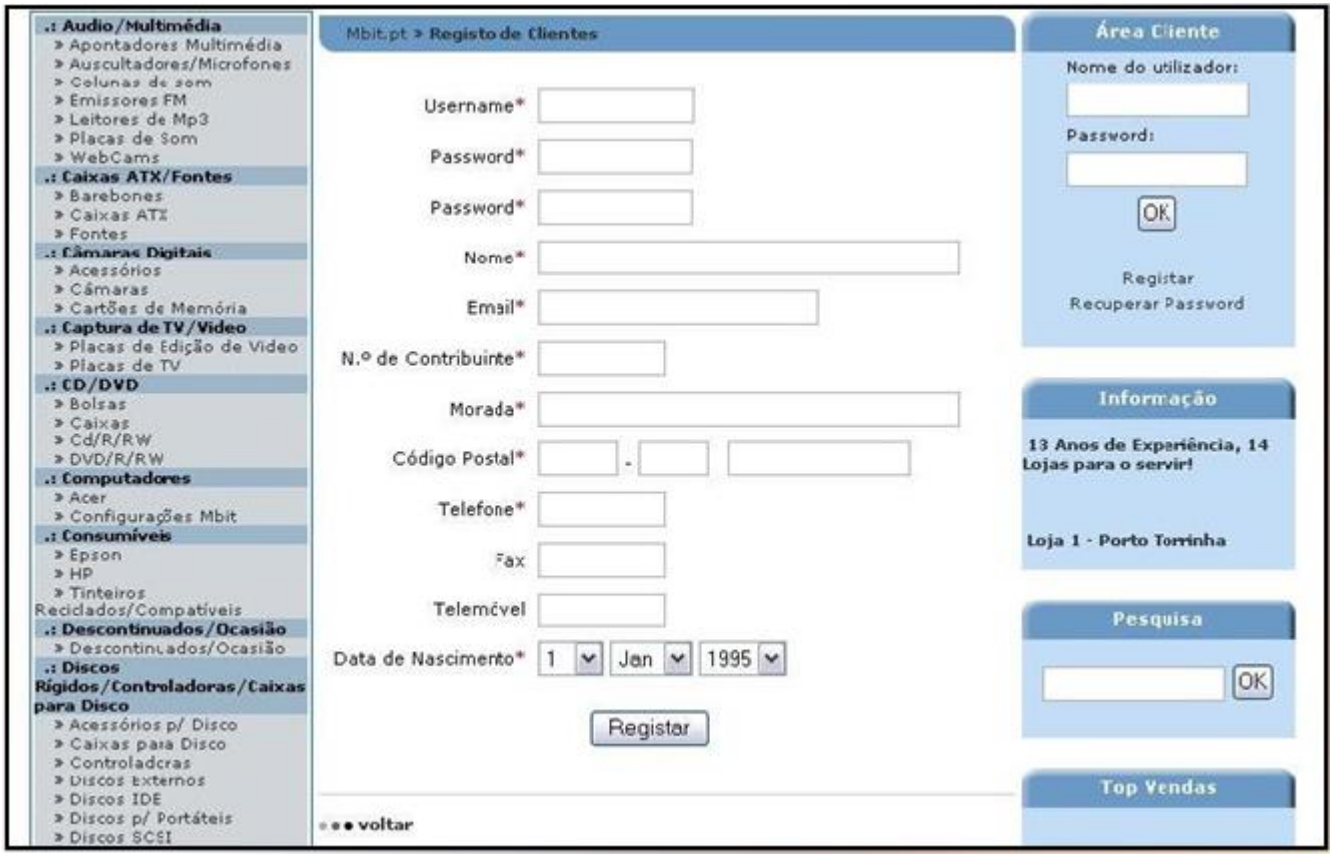






Provide a format for fields that may be ambiguous

Show which fields are mandatoty



Usually indicated by *

Input format must be familiar and clear

Date:_____
(eg. 1/12/2000)

Date:_____
(e.g. 01122000)

Time:_____
(eg. 8-15)

Time:_____
(e.g. 0815)

Card#:_____
(e.g. 123456789012)

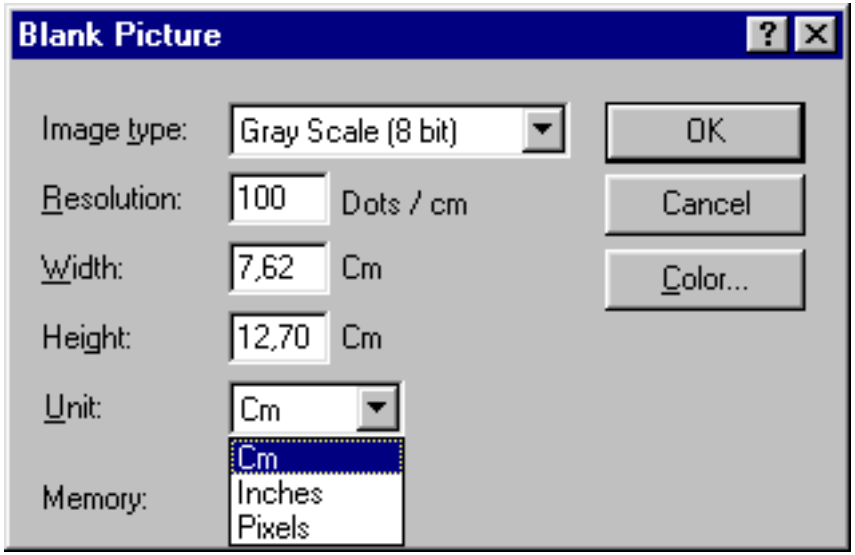
Better:

Date:___/__/_____
(e.g. 1/ 12 /2022)

Time:___-_____
(e.g. 08-15)

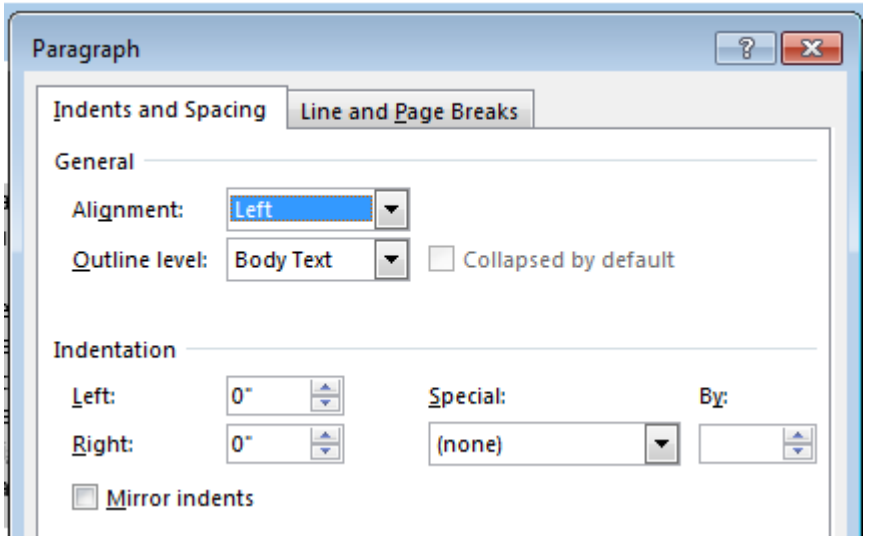
Card#:___-_____-_____
(1234-5678-9012)

It should be possible for the user to choose the type of input (it prevents errors) or adapt to the context

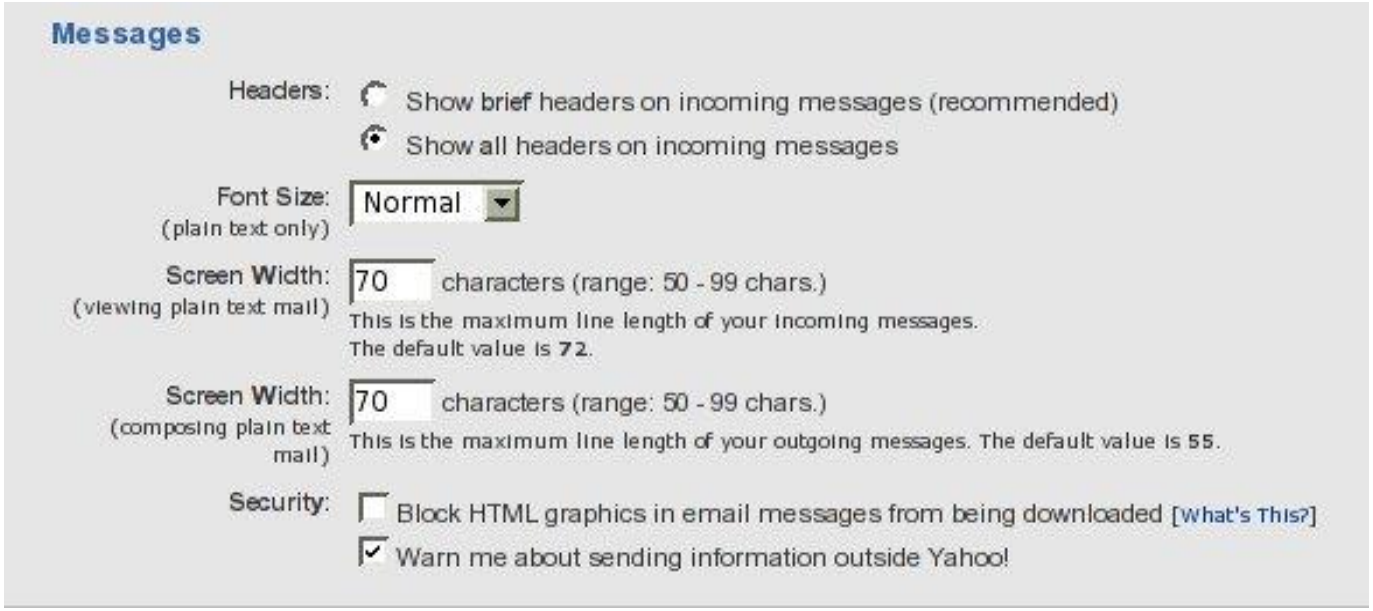


Portuguese version (cm):

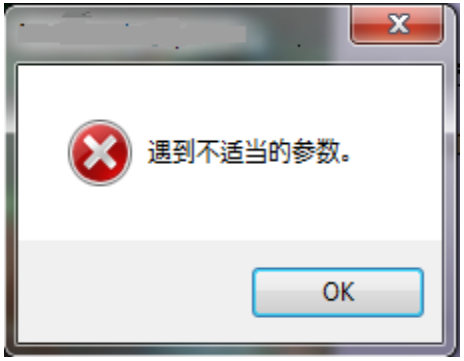
English version (inches):

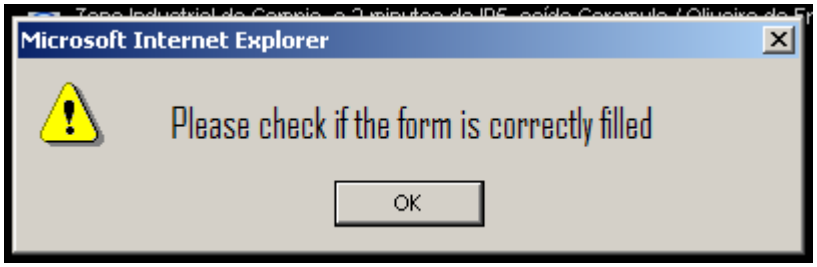


Instructions to fill the fields should be clear as well as messages

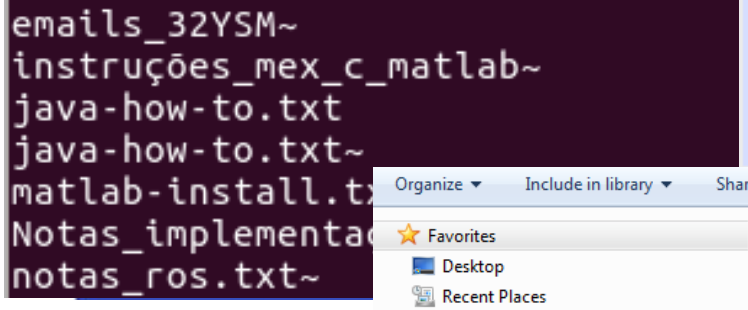


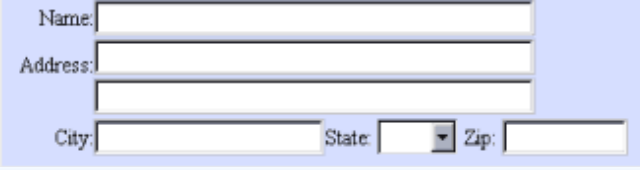
These messages are not much helpful…

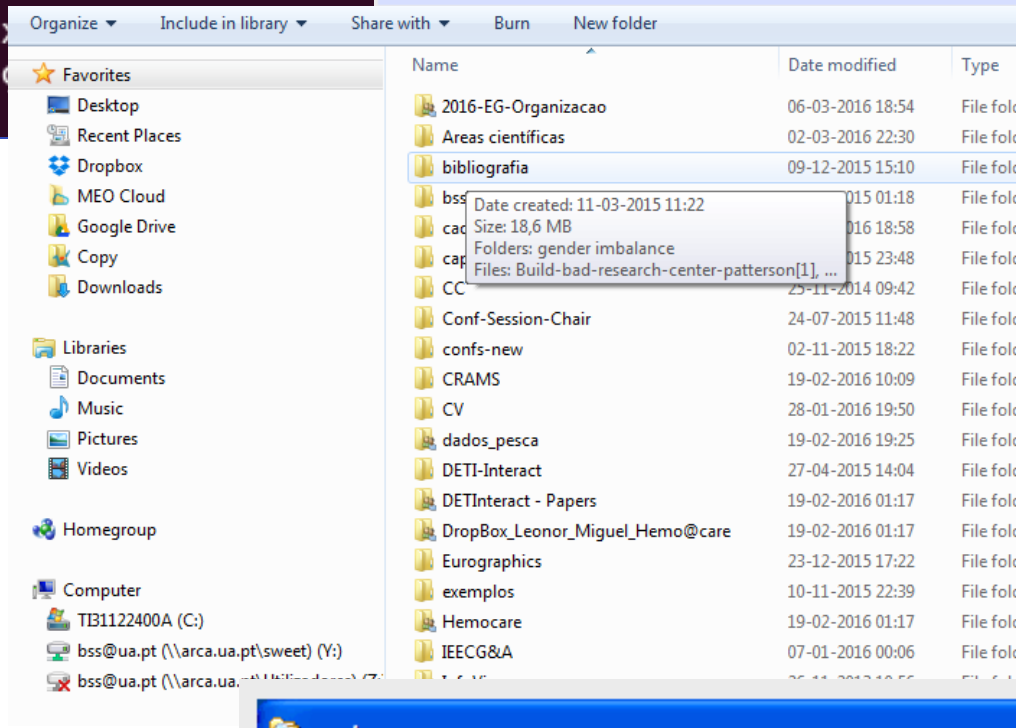




Interaction/ Dialog styles



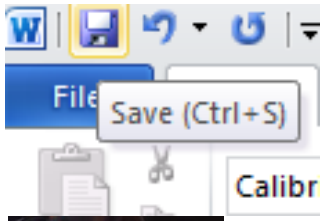


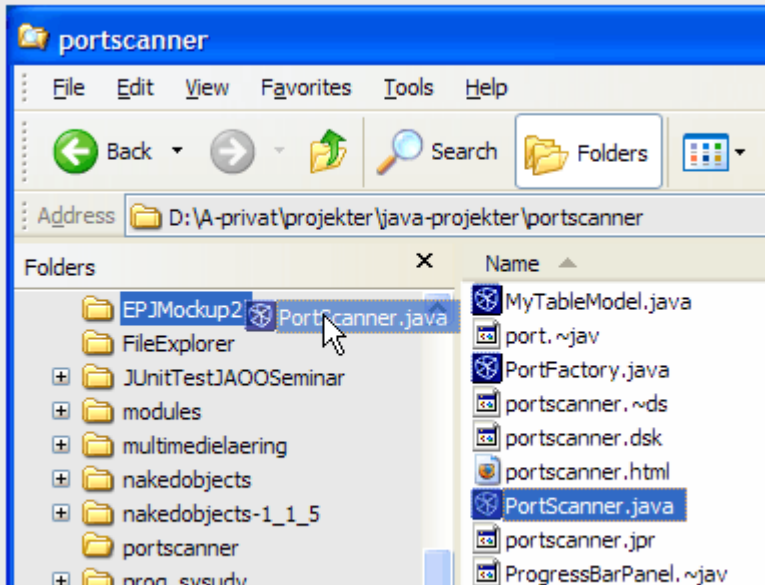


A possible classification:

- Menus
- Direct manipulation
- Fill-in-forms
- Function keys
- Question and answer
- Command languages
- Natural languages

Often two or more styles are used simultaneously





Function keys

- Two types:
  - *Hard Keys* – Always invoke the same functionality (as the keys of a calculator and some specific keys of PCs)
  - *Soft Keys* – invoke different functionality according the context of use (as the keys (F1...Fn) and the generic keys of an Automated Telling Machine, e.g. Multibanco)

- PCs have 12 generic Keys (F1 a F12) and a few other specific keys

Keys that invoke specific functionality in PCs and MACs

Soft Keys

Soft function keys don’t have abbreviations of default actions printed on/besides them, they may have "F-number" designations.

Function keys (generic)

https://en.wikipedia.org/wiki/Function_key

Main advantages and disadvantages

Advantages (potential)

- Self-explanatory
- Recognition instead of recall
- Easy to use
- Flexible
- Require little or no screen real estate

Disadvantages

- Limited number of keys
- Hardware expansions are expensive

Function keys design: guidelines

Provide enough keys to call the functionality

But no too many as not to make it difficult to learn

Use:

- free space
- different size, color and shape to different groups
- category groups
- clear and distinctive names

TV remote control

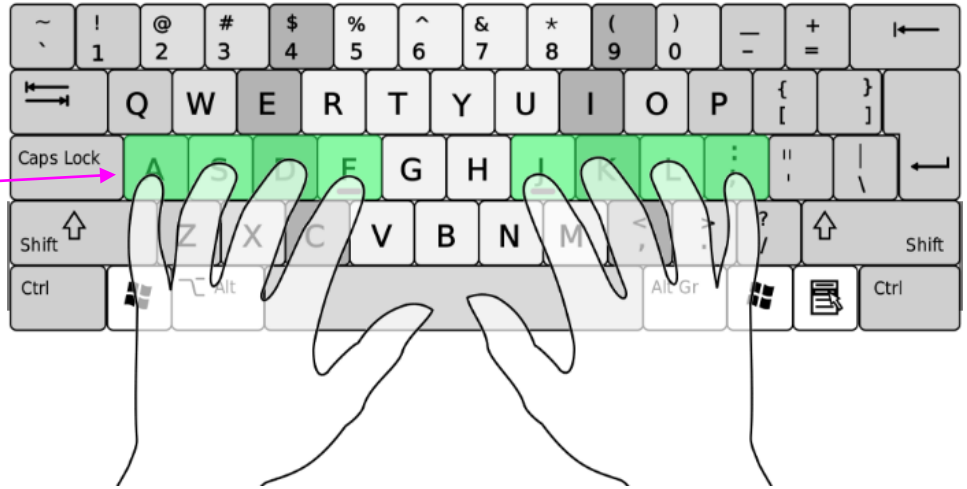
Multi-media remote control keyboard

Industrial keyboard

ATM keyboard

Shop system keyboard

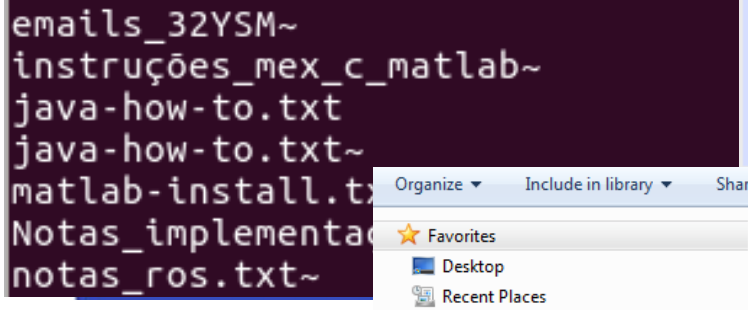
Frequently used keys should be near the “home row”

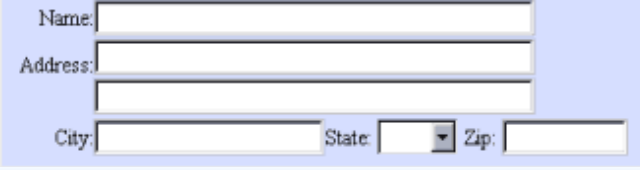


Keys with serious consequences should not be easy to activate

(e.g. ctrl Alt Del)

Interaction/ Dialog styles



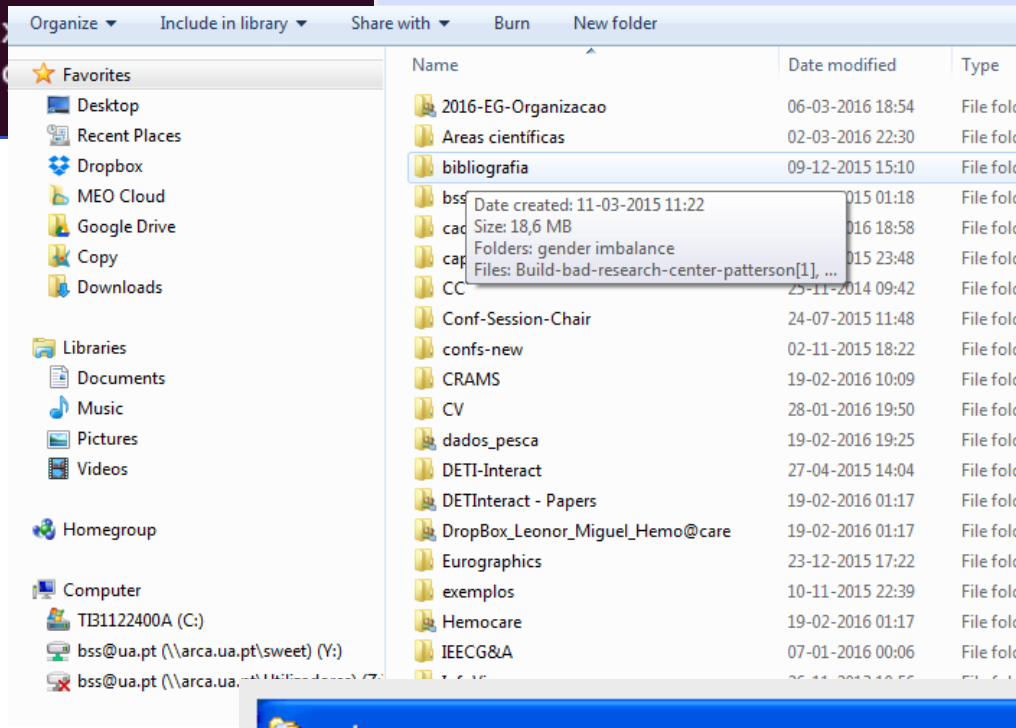


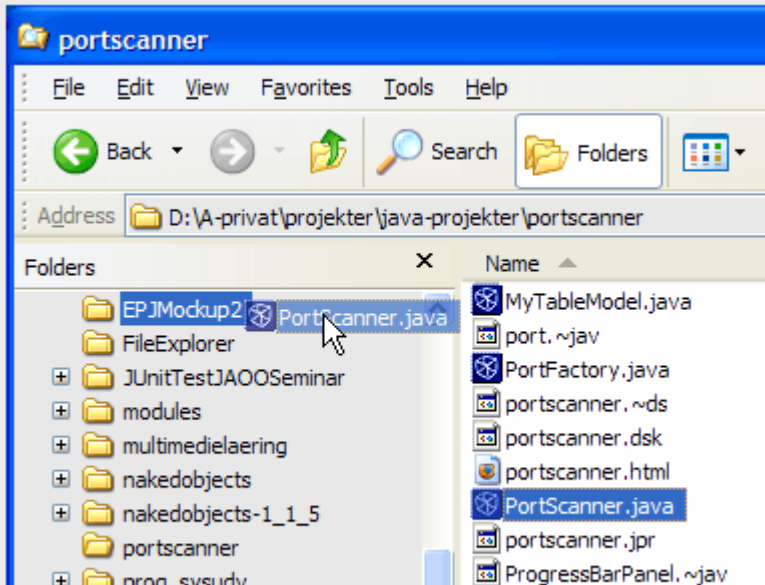
A possible classification:

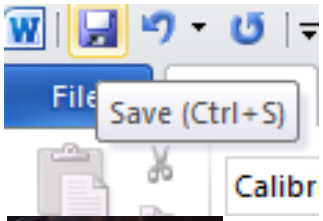
- Menus
- Direct manipulation
- Fill-in-forms
- Function keys
- Question and answer
- Command languages
- Natural languages

…

Often two or more styles are used simultaneously







Command languages

```
cd /tmp
echo "line 1
line 2
line 4" > tmp1$$
echo "line 2
line 3" > tmp2$$
diff tmp1$$ tmp2$$
rm tmp1$$ tmp2$$
```

```
guru99@VirtualBox:~$ history
    1  cat > sample
    2  cat sample
    3  cat sample ^a
    4  cat sample a
    5  cat sample | grep a
    6  cat sample | grep ^a
    7  useradd home
    8  useradd mycomputer
    9  sudo useradd mycomputer
   10  sudo adduser MyLinux
   11  sudo adduser mylinux
   12  vi scriptsample.sh
```

Command languages shall also be **designed as to be as usable as possible**

Basic Goals of Language Design

- Precision
- Compactness
- Ease in writing and reading
- Speed in learning
- Simplicity to reduce errors
- Ease of retention over time

Usability Questions concerning a command language

- Does the language support necessary functions?
- Is it fast to enter a command?
- Is it easy to recognize what the command might do?
- Is it easy to recall a command?
- Are there few errors when using the language?

Main advantages and disadvantages

Advantages (potential)

- Powerful
- Flexible
- Efficient
- Do not take much screen real estate

Disadvantages

- Difficult to learn
- Not self-explainable
- Error prone
- Improvements are not visible

Note that:

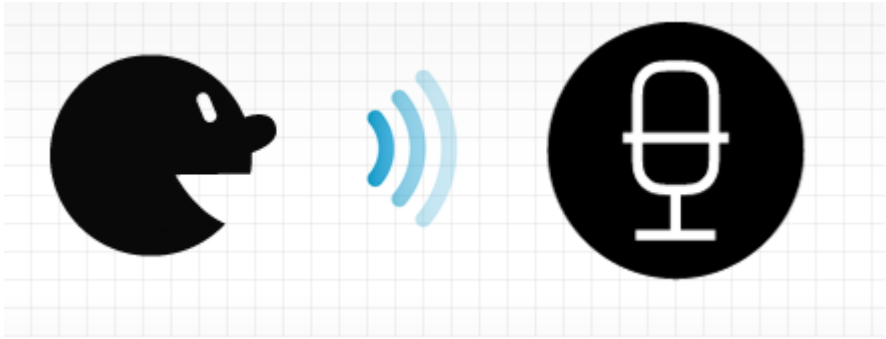
Command languages may be used

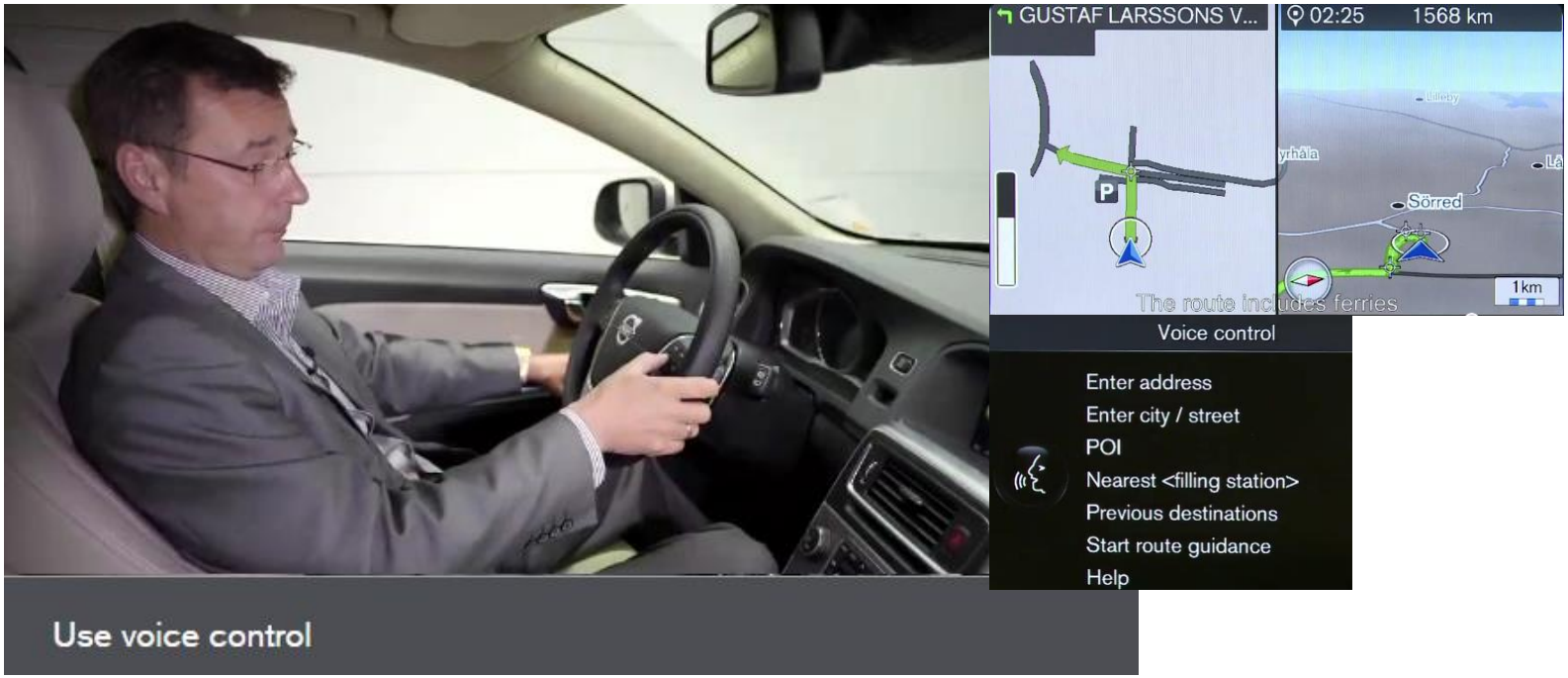
not only through text but also via voice

But they must be very simple …

e.g.

While driving a car to control the media, the phone or navigate





Relevant issues in Command Language design

- Semantics
- Syntax
- Lexicon
- Interaction

Command Languages Design guidelines

Balance richness and minimalism
(similar to semantic distance in direct manipulation)

Examples :

Rich	**Minimal**
Delete	Delete
Insert	Insert
Replace	
Copy	Copy
Move	Delete
Rename	
Delete	

(the functionality is the same)

Use a coherent syntaxe

Use a natural and easy to remember action-object grammar

```
VolB!FileA!D$$
FileA!VolB!ER$L!:KO:!*$$
```

Uncoherent syntax and unfamiliar commands

```
search filea volb.
open filea volb.
list all lines with “KO”.
```

or

```
s filea volb.
o filea volb.
lal “KO”.
```

Command abbreviations should be simple and coherent

Easy to remember (not easy to recognize as for function keys)

Name	Abbreviations	
	Poor:	Improved:
Move forward	MovF	MovF
Move backward	Mvb	MovB
Insert	I	Ins
Delete	Dl	Del
Replace	Repl	Rep
Search	Srch	Sea
Delete	X	Del
Send	Sn	Sen
Print	Prt	Pri
Search	Srch	Sea
Send	Sn	Sen
Find	Fi	Fin
Choose	Ch	Cho

Allow the following interaction features:

- Defaults
- Command edition
- Intelligent interpretation
- Type-ahead
- Feedback
- Help and documentation
- Make the language “user tailorable”

Example of intelligent interpretation:
“delate”: did you mean “delete”? Y or N

Example of a (complex) command with defaults

ls - Linux man page

Name

ls - list directory contents

Synopsis

ls [*OPTION*]... [*FILE*]...

Description

List information about the FILEs (the current directory by default). Sort entries alphabetically if none of **-cftuvSUX** nor **--sort**.

Mandatory arguments to long options are mandatory for short options too.

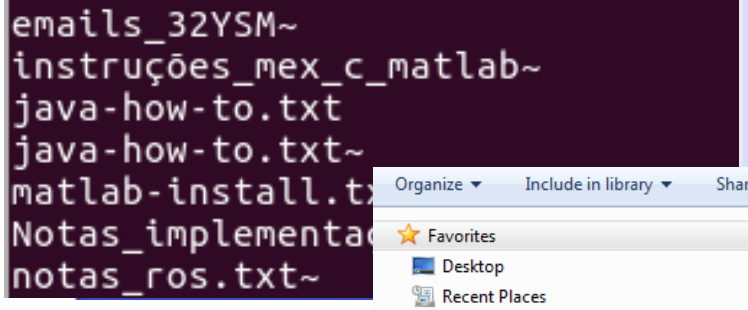
-a, **--all**
do not ignore entries starting with .
-A, **--almost-all**
do not list implied . and ..
--author
with **-l**, print the author of each file
-b, **--escape**
print octal escapes for nongraphic characters

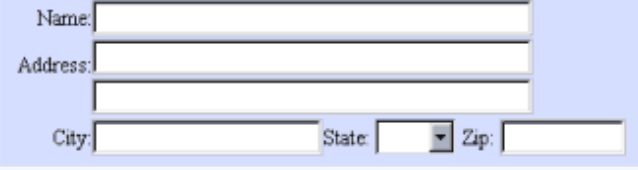
You don't need to use all arguments; there are default values

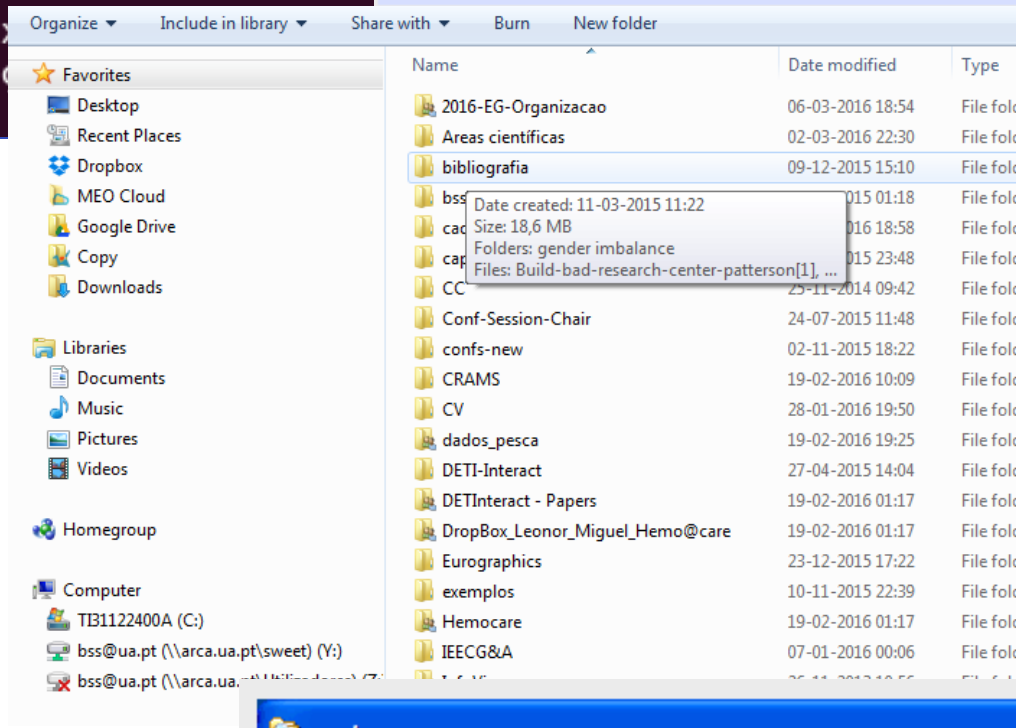
-d, **--directory**
list directory entries instead of contents, and do not dereference symbo
-D, **--dired**
generate output designed for Emacs' dired mode
-f
do not sort, enable **-aU**, disable **-ls --color**
-F, **--classify**
append indicator (one of */=>@|) to entries
--file-type
likewise, except do not append '*'
--format=*WORD*
across **-x**, commas **-m**, horizontal **-x**, long **-l**, single-column **-1**, verbose
--full-time
like **-l --time-style**=*full-iso*
-g
like **-l**, but do not list owner
--group-directories-first
group directories before files.
augment with a **--sort** option, but any
use of **--sort**=*none* (**-U**) disables grouping
-G, **--no-group**
in a long listing, don't print group names
-h, **--human-readable**
with **-l**, print sizes in human readable format (e.g., 1K 234M 2G)
--si
likewise, but use powers of 1000 not 1024
-H, **--dereference-command-line**
follow symbolic links listed on the command line

Etc.., etc., etc.

Interaction/ Dialog styles



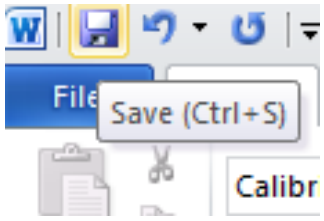


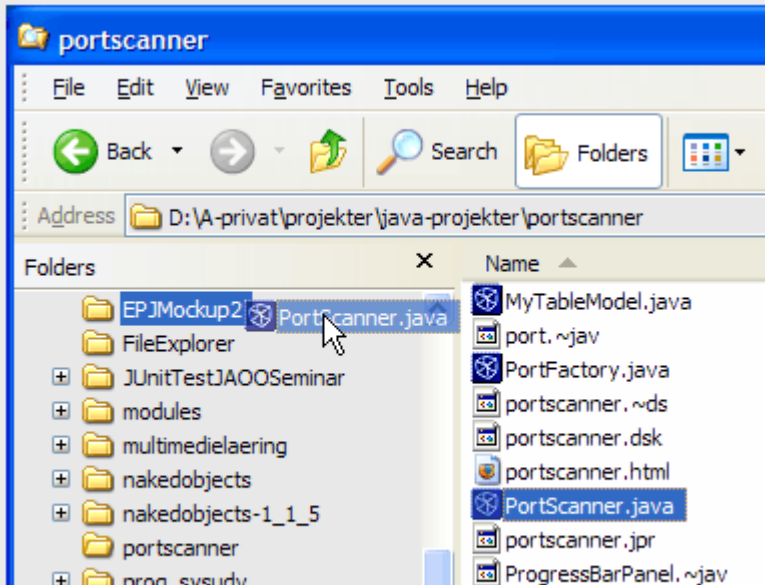


A possible classification:

- Menus
- Direct manipulation
- Fill-in-forms
- Function keys
- Question and answer
- Command languages
- Natural languages

Often two or more styles are used simultaneously





Natural language

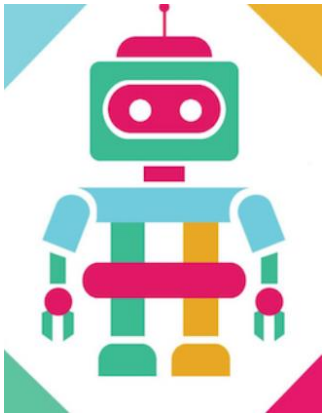
- Communication between humans and computers through natural language involves:
  - recognition
  - generation
- Natural language processing (NLP) has been evolving a lot …

Note:
natural language as a interaction style and voice interaction are different things!

Conversational User interfaces (CUIs)

Based on natural language

Think of the potential advantages and disadvantages of CUIs:



- Chatbots

https://www.nngroup.com/articles/chatbots/

- Voice assistants

"Just like the touch interface, not everything will become conversational"

What doesn't fit the principles of Conversational UI well?
Products where the use case involves a technical user who wants fine grain control over the interface, …. e.g. CAD software, or a programming IDE…."

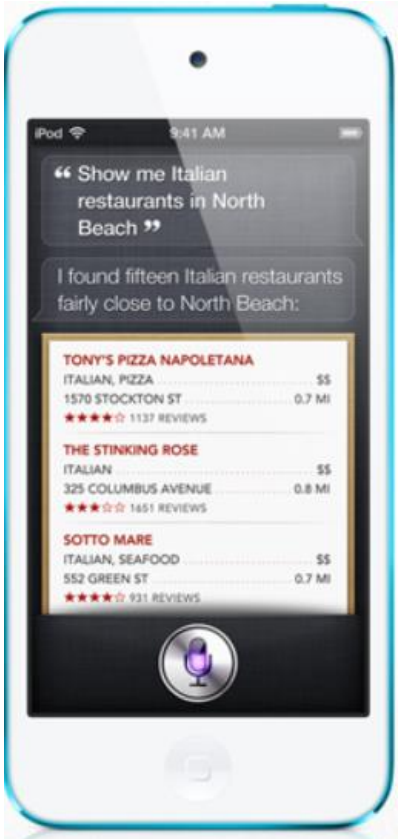
https://uxdesign.cc/conversational-ui-its-not-just-chat-bots-and-voice-assistants-case-study-cb1865da306a

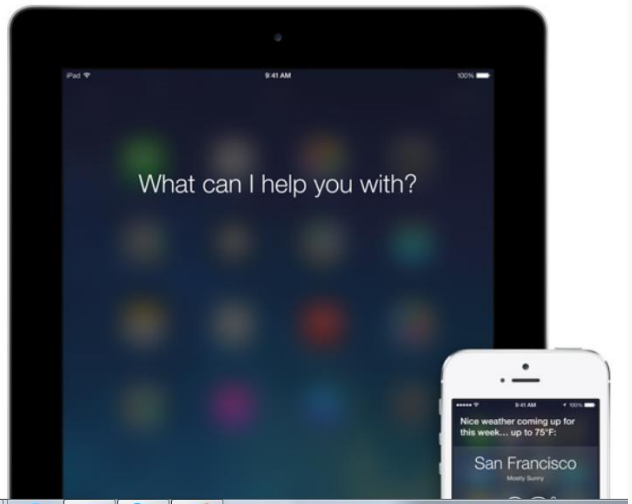
Current examples of Natural language interaction (mostly via voice)

Mobile phone personal assistants:

- Siri for Apple's iOS

- Google assistant

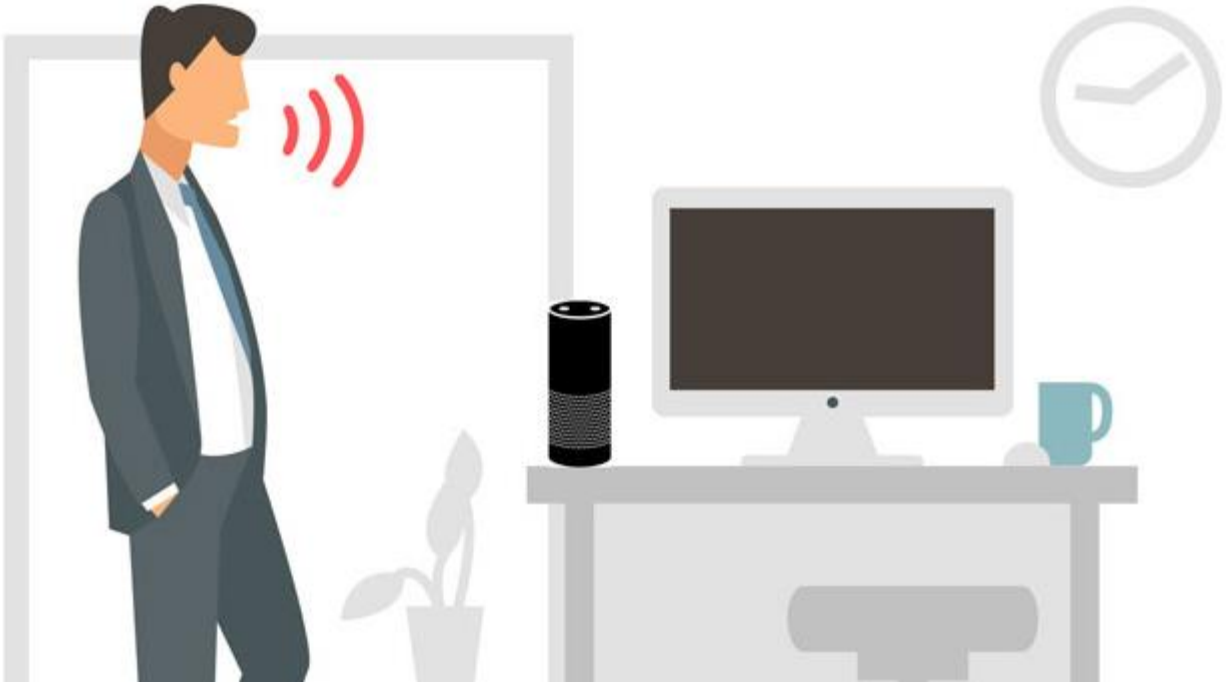




Another example
(natural language via voice)

amazon echo

- Amazon Alexa



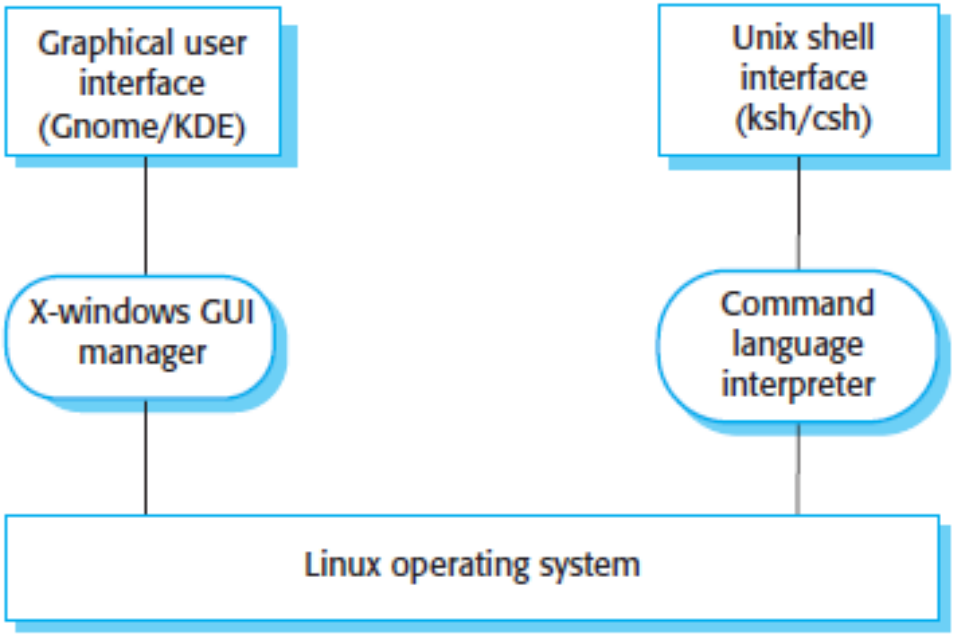
https://www.nngroup.com/articles/voice-interaction-ux/

Main advantages and disadvantages of interaction styles

Interaction style	Main advantages	Main disadvantages	Application examples
Direct manipulation	Fast and intuitive interaction Easy to learn	May be hard to implement Only suitable where there is a visual metaphor for tasks and objects	Video games CAD systems
Menu selection	Avoids user error Little typing required	Slow for experienced users Can become complex if many menu options	Most general-purpose systems
Form fill-in	Simple data entry Easy to learn Checkable	Takes up a lot of screen space Causes problems where user options do not match the form fields	Stock control Personal loan processing
Command language	Powerful and flexible	Hard to learn Poor error management	Operating systems Command and control systems
Natural language	Accessible to casual users Easily extended	Requires more typing Natural language understanding systems are unreliable	Information retrieval systems

(Sommerville, 2010, chap.29)

Multiple user interfaces example



(Sommerville, 2010, chap.29)

3D User Interfaces

- User interfaces involving 3D interaction (i.e. interaction in which the user's tasks are performed directly in a 3D spatial context).

- Are more and more used:

  - Virtual and augmented reality
  - 3D workspaces
  - Data Visualization …

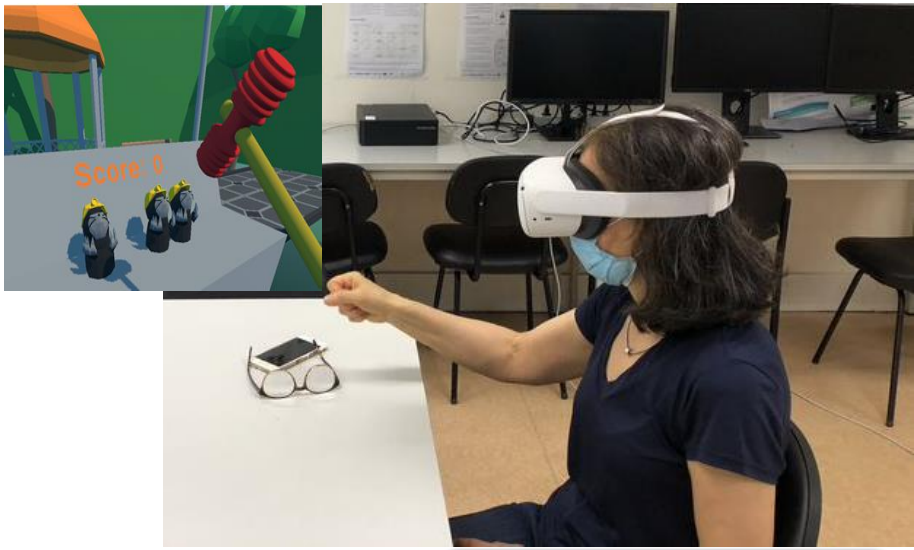
- But have some issues:
  - User disorientation

  (in the real world we have more information)

Applications of virtual and augmented reality:

- Training and simulation
- Assistance in tasks
- Project review
- Therapy
- Experiments
- Entertainment

…



Main bibliography

- Soegaard, Mads. Interaction Styles

  http://www.interactiondesign.org/encyclopedia/interaction_styles.html

  https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/3d-user-interfaces

- Ian Sommerville, Software Engineering, 9 ed, Addison Wesley , 2010
  https://ifs.host.cs.st-andrews.ac.uk/Books/SE9/WebChapters/PDF/Ch_29%20Interaction_design.pdf