

# Web Fundamentals



**Session-08**

**Accessibility & Best practices**

# Agenda



- 01 Accessibility Intro
- 02 Accessibility Tools
- 03 Labels, Aria Roles, Alt, Tab Index
- 04 Performance, Security
- 05 HTML, CSS Best Practices
- 06 Hands-On, Summary & Resources

## What is Accessibility?



## Accessibility

- **Visually impaired** users will generally use a ***screen reader*** that describes the page using speech.
- **Low vision** users will generally use large fonts or something called a ***screen magnifier***.
- **Motor-impaired** users, who can't use a mouse, might use some kind of ***special keyboard*** or even a ***voice control interface*** to interact with your app.



## Why Accessibility?

- Improved search engine optimization (SEO)
- Compliance with laws and regulations – Many countries have rules that websites need to be accessible
- Better User Experience



# Accessibility Tools

## Screen Readers Talkbacks

Demo  
Walkthrough

[Home](#) > [Extensions](#) > [Screen Reader](#)



### Screen Reader

[chrome.google.com](https://chrome.google.com)

★★★★★ 1,008 ⓘ | [Developer Tools](#) | 200,000+ users

 By Google

Available on Chrome



### Caret Browsing

Navigate the web using keyboard instead of touchpad or mouse.



### Dark Reader

Apply dark themes for websites.



### JustRead

Create distraction free reading environments.



### High Contrast

Adjust colour contrast in webpages .

**No <Div> Soup**

**No <span> Salad**

# Create meaningful HTML structure

<div>

<div>

<div>

<header>

<section>

<footer>

---

👎 AVOID THIS

---

👍 TRY THIS

## Screen Readers Can't Identify Divs & Spans

- Not focusable
- Not activated by keyboard

---

```
// Instead of 'span' use 'a' tag for onClick event
```

```
<span onclick=""></span>
```

```
<a href="#"></a>
```

```
// Instead of 'div' use 'button' tag for onClick event
```

```
<div onclick="">Click Me</div>
```

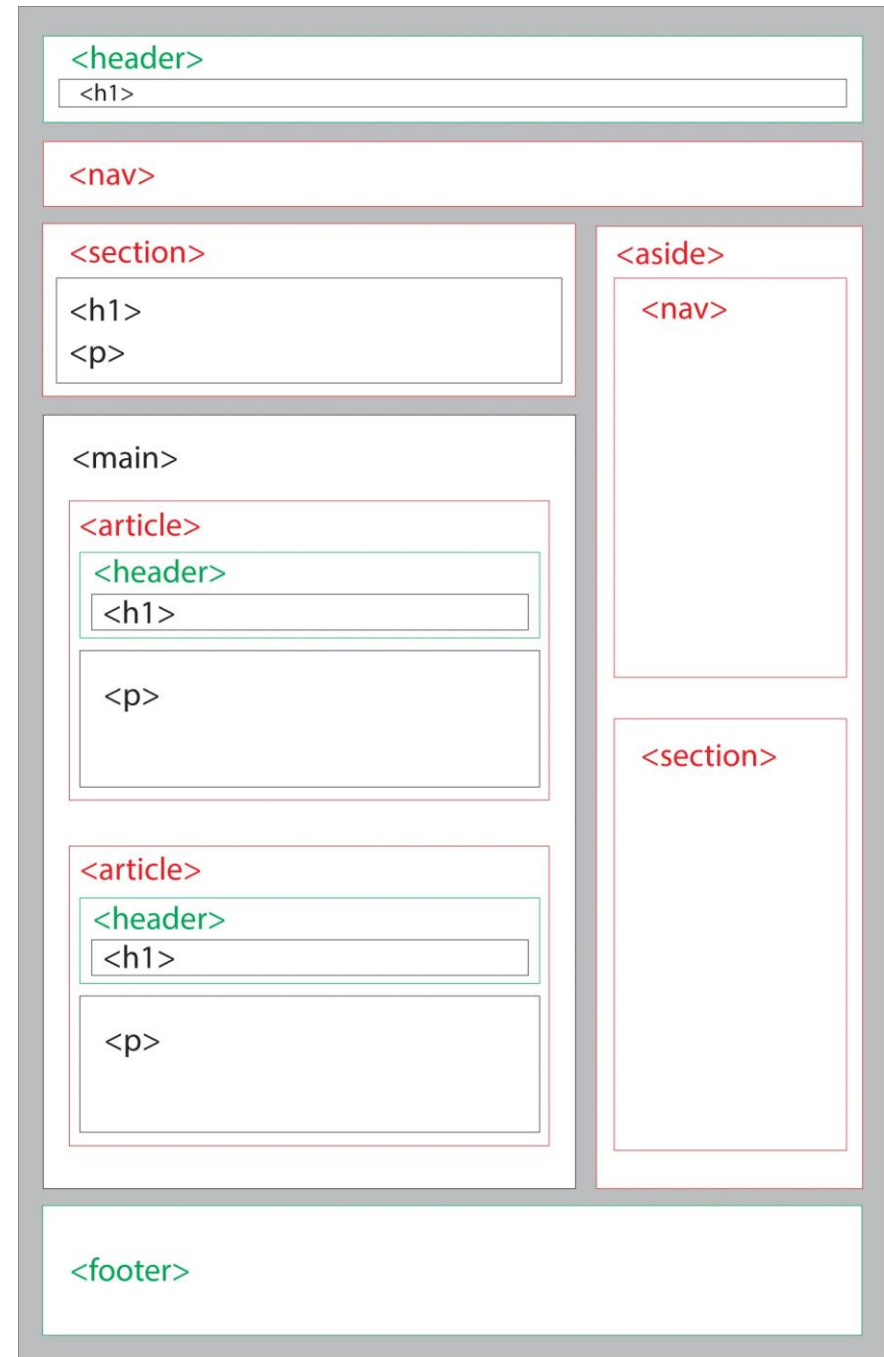
```
<button onclick="">Click Me</button>
```

---



# Accessibility #1

## - Semantic HTML



## Accessibility #2

**All images should have the alt attribute**

**All Form controls should have labels**

```
<label for="firstname">First name:</label>  
<input type="text" id="firstname"/>
```

```

```

## Accessibility #3

**For Keyboard only users  
use “TabIndex” to focus**

**Order in which elements  
receive focus when the  
user navigates**

---

```
// Natural tab order
<div onclick="" tabindex="0"> Clicky 1</div>

//Custom tab order
<div onclick="" tabindex="7">Clicky 2</div>

//Focusable but not in tab order. This is useful when hiding focusable controls
<div onclick="" tabindex="-1">Clicky 3</div>
```

## Accessibility #4

***Role: Purpose of HTML element***

```
<!-- Widget Roles -->  
<div role="button" tabindex="0">Click me!</div>  
  
<input type="text" role="textbox" aria-label="Type your name here">  
  
<div role="slider" aria-valuemin="0" aria-valuemax="100" aria-valuenow="50">
```

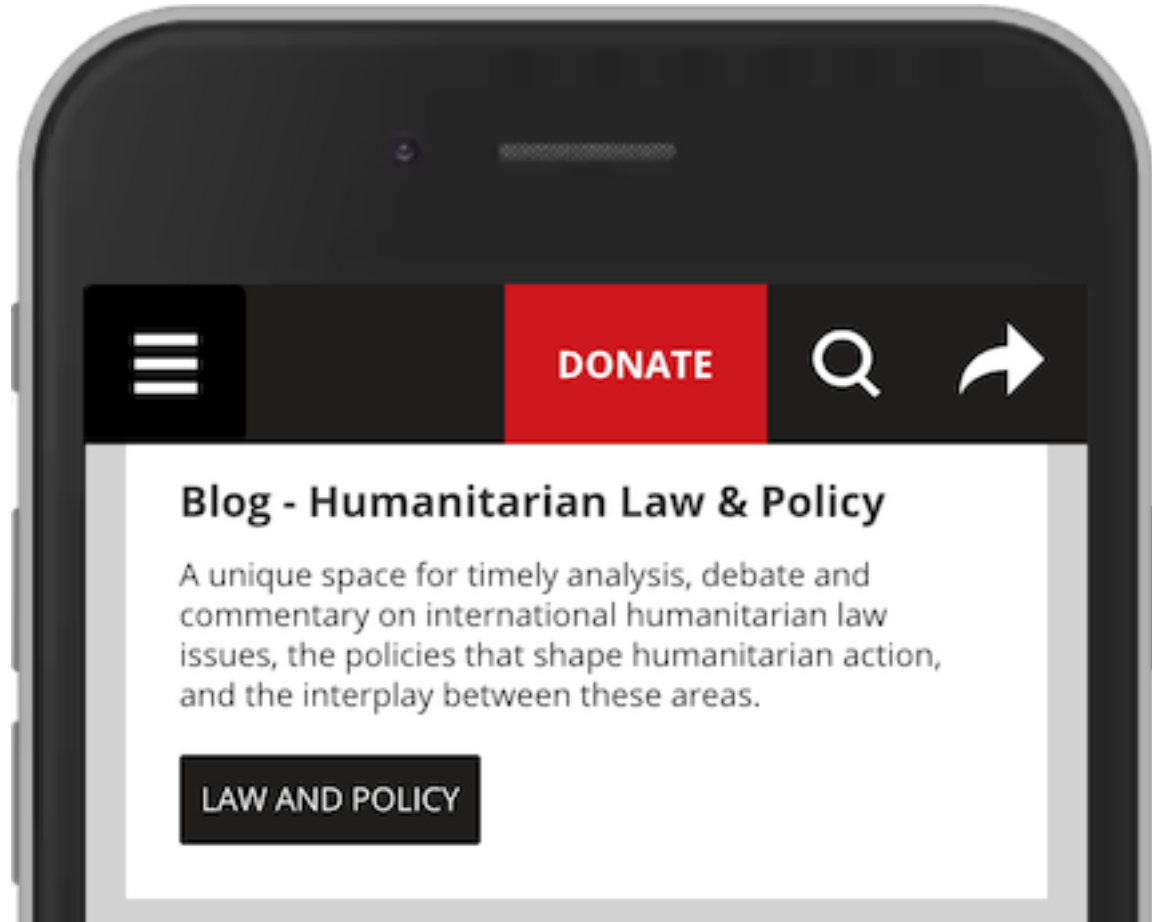
## Accessibility #5

***Aria-Label:*** label or name  
for an HTML element

```
<!-- Widget Roles -->  
<div role="button" tabindex="0">Click me!</div>  
  
<input type="text" role="textbox" aria-label="Type your name here">  
  
<div role="slider" aria-valuemin="0" aria-valuemax="100" aria-valuenow="50">
```

## Accessibility #6

***Link Text:*** A link text should explain clearly what information the reader will get



## Accessibility #7

***Contrast Ratio:*** web page need good contrast so that we make sure that it is perceivable for visually challenged users.



## **Performance Best Practices:**

- Minify HTML, CSS, JS
- Lazy load assets ( images/files/css/js)
- Reduce size of DOM, Load only what is required



## Security Best Practices:

- Encrypt/Obfuscate HTML, CSS, JS
- Use HTTPS for data transmission
- Always validate user inputs (interaction)

Option 1: Copy-paste your HTML here

```
<!DOCTYPE html><html lang="en"><head></head><body class="text-center"><form class="form-signin"><h1 class="h3 mb-3 font-weight-
normal">Please sign in</h1><label for="inputEmail" class="sr-only">Email address</label><input type="email" id="inputEmail" class="form-control"
placeholder="Email address" required autofocus><div class="checkbox mb-3"><label><input type="checkbox" value="remember-me"> Remember
me</label></div><button class="btn btn-lg btn-primary btn-block" type="submit">Sign in</button><p class="mt-5 mb-3 text-muted">&copy; 2017-
2018</p></form></body></html>
```

Option 2: Or upload your HTML file

File encoding

Choose File sample.html

UTF-8

Obfuscate

-Obfuscated HTML-

Copy

Save

```
<script
language="javascript">document.write(unescape("%3C%21%44%4F%43%54%59%50%45%20%68%74%6D%6C%3E%3C%68%74%6D%6C%20%6C%61%
6E%67%3D%22%65%6E%22%3E%3C%68%65%61%64%3E%3C%2F%68%65%61%64%3E%3C%62%6F%64%79%20%63%6C%61%73%73%3D%22%74%6
5%78%74%2D%63%65%6E%74%65%72%22%3E%3C%66%6F%72%6D%20%63%6C%61%73%73%3D%22%66%6F%72%6D%2D%73%69%67%6E%69%6E
%22%3E%3C%69%6D%67%20%63%6C%61%73%73%3D%22%6D%62%2D%34%22%20%73%72%63%3D%22%68%74%74%70%73%3A%2F%2F%67%65%
74%62%6F%6F%74%73%74%72%61%70%2E%63%6F%6D%2F%64%6F%63%73%2F%34%2E%30%2F%61%73%73%65%74%73%2F%62%72%61%6E%64
%2F%62%6F%6F%74%73%74%72%61%70%2D%73%6F%6C%69%64%2E%73%76%67%22%20%61%6C%74%3D%22%22%20%77%69%64%74%68%3D%
22%37%32%22%20%68%65%69%67%68%74%3D%22%37%32%22%3E%3C%68%31%20%63%6C%61%73%73%3D%22%68%33%20%6D%62%2D%33%2
0%66%6F%6E%74%2D%77%65%69%67%68%74%2D%6E%6F%72%6D%61%6C%22%3E%50%6C%65%61%73%65%20%73%69%67%6E%20%69%6E%3C
%2F%68%31%3E%3C%6C%61%62%65%6C%20%66%6F%72%3D%22%69%6E%70%75%74%45%6D%61%69%6C%22%20%63%6C%61%73%73%3D%22%
```

## Best Practices- HTML

- Indent and format properly.
- Use semantic HTML.
- Optimize images for web.
- Test code on browsers.
- Use fallback fonts.
- Don't keep empty ***“src”*** or ***“href”***
- Use accessibility best practices.
- Add meaningful comments to describe your code

## Best Practices- CSS

- Indent and format properly.
- Avoid inline styles
- DRY- Do not repeat yourself, Reuse by using classes
- Use external CSS.
- Use CSS selectors efficiently.
- Keep CSS organized.
- Test code on browsers.
- Use responsive design.

## Best Practices for your mini-project

- Minify CSS & JS
- Refactor your code
- Remove Duplicate code
- Avoid unnecessary tags in head tag
- Use Async & Defer wherever required

“If you want to learn to swim,  
jump into the water.”

–Bruce Lee



**Q & A**