

# Ten usability heuristics

---

Jakob Nielsen

# Ten usability heuristics

1. Match between system and the real world
2. Consistency and standards
3. Visibility of system status
4. User control and freedom
5. Error prevention
6. Help users recognize, diagnose, and recover from errors
7. Recognition rather than recall
8. Flexibility and efficiency of use
9. Aesthetic and minimalist design
10. Help and documentation

# #1: 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.*

# #1: Match between system and the real world

Computer science language: `[0][0]`, `[0][2]`

Excel speaks users' language: `[A][1]`, `[B][3]`

	A	B	C	D	E	F	G
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							

# #1: Match between system and the real world



## #2: Consistency and standards

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

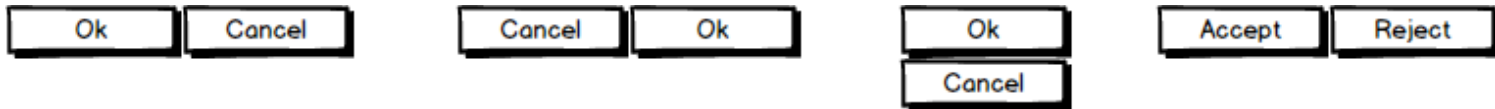
# #2: Consistency and standards

## Principle of least surprise

- Similar things should look and act similar
- Different things should look different
- Users should not have to wonder whether different words, situations, or actions mean the same thing (follow platform conventions)

## Consistent language and graphics

- Same visual appearance across the system
- Same information/controls in same location on all windows



## Consistent effects

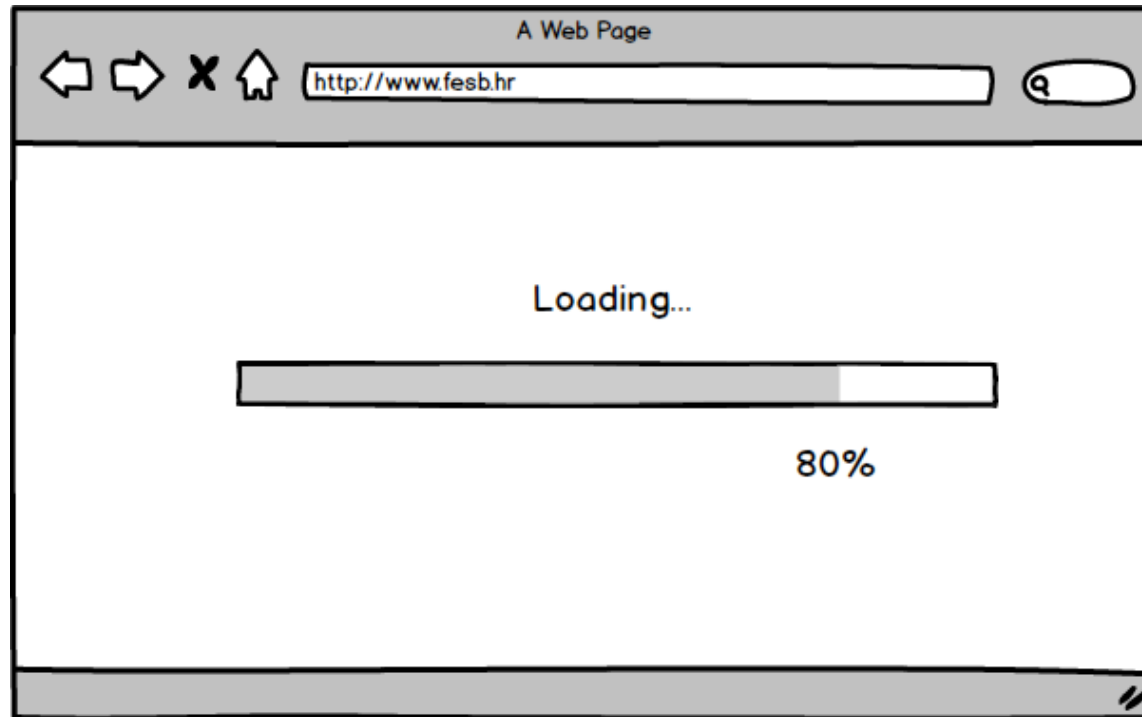
- Commands, actions have same effect in equivalent situations

### #3: Visibility of system status

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



# #3: Visibility of system status



# #3: Visibility of system status

## Response times - important limits

- **< 0.1 second** - seems instantaneous
- **0.1-1 seconds** - user notices, but no feedback needed
- **1-5 seconds** - display busy cursor
- **> 1-5 seconds** - display progress bar

*Response Times: The 3 Important Limits, [J. Nielsen'14]*

## #4: 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.*

# #4: User control and freedom

Users don't like to feel trapped by the computer

- Should offer an easy way out of as many situations as possible
- Provide clearly marked exits

Strategies:

- Cancel button
- Universal undo
- Interrupt (especially for lengthy operations)
- Quit (for leaving the program at any time)
- Defaults (for restoring default properties)

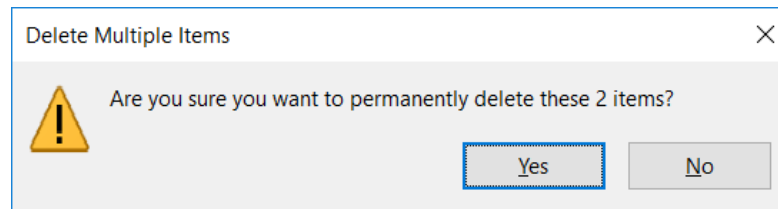
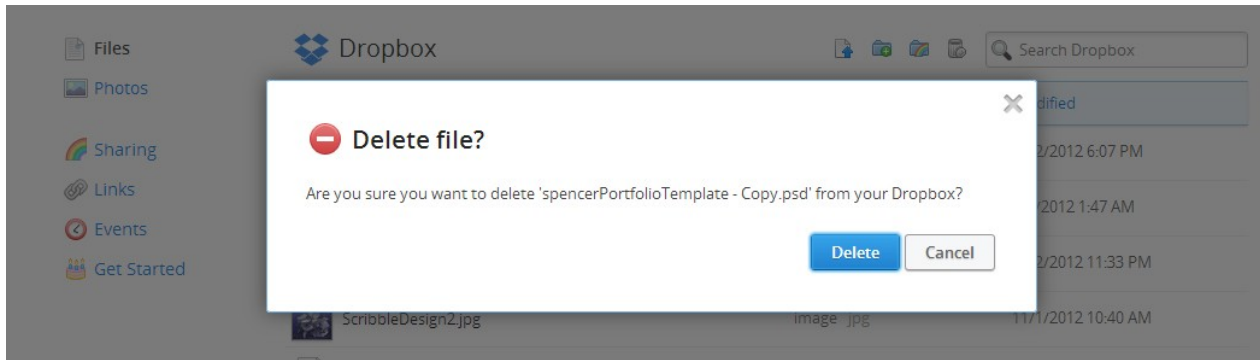
## #5: Error prevention

*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.*

# #5: Error prevention

## Design for errors

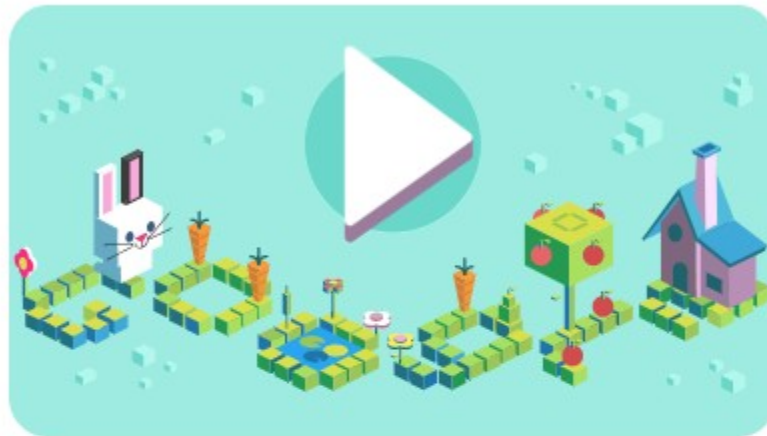
- Double-check with users



# #5: Error prevention

## Design for errors

- Remove memory burdens



nilsen heu|

**nielsen** heuristics  
heuristicsas nilsen  
**dennis** nilsen heute  
**kurt** nilsen heute

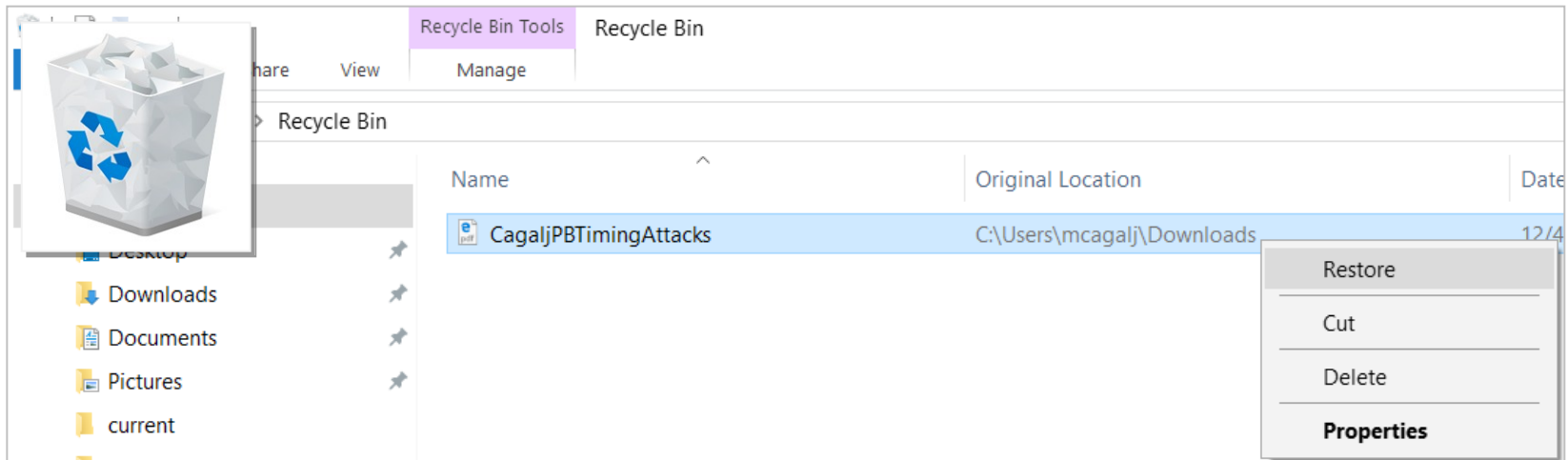
Google Search I'm Feeling Lucky

*Report inappropriate predictions*

# #5: Error prevention

## Design for errors

- Support Undo and Re 



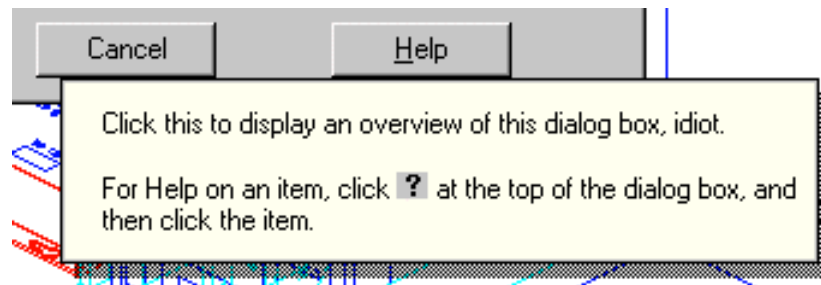
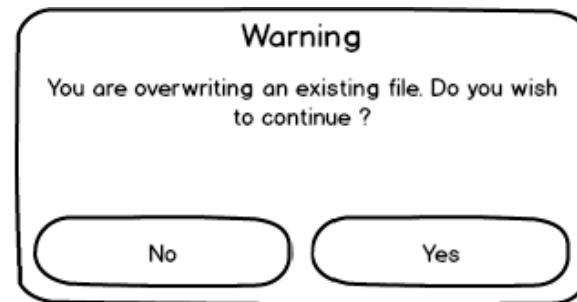
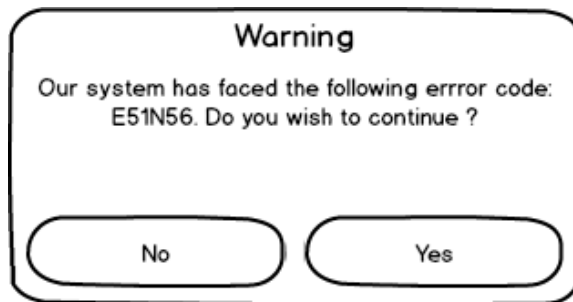


## #6: Help users recognize, diagnose, and recover from errors

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

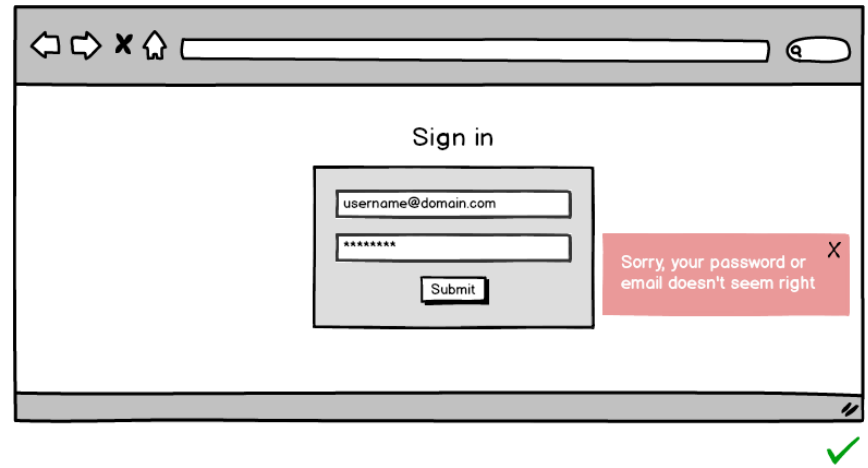
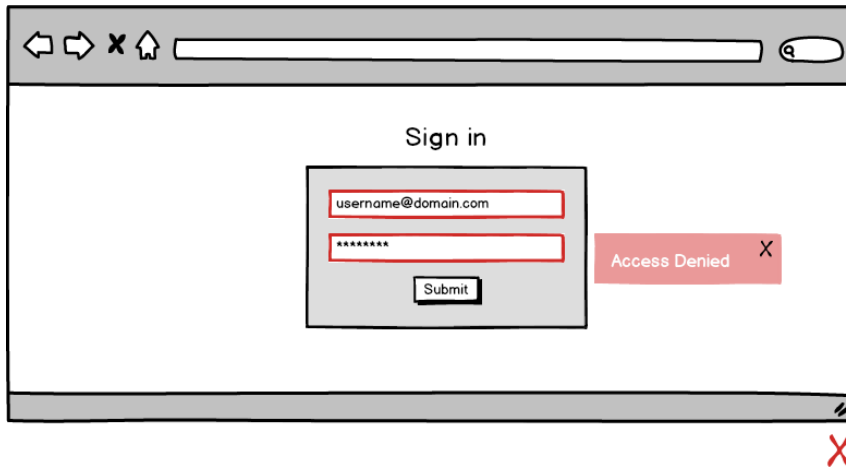
# #6: Help users recognize, diagnose, and recover from errors

Deal with errors in a positive manner (be polite, speak human-readable language)



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

Deal with errors in a positive manner (be polite, speak human-readable language)



## #7: Recognition rather than recall

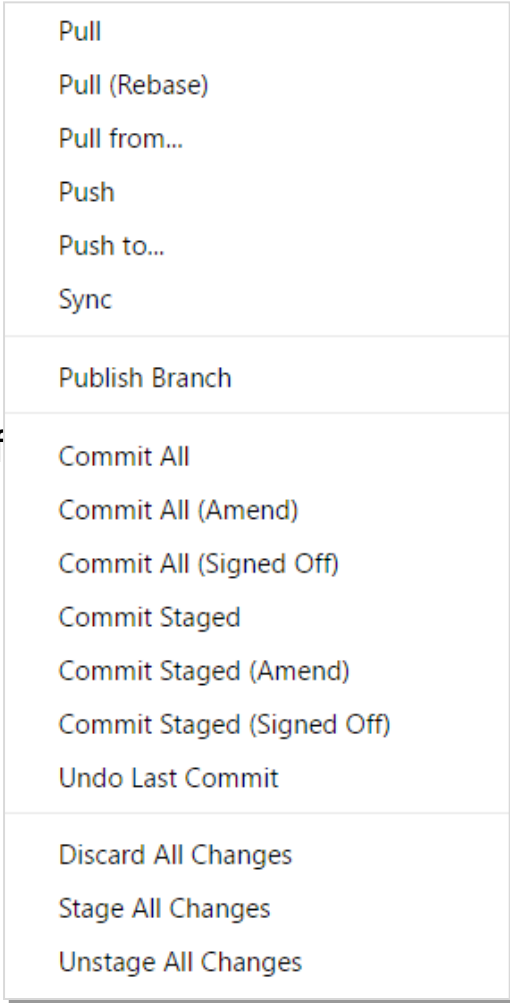
*Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.*

# #7: Recognition rather than recall

Short-term memory can hold around 5-7 elements

- CLI vs GUI

```
C:\> git add -A -- c:\Users\mcagalj\...\index.js
C:\> git commit --quiet --allow-empty-message --f
C:\> git status -z -u
C:\> git symbolic-ref --short HEAD
C:\> git remote --verbose
C:\> git config --get commit.template
C:\> git push
```



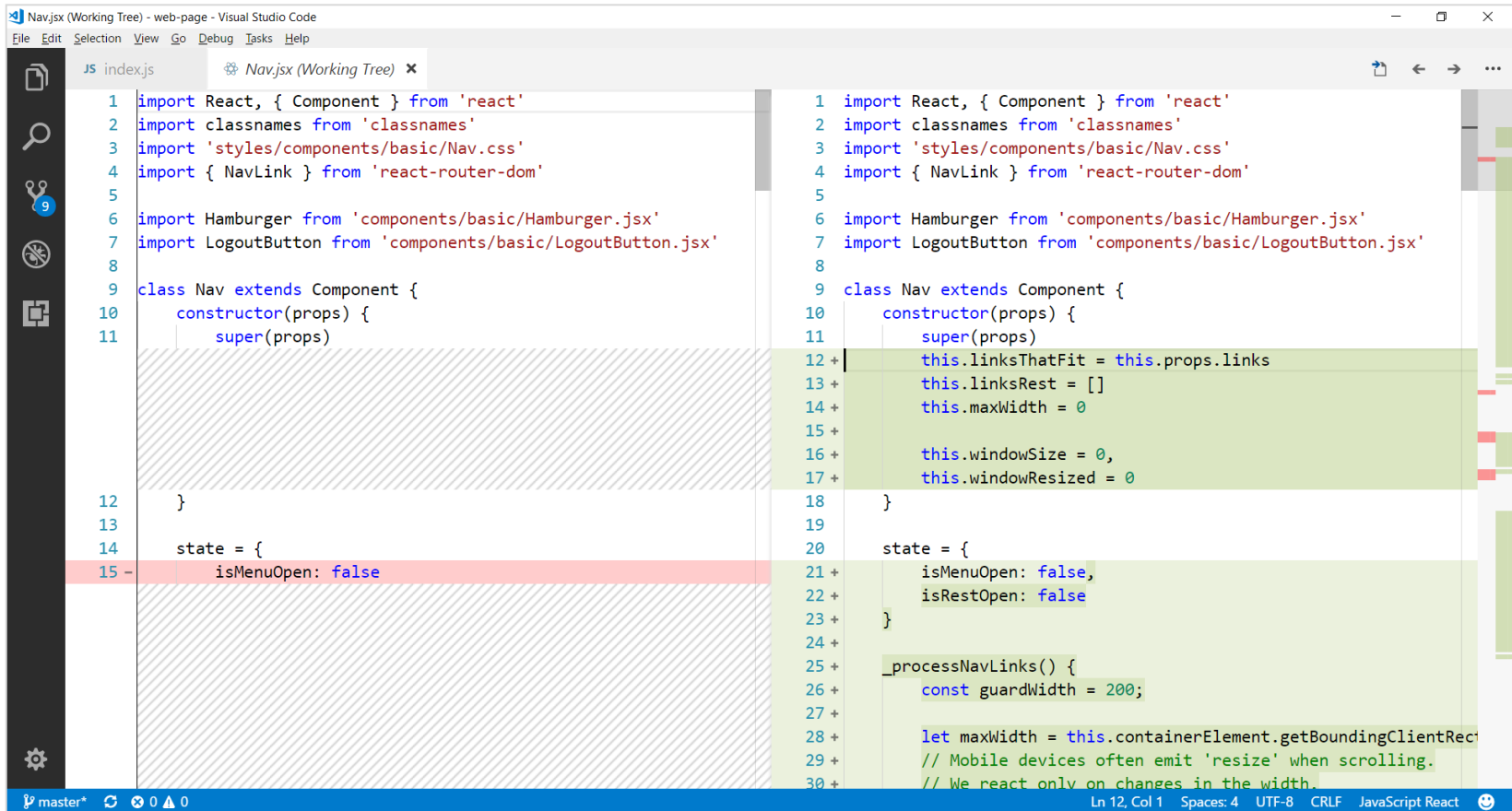
Pull  
Pull (Rebase)  
Pull from...  
Push  
Push to...  
Sync

Publish Branch

Commit All  
Commit All (Amend)  
Commit All (Signed Off)  
Commit Staged  
Commit Staged (Amend)  
Commit Staged (Signed Off)  
Undo Last Commit

Discard All Changes  
Stage All Changes  
Unstage All Changes

# #7: Recognition rather than recall

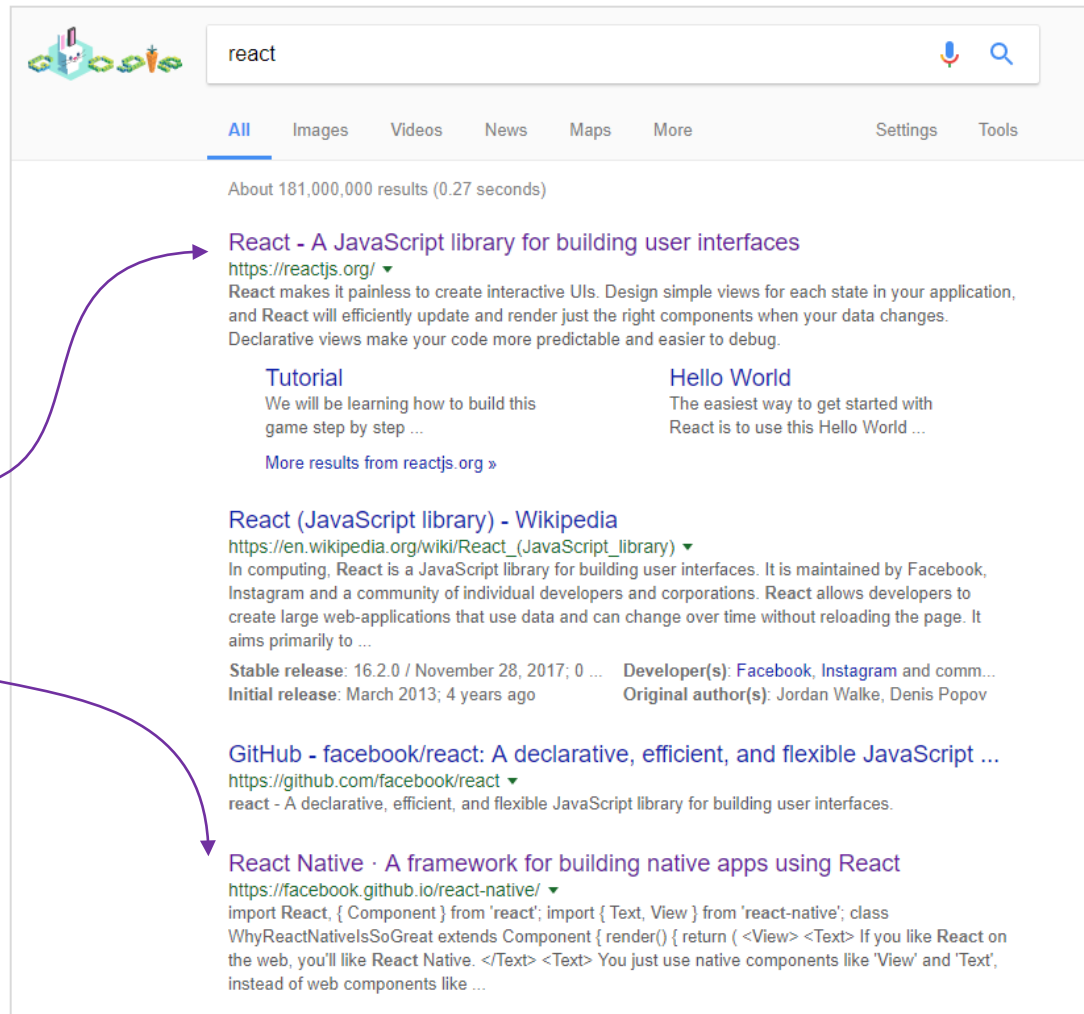


```
1 import React, { Component } from 'react'
2 import classNames from 'classnames'
3 import 'styles/components/basic/Nav.css'
4 import { NavLink } from 'react-router-dom'
5
6 import Hamburger from 'components/basic/Hamburger.jsx'
7 import LogoutButton from 'components/basic/LogoutButton.jsx'
8
9 class Nav extends Component {
10   constructor(props) {
11     super(props)
12   }
13
14   state = {
15     isMenuOpen: false
16   }
17
18   constructor(props) {
19     super(props)
20     this.linksThatFit = this.props.links
21     this.linksRest = []
22     this.maxWidth = 0
23
24     this.windowSize = 0,
25     this.windowResized = 0
26   }
27
28   state = {
29     isMenuOpen: false,
30     isRestOpen: false
31   }
32
33   _processNavLinks() {
34     const guardWidth = 200;
35
36     let maxWidth = this.containerElement.getBoundingClientRec
37     // Mobile devices often emit 'resize' when scrolling.
38     // We react only on changes in the width.
```

Ln 12, Col 1 Spaces: 4 UTF-8 CRLF JavaScript React

# #7: Recognition rather than recall

Visited links



The image shows a Google search interface for the term "react". The search bar at the top contains the text "react". Below the search bar, there are tabs for "All", "Images", "Videos", "News", "Maps", and "More". The "All" tab is selected. The search results show "About 181,000,000 results (0.27 seconds)".

The first result is "React - A JavaScript library for building user interfaces" with the URL <https://reactjs.org/>. Below this title, there is a description: "React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes. Declarative views make your code more predictable and easier to debug." There are two sub-links: "Tutorial" (with the text "We will be learning how to build this game step by step ...") and "Hello World" (with the text "The easiest way to get started with React is to use this Hello World ..."). A link "More results from reactjs.org »" is also present.

The second result is "React (JavaScript library) - Wikipedia" with the URL [https://en.wikipedia.org/wiki/React\\_\(JavaScript\\_library\)](https://en.wikipedia.org/wiki/React_(JavaScript_library)). The description states: "In computing, React is a JavaScript library for building user interfaces. It is maintained by Facebook, Instagram and a community of individual developers and corporations. React allows developers to create large web-applications that use data and can change over time without reloading the page. It aims primarily to ...". Below this, it lists "Stable release: 16.2.0 / November 28, 2017; 0 ..." and "Developer(s): Facebook, Instagram and comm...". It also lists "Initial release: March 2013; 4 years ago" and "Original author(s): Jordan Walke, Denis Popov".

The third result is "GitHub - facebook/react: A declarative, efficient, and flexible JavaScript ..." with the URL <https://github.com/facebook/react>. The description says: "react - A declarative, efficient, and flexible JavaScript library for building user interfaces."

The fourth result is "React Native · A framework for building native apps using React" with the URL <https://facebook.github.io/react-native/>. The description includes a code snippet: `import React, { Component } from 'react'; import { Text, View } from 'react-native'; class WhyReactNativeIsSoGreat extends Component { render() { return ( <View> <Text> If you like React on the web, you'll like React Native. </Text> <Text> You just use native components like 'View' and 'Text', instead of web components like ...`

Three purple arrows originate from the text "Visited links" on the left. The first arrow points to the "React - A JavaScript library for building user interfaces" result. The second arrow points to the "React (JavaScript library) - Wikipedia" result. The third arrow points to the "React Native · A framework for building native apps using React" result.

# #7: Recognition rather than recall



recognition rath|



recognition rather **than recall**  
recognition rather **than recall** definition  
recognition rather **than recall** meaning  
recognition rather **than recall** website  
recognition rather **than recall** principle  
**6.** recognition rather **than recall**  
**contoh** recognition rather **than recall**

Google Search

I'm Feeling Lucky

[Report inappropriate predictions](#)



## #8: 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.

# #8: Flexibility and efficiency of use

Provide easily-learned shortcuts for frequent operations

- Keyboard accelerators
- Command abbreviations
- Bookmarks
- History (command line interfaces)
- Templates e.g. ppt

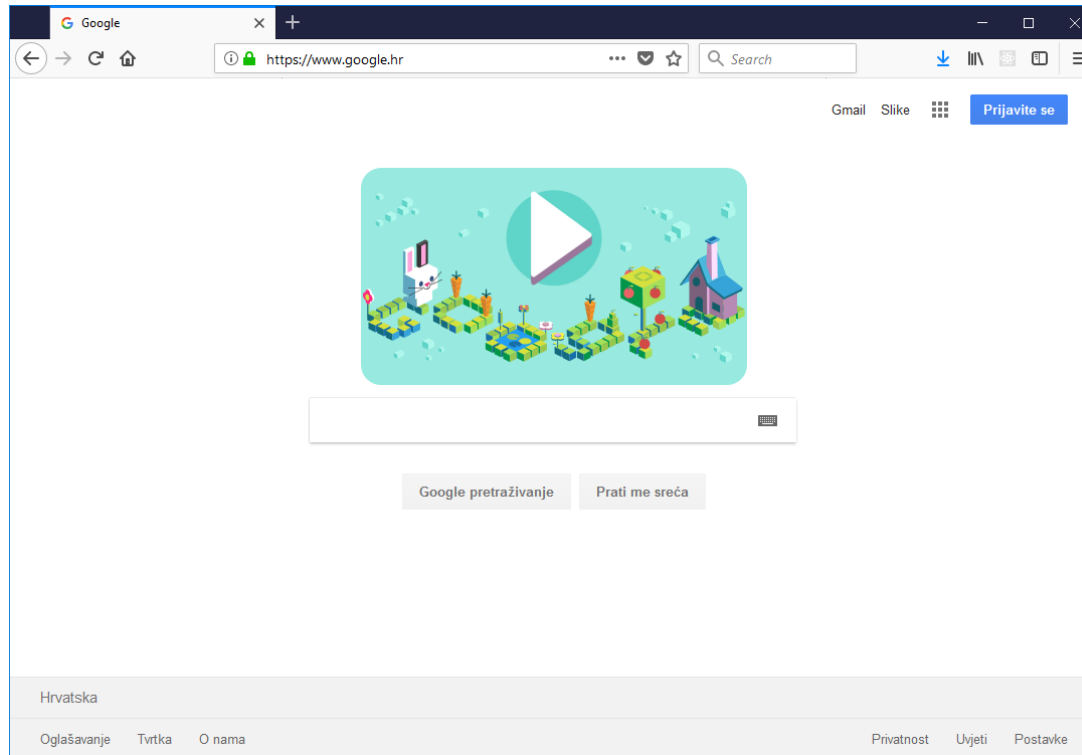
## #9: Aesthetic and minimalist design

*Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.*

# #9: Aesthetic and minimalist design

## Less is more

- Omit extraneous info, graphics, features
- Provide only relevant data



## #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

Users don't read manuals 😊

But manuals and online help are vital when user is frustrated or in crisis

Help should be:

- Searchable
- Context-sensitive
- Task-oriented
- Concrete
- Short

# #10: Help and documentation

