

WEB & DIGITAL ACCESSIBILITY

**How do I know if a website is
accessible?**



W3C[®] WCAG 2.1
Web Content
Accessibility
Guidelines

Web Content Accessibility Guidelines

Set of principles, guidelines, and success criteria to build accessible web content

Published by W3C

Web content creators and web developers should adopt them into their practice

- **WCAG 1.0** (1999, now obsolete)
- **WCAG 2.0** (2008, complete rewrite, with new principles and a new structure)
- **WCAG 2.1** (2018, adds new criteria without removing or changing anything from 2.0 guidelines)
- **WCAG 2.2** coming soon (<https://www.w3.org/TR/WCAG22/>)

WCAG 2.x Structure

1. Principles

- **basic qualities** web sites need to have when considering accessibility

2. Guidelines

- allow the design of web sites **staying in the right track when considering accessibility**

3. Success criteria

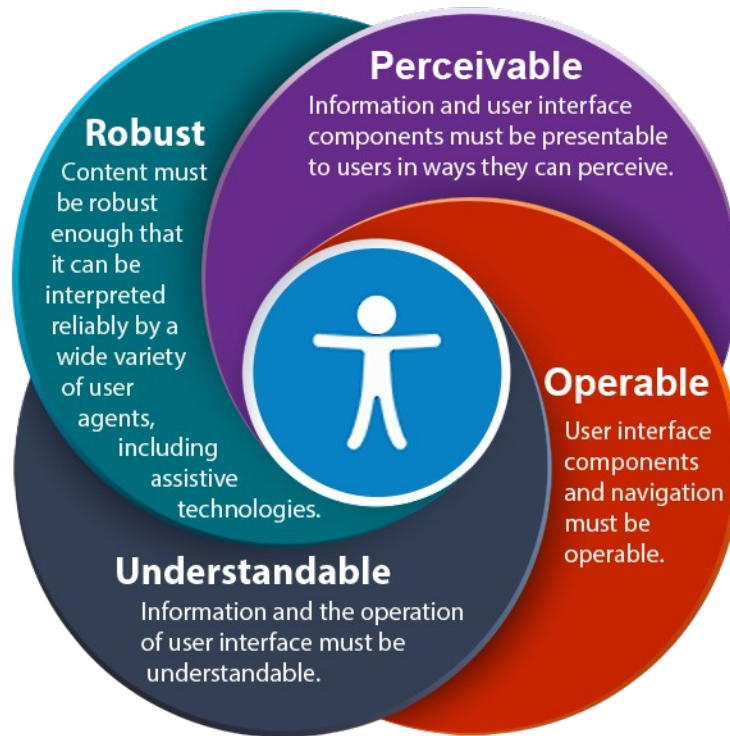
- allow to **verify the work done** to see if principles and guidelines were successfully followed

4. Levels of conformance (testable accessible criteria)

- A, **AA**, AAA

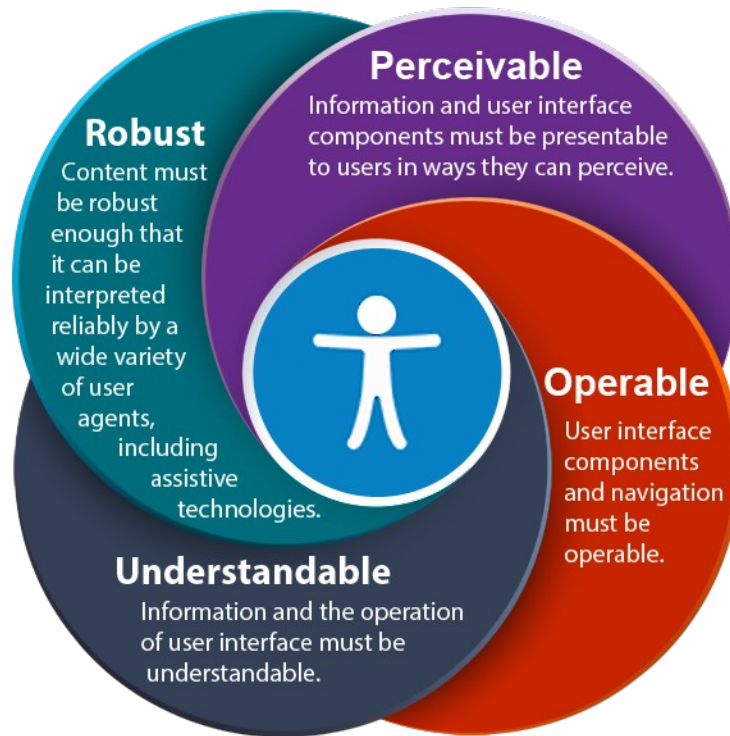
WCAG 2.x Principles (POUR)

Four main principles (technology agnostic)



WCAG 2.x Principles (POUR)

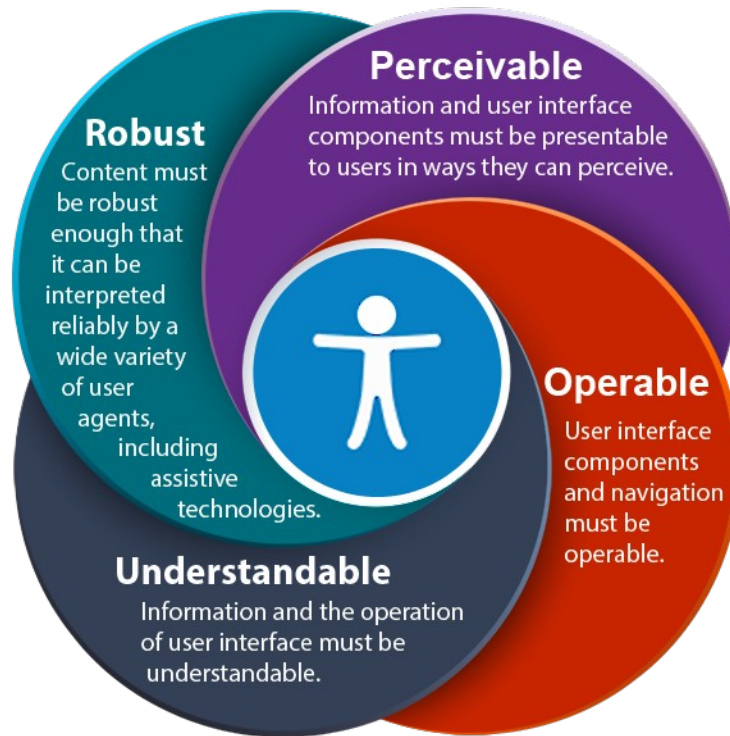
Four main principles (technology agnostic)



For example, make images available with alt-text, videos available through captions and audio descriptions, provide sufficient color contrast between foreground and background

WCAG 2.x Principles (POUR)

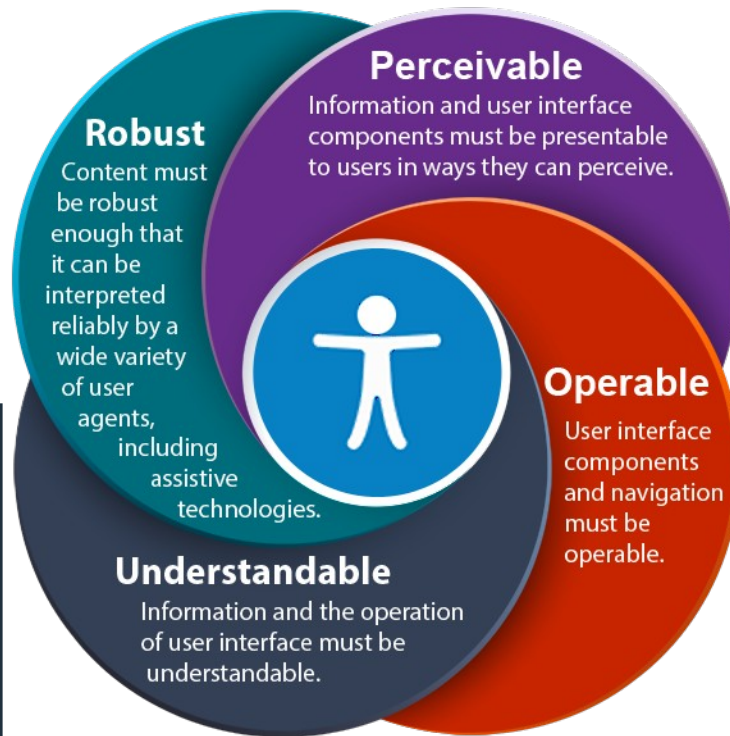
Four main principles (technology agnostic)



For example, enabling navigation using only the keyboard, meaningful links, allowing enough time to complete a task

WCAG 2.x Principles (POUR)

Four main principles (technology agnostic)



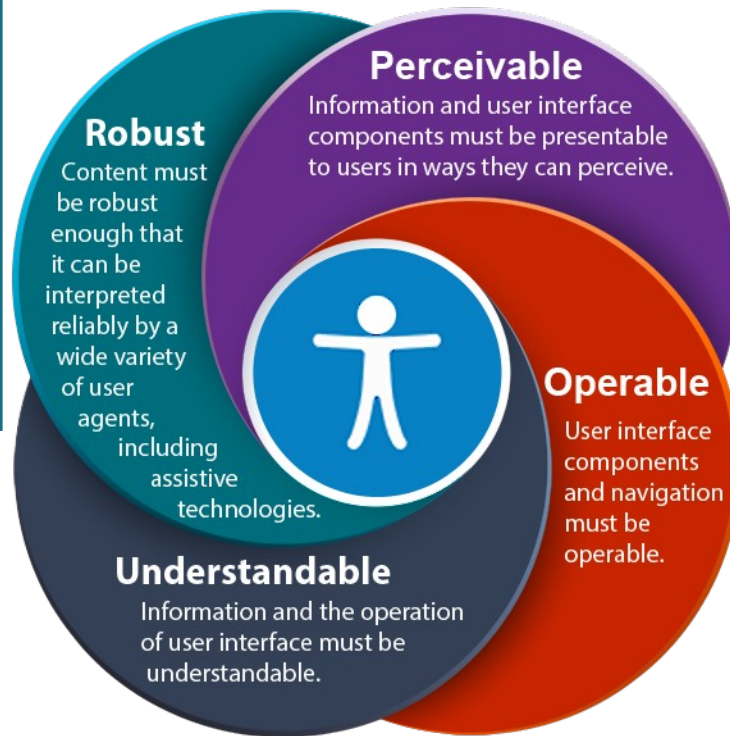
For example, specifying the language of the page and assisting users when filling forms fields, predict or correct mistakes

WCAG 2.x Principles (POUR)

Four main principles (technology agnostic)

Maximizing compatibility with current and future devices

Code with bugs or wrong use of components might have a negative impact on assistive technologies, that cannot interpret correctly the content



WCAG 2.x Guidelines

1. Perceivable has 4 guidelines

- Guideline 1.1 – Text Alternatives
- Guideline 1.2 – Time-based Media
- Guideline 1.3 – Adaptable
- Guideline 1.4 – Distinguishable

See <https://www.w3.org/WAI/WCAG21/quickref/?versions=2.1#principle1>

WCAG 2.x Guidelines

2. Operable has **5** guidelines

- Guideline 2.1 – Keyboard Accessible
- Guideline 2.2 – Enough Time
- Guideline 2.3 – Seizures and Physical Reactions
- Guideline 2.4 – Navigable
- Guideline 2.5 – Input Modalities

See <https://www.w3.org/WAI/WCAG21/quickref/?versions=2.1#principle2>

WCAG 2.x Guidelines

3. Understandable has **3** guidelines

- Guideline 3.1 – Readable
- Guideline 3.2 – Predictable
- Guideline 3.3 – Input Assistance

4. Robust has **1** guideline

- Guideline 4.1 – Compatible

See <https://www.w3.org/WAI/WCAG21/quickref/?versions=2.1#principle3>

See <https://www.w3.org/WAI/WCAG21/quickref/?versions=2.1#principle4>

WCAG 2.x Success criteria

Many success criteria have been defined by W3C, they are very technical, with examples of success and failures

- A, 30
- AA, 20
- AAA, 28

WCAG 2.x Success criteria

Example

Success Criterion 3.2.4 Consistent Identification

(Level AA)

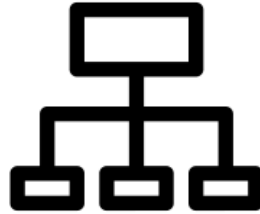
Components that have the same functionality within a set of Web pages are identified consistently.

See <https://www.w3.org/WAI/WCAG21/Understanding/consistent-identification.html>

Most common accessibility issues



Navigation



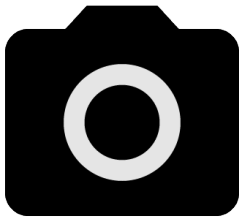
Site structure



Text



Links



Images



Multimedia



Forms

Testing for accessibility


Accessibility should be introduced from the early stages of the design, not at the end

- Determine the **required level of compliance**
- Define and use an **accessibility check list**

Testing for accessibility

Accessibility should be introduced from the early stages of the design, not at the end

- Determine the **required level of compliance**
- Define and use an **accessibility check list**
- **Ask yourself questions** such as:



Why is the screen reader reading the sidebar before the main article?

Do I have to tab through every page and every navigation before getting to the content?

What did I miss on the page?

Automated testing

Quick

Not all errors need human check

Can review hundreds of pages

First **validate HTML 5 code**, for example with <https://validator.w3.org/>

Then **automatically check for accessibility** (25% to 40% issues can be automatically detected)

- <https://wave.webaim.org/>
- <https://tenon.io/>

Automated testing: problems

Incomplete coverage

Alt-text

Script and dynamic content



Compliance != Accessibility

Manual testing

Slower

Human judgment

Depends on the experience of the tester

To perform **manual testing** you can create **user stories**

- a user story usually focuses on the value a software feature will deliver to an end-user, and an accessibility user story is no different

Manual testing: user stories

Examples of user stories

- As a keyboard-only user, I want to know where I am on the screen so that I can perform an action or navigate to other areas of the site
- As a screen reader user, I want to hear the text equivalent for each image button so that I will know what function it performs
- As a user who is color blind, I want links to be distinguishable on the page so that I can find the links and navigate the site

<https://tetralogical.com/blog/2022/05/26/how-to-write-user-stories-for-accessibility/>

User testing

Real users

- including users with disabilities who access the web pages with assistive technologies

Real users usually discover problems testers can miss

Testing for accessibility: how to

Use assistive technologies

- Screen readers
 - JAWS (Windows) / Edge
 - NVDA (Windows) / Firefox
 - VoiceOver (Mac) / Safari
 - Orca (Linux)
- Others
 - Screen magnifiers
 - Speech recognition

Testing for accessibility: how to

Checklist

- WebAIM
(see <https://webaim.org/standards/wcag/checklist>)
- The A11Y Project
(see <https://www.a11yproject.com/checklist/>)

Testing for accessibility: how to

Allocating time and resources to accessibility planning and testing results in a **more inclusive product**, it can help **avoid expensive mistakes** which need to be remedied when accessibility is only considered after a feature has been released

This is especially true if accessibility standards have not been met which might result in a legal case

Very nice video

Understanding Accessibility: WCAG's 13 Guidelines with
Kasey Bonifacio

<https://www.youtube.com/watch?v=RjpvOqZigao>



UNDERSTANDING ACCESSIBILITY: **WCAG's 13 GUIDELINES**

Additional technical details

If you like the subject you can read the introductory module on Accessibility

<https://developer.mozilla.org/en-US/docs/Learn/Accessibility>