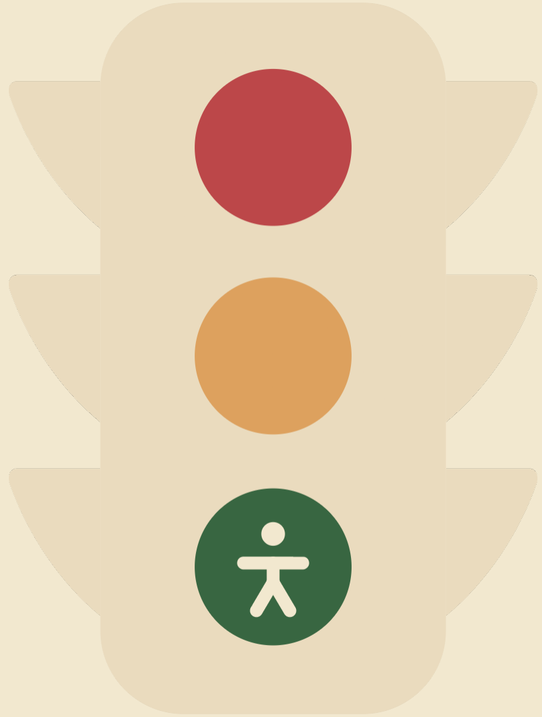
PORTO WORDPRESS MEETUP

JULY 30, 2026

Web and email *accessibility* isn't optional anymore.



Fabio Biocchetti
Web Developer

Hi, I'm *Fabio*. 👋

- Over ten years building things for the web
- Started front-end, then specialised in email development and marketing technology
- Previously at Babbel and Automattic
- Curious and passionate about digital experiences

> One subject kept coming back during these years:
>
> *Accessibility.*

Before we begin.

🇮🇹 → 🇩🇪 → 🇵🇹

Italy → Germany → Portugal

First names with the worst reputation in EU countries.

Thank you for coming anyway.

Today's *plan*.

1
What a11y means
Not (just) the acronym.

2
Why now?
The legal and practical context.

3
Web accessibility
Common problems, practical fixes.

4
Email accessibility
Same principles, different battlefield.

5
Deterministic AI use
Where it helps and where it doesn't.

6
Starter kit
What you can start doing tomorrow morning.

AND THEN

Q&A, plus a couple of live audits if we have time.

01

What *accessibility* means.

Spoiler: it’s not just screen readers.

What is *accessibility*?

Removing barriers that prevent people from using websites, apps, and emails.

a11y = accessibility

11 letters between the a and the y.

Users we sometimes forget to *design* for.

Visually impaired

Grey on grey looks clean on your monitor. For a lot of people it looks like nothing at all.

Hearing impaired

Video with no captions, audio with no transcript. That only works for people who can hear, in a quiet room.

Motor impaired

Hover-only menus and tiny click targets are fine if you have a steady hand and a mouse.

Dyslexia & cognitive

Walls of text, no headings, no structure. Reading the page becomes difficult.

Does it mean it's all about disability?

Not *exclusively*.

Some barriers reach *everyone*.

The same barrier can be permanent for one person, temporary for another, and situational for you, this afternoon. [1]

PERMANENT

Blindness
→ screen reader, or nothing

Deafness
→ captions, or nothing

Limited motor control
→ keyboard only

TEMPORARY

Broken arm
→ one hand for a month

Eye surgery
→ text zoomed to 200%, dark mode

Migraine
→ no motion, no glare

SITUATIONAL

Bright sunlight
→ low contrast disappears

Noisy train
→ video without subtitles

Holding a baby
→ one thumb, tiny buttons

> **There is no “normal user” to design for.**

[1] Persona spectrum, Microsoft Inclusive Design toolkit

What is *WCAG*?

Web Content Accessibility Guidelines, developed by the W3C. [1]

Technical guidelines that laws (including the European Accessibility Act) reference as the standard to meet.

Born for the web, but the principles apply to all digital products (email and apps) too.

VERSIONS STACK, THEY DON'T REPLACE

2.0 (2008) → 2.1 (2018) → 2.2 (2023)

EN 301 549, the European standard the law points to, requires WCAG 2.1 level AA. 2.2 adds criteria that are not mandatory everywhere yet.

[1] W3C, Understanding WCAG 2.2

Four *principles*: P.O.U.R.

Every WCAG criterion sits under one of these four[1], and they apply just as well to email and apps.

Perceivable

People can take the content in.

Alt text for images, captions for video, enough contrast.

Operable

People can actually use the interface.

Full keyboard navigation, visible focus, no impossible time limits.

Understandable

Content and behaviour make sense.

Clear language, errors that say how to fix them, consistent navigation.

Robust

It keeps working on other people’s technology.

Semantic HTML, ARIA only where needed, screen reader support.

[1] W3C, Web Content Accessibility Guidelines 2.1

Myths worth *busting*.

“Accessibility is only for blind users.”	**It also covers motor, cognitive and hearing needs, plus every situational barrier.**
“Accessibility is too expensive.”	**Expensive if you add it at the end. Free if it is part of how you build.**
“My users don’t need it.”	**You do not know that.**

> **87 million people** in the EU live with some form of disability, roughly one in five.[1]

[1] European Commission, Persons with disabilities in the EU

02

Why *now*?

The legal and practical context.

The law is no longer *theoretical*.

The Web Accessibility Directive (2016) made public-sector sites and apps accountable.

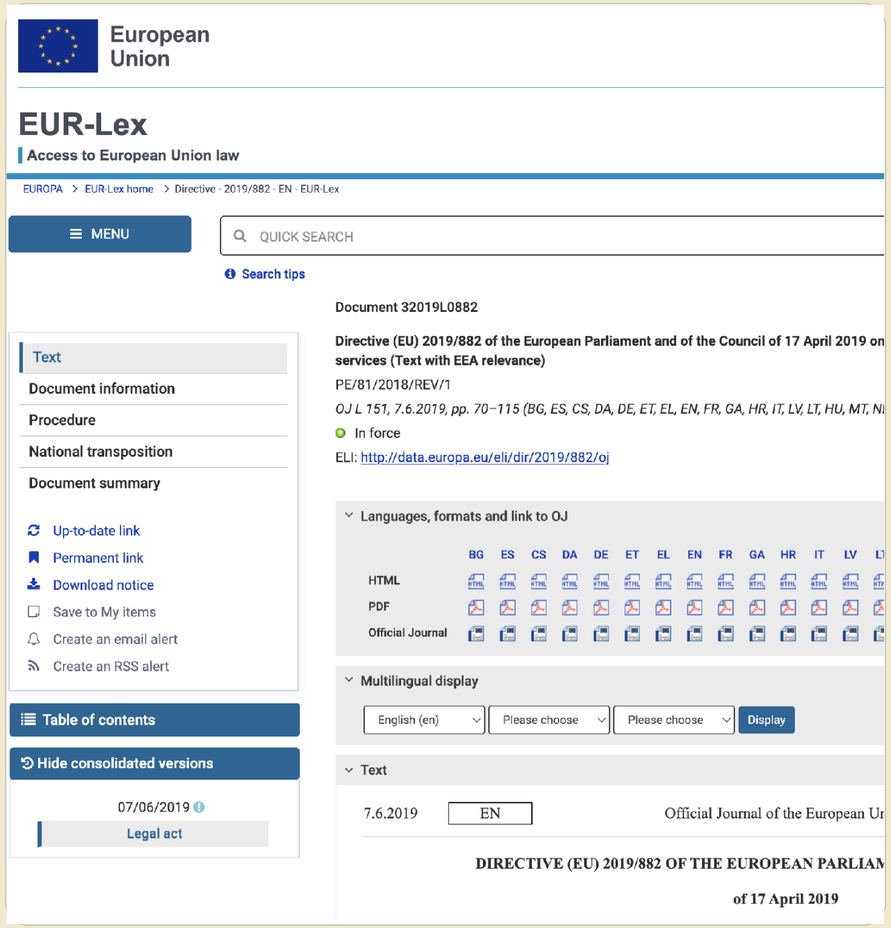
The European Accessibility Act (2019) extended that to the private sector. [1]

National transposition: June 2022.

Applies since: 28 June 2025.

> **In Portugal:** Decreto-Lei 82/2022.[2] Enforcement sits with the sector regulators, ASAE, ANACOM, Banco de Portugal, CMVM, IMT.

Yes, it is already in force.



[1] Directive (EU) 2019/882 (EUR-Lex)

[2] Decreto-Lei n.º 82/2022, de 6 de dezembro (DRE)

What *the law* says (and doesn't).

COVERED

- E-commerce
- Banking and payments
- Transport
- E-books
- Audiovisual services

NOT COVERED

- Purely informational websites
- Personal blogs
- Microenterprises: less than 10 employees and under €2M turnover [1]

> **ACCESSIBILITY STATEMENT**
>
> A public page saying what you comply with, if/what still fails, and how a user reports a problem.
> Public bodies have to publish one; private services only if in scope.

[1] Directive (EU) 2019/882, Art. 4 and Annex I

[2] Decreto-Lei n.º 82/2022, Art. 36

What happened in the *first year*.

No record fine has landed yet, but things are changing. [1]

France	A court ordered Carrefour France to make its shop and app accessible within six months, with a fine for every day of delay.
Sweden	The telecom regulator PTS began inspections in October 2025 and logged 124 consumer complaints, 110 of them about services.
Netherlands	The ACM sent formal information requests to online retailers, including companies based outside the EU.
Germany	Warning letters from law firms and competitors, using accessibility as an unfair-competition lever.

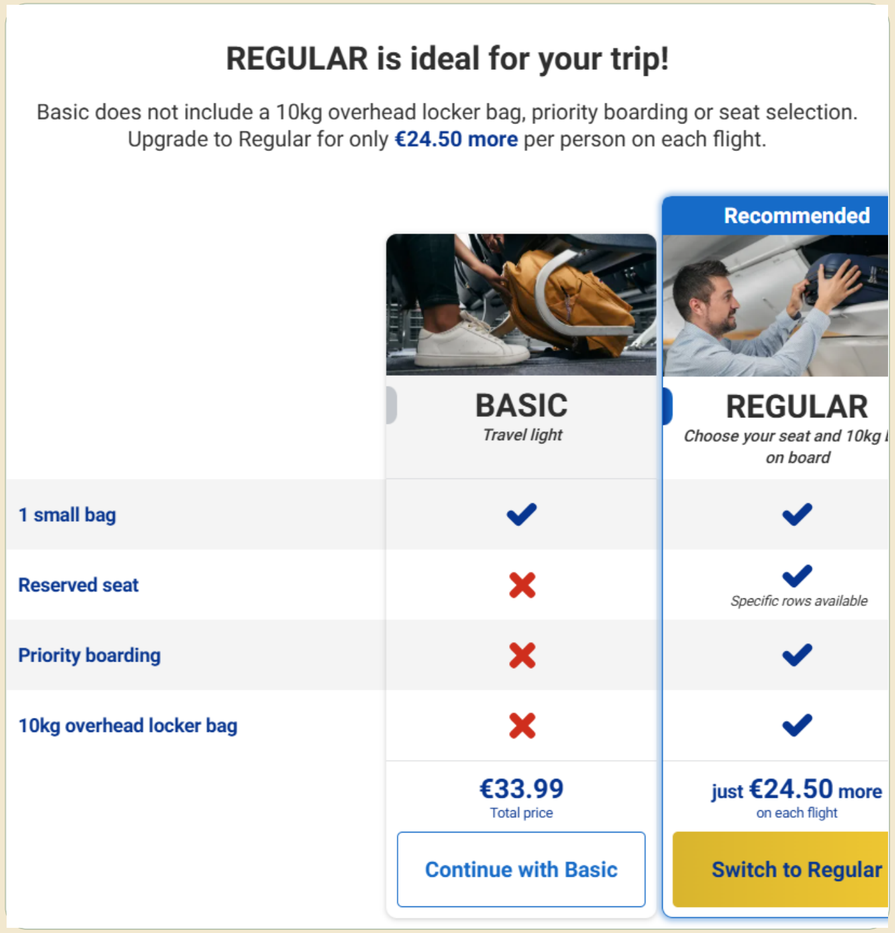
[1] Level Access, “EAA compliance in 2026: how enforcement has evolved”, June 2026

Compliance is just the *floor*.

Meeting WCAG 2.1 AA through EN 301 549 gets you legally safe. It does not get you a good product.

A form that passes automated testing but confuses every real user is still a bad form.

> **Compliance does not automatically create a good experience.**



03

The *web.*

The good old stuff.

If it only works with a *mouse*, it doesn't work.

`Tab` `⇧ Tab` `Enter` `Space` `Esc` That's all some users have.

The usual suspects: hover-only dropdowns, modals that ignore Esc, carousels with no key support.

BAD **HOVER ONLY**

Menu ▼

- Profile
- Settings
- Logout

```
<div class="trigger">Menu</div>
```

Opens on hover. Ignores the keyboard.

GOOD **BUTTON + KEYBOARD**

Menu ▼

- Profile
- Settings
- Logout

```
<button aria-expanded="true">Menu</button>
```

Enter to open, arrows to move, Esc to close.

One of the few ARIA attributes to keep in mind: `aria-expanded` is the only way to say "this menu is open".

Tab order follows your HTML.

Focus moves in the order your markup is written, until someone “fixes” it with **`tabindex`** numbers.

BAD **POSITIVE TABINDEX**

Third First Fourth Second

tab 1 tab 2 tab 3 tab 4

```
<a tabindex="3">Third</a>
```

Focus jumps 3 → 1 → 4 → 2. Chaos.

GOOD **NATURAL DOM ORDER**

First Second Third Fourth

tab 1 tab 2 tab 3 tab 4

```
<a href="…">First</a>
```

No tabindex needed. Order follows the HTML.

You removed the *focus ring* because it was ugly.

The most expensive CSS line you will ever write. It removes the only way some users know where they are. Restyle the focus ring instead of deleting it.

BAD OUTLINE REMOVED

Click me

```
:focus { outline: none; }
```

Focus is invisible. The user is lost.

GOOD VISIBLE FOCUS

Click me

```
:focus-visible { outline: 3px solid #BC4749; }
```

Clear, visible, on-brand.

Your images are invisible without *alt text*.

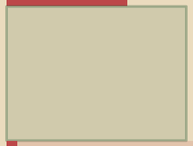
Every `<img>` without an alt attribute is a black box for a screen reader.

Write what the image communicates.

Decorative → `alt=""` (empty, correct: the reader skips it)

Informative → describe what it says

BAD **MISSING ALT**



Screen reader says: **“team underscore 001 dot jpg”**

```
<img src="team_001.jpg">
```

No alt attribute, so the file name gets read out loud.

GOOD **DESCRIBES CONTENT**

Screen reader says: **“Our design team, five people”**

```
<img src="team.jpg" alt="Our design team">
```

Communicates meaning.

Grey on grey isn't *minimalism*.

WCAG 2.1 AA MINIMUM CONTRAST RATIO

4.5:1 normal text

3:1 large text[1]

Also, an error marked with colour alone, a red border with no text and no icon, is invisible to color blind users.

BAD **LOW CONTRAST**

This is subtle grey text.

```
color: #999999; /* 2.8:1 on white */
```

Fails WCAG. Barely readable for many.

GOOD **SUFFICIENT CONTRAST**

This is readable text.

```
color: #333333; /* 12.6:1 on white */
```

Passes AA. Readable for everyone.

[1] WCAG 2.1, Success Criterion 1.4.3 Contrast (Minimum): large text is 24px, or 19px bold

Your placeholder is not a *label*.

Placeholder text disappears the moment your users start typing.

Make sure to include a label and provide clear guidance on what is going on.

Stop using *divs* for everything.

A `div` pretending to be a button is bad practice.

You give up keyboard support, focus management and screen-reader semantics, all of which the real element hands you for free.

> The first rule of ARIA (Accessible Rich Internet Applications) is not to use ARIA. Only use it as fallback when no native option is available. i.e. `aria-expanded`, `aria-describedby`, `aria-live`.

Just use the right HTML element. It's less work.

BAD **DIV PRETENDING**

Submit

```
<div role="button" tabindex="0"
onclick="…">Submit</div>
```

No keyboard, no focus, no semantics.

GOOD **NATIVE BUTTON**

Submit

```
<button>Submit</button>
```

Everything works. For free.

04

Now do it in *email*.

Everything you just learned, but with more chaos.

Same rules. *Older* tools.

Everything we saw earlier applies to email, as well.
What changes is what you are allowed to build it with, and how little room there is for error.

STILL TRUE

- Real text beats text in an image
- Alt text and reading order matter
- Contrast rules do not change
- Links say where they go

EMAIL-SPECIFIC LIMITATIONS

- No flexbox and no grid, so tables for layout
- Inline styles, because stylesheets get stripped
- No JavaScript, patchy ARIA support
- Dark mode applied to you, not by you

Email is still built on *tables*.

Tables for layout are fine in email. Six of them nested inside each other are not, because the screen reader reads the scaffolding instead of your message.

Keep the tables for layout where possible, use `role="presentation"`.

BAD TABLE SOUP

Content buried in nested tables

Screen reader: “table, 1 row… table, 1 row… table…”

GOOD SEMANTIC STRUCTURE

Newsletter title

One heading level per section, then the copy.

The reader navigates by headings, not by tables.

Dark mode will *eat* your email.

Many email clients will invert your colours automatically.
Transparent logos and hard-coded black text are the first casualties.

Update your assets, and test both modes before you send.

BAD **DISAPPEARING LOGO**

LOGO

Black text on a forced dark background.

Invisible: the client inverted the background and left the text alone.

GOOD **DARK-MODE AWARE**

LOGO — Solid plate behind the mark.

LOGO — Or an outlined mark that reads on any background.

LOGO — Or a stroked wordmark, where the letters carry the outline.

Your CTA is a *picture* of a button.

Images off, images blocked, images still loading: text baked into a graphic disappears in all three cases.

If it matters, it should be text.

BAD TEXT AS IMAGE


```
<img src="promo.jpg" alt="">
```

Screen reader: silence. Images off: nothing.

GOOD REAL TEXT

Shop Now — 50% Off

```
<a href="…">Shop Now — 50% Off</a>
```

Selectable, scalable, readable, translatable.

Bold text is not a *heading*.

Both emails look identical on screen, but only one of them tells a screen reader what it is looking at.

Semantic can be read out loud.

BAD NO STRUCTURE

Newsletter Title `<div>`

`<img> hero image` `alt=""`

Screen reader announces: **nothing. No heading, no image.**

Looks like a title, reads like nothing.

GOOD SEMANTIC HTML

Newsletter Title `<h1>`

`<img> hero image` `alt="Team at work"`

Screen reader announces: **“heading level 1: Newsletter Title”, “image: team at work”**

Same look, plus a landmark you can jump to.

05

AI, without making things *worse*.

Yes, you can use it. No, not like that.

Some of it well. Some of it *badly*.

With accessibility, AI behaves like it does everywhere else: a fast drafter and, at times, a terrible reviewer.

TYPICALLY WORKS WELL

- First drafts of alt text
- Semantic markup from a design
- Explaining a WCAG criterion in plain words
- Palette pairs that pass 4.5:1

NOT IDEAL

- Invents ARIA attributes that don’t exist
- Claims contrast passes when it doesn’t
- “Image of a smiling person” for a data chart
- Adds `role="button"` to a div and calls it done

How to write a *deterministic* prompt.

Specificity beats politeness.

BAD VAGUE PROMPT

> “Make this accessible.”

It “guesses” what accessible means to you.

GOOD SPECIFIC PROMPT

> “Review this web page against WCAG 2.1 AA standards, going through the Perceivable, Operable, Understandable and Robust principles.
>
> Tell me where it fails on keyboard use, visible focus, labels, error messages, contrast and alt text. Give me the line number and the fix for each.”

Name the standard and what you care about.

06

Your *starter kit.*

Platforms and tools to make your life easier.

Quick *recap*.

Accessibility is inclusivity.

Everyone sits somewhere on the spectrum, situational or not.

You don’t have to be obsessed.

Look for design compromises to fix what blocks people, and avoid bad UI/UX.

Build it in, don’t bolt it on.

Design and develop sites, emails and apps with accessibility in mind.

Accessibility = Inclusivity ≠ Perfection

What you can use *today*.

See below, and open the **A11Y Project checklist** and keep it next to you.

WEBSITES

axe DevTools
Browser extension, automated issue detection

WAVE
Visual overlay: see the problems on the page

Lighthouse
Built into Chrome, a score and a starting point

IF IT'S WORDPRESS

Ally (Elementor)
Scans pages, suggests fixes, drafts your statement

Accessibility Checker
Equalize Digital: scans posts while you edit

WP Accessibility
Skip links and common fixes, no code

EMAIL

AccessiMail
Paste your HTML, get a report with fixes

Email on Acid / Litmus
Rendering tests, plus built-in accessibility checks

Parcel
Checker built specifically for email accessibility

Can I email
Which HTML and CSS actually work, client by client

MANUAL TESTING

NVDA
Free screen reader for Windows

VoiceOver
Built into macOS, Cmd+F5

Colour Contrast Analyser
TPGi, free desktop app

Your keyboard
Tab through the whole page, no mouse

Thank you. *Questions?*

If not, let's run a couple of live audits together.

Fabio Biocchetti

Find me on GitHub and LinkedIn