

# Accessibility for the Apathetic

*Lyza D. Gardner*

Funka Accessibility Days | April, 2015 |

*[@lyzadanger](#)*



The *web*

should work

for *everyone*





Lately: much *talk* about accessibility



Elegant, thoughtful coverage



ARTICLES COLUMNS BLOG EVENTS TOPICS WRITE FOR US



# A LIST APART

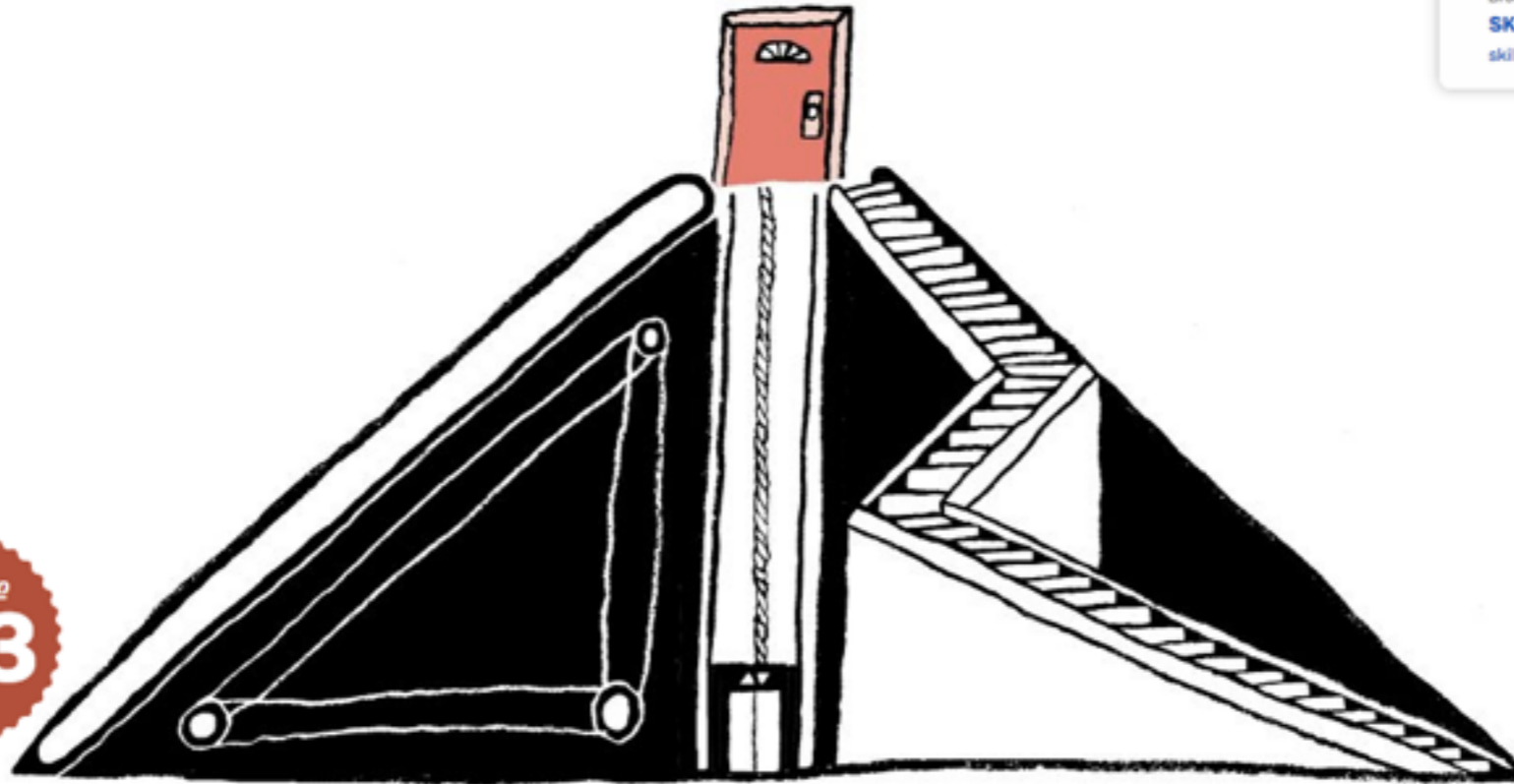
Skillfeed

Brought to you by:

**SKILLFEED**

[skillfeed.com/AlistApart](http://skillfeed.com/AlistApart)

Issue №  
**413**



## Reframing Accessibility for the Web

by **ANNE GIBSON** · February 03, 2015

Published in *Accessibility, Usability* · 43 Comments

<http://alistapart.com/article/reframing-accessibility-for-the-web>

Books

eBooks

Workshops

Job Board



## CODING

CSS

HTML

JavaScript

Techniques

## DESIGN

Web Design

Typography

In

B

M

iP

A

T

F

S

E

R

T

E

# Accessibility Originates With UX: A BBC iPlayer Case Study

By [Henny Swan](#)

February 23rd, 2015

Accessibility, Case Studies, User Experience

22 Comments

Not long after I started working at the BBC, I fielded

Advertisement



**tylersticka** 9:04 AM

Essential case study on accessibility issues with BBC's iPlayer UX:

<http://www.smashingmagazine.com/2015/02/23/bbc-iplayer-accessibility-case-study/>

Smashing Magazine

**A BBC iPlayer Accessibility Case Study**

Accessibility is about people. Henny Swan shows how BBC listened to users and actively included their feedback to redesigned the iPlayer.



**lyzadanger** 10:25 AM

@tylersticka was checking in and saw your Smashing link. That article is astonishingly well-written. Dang.

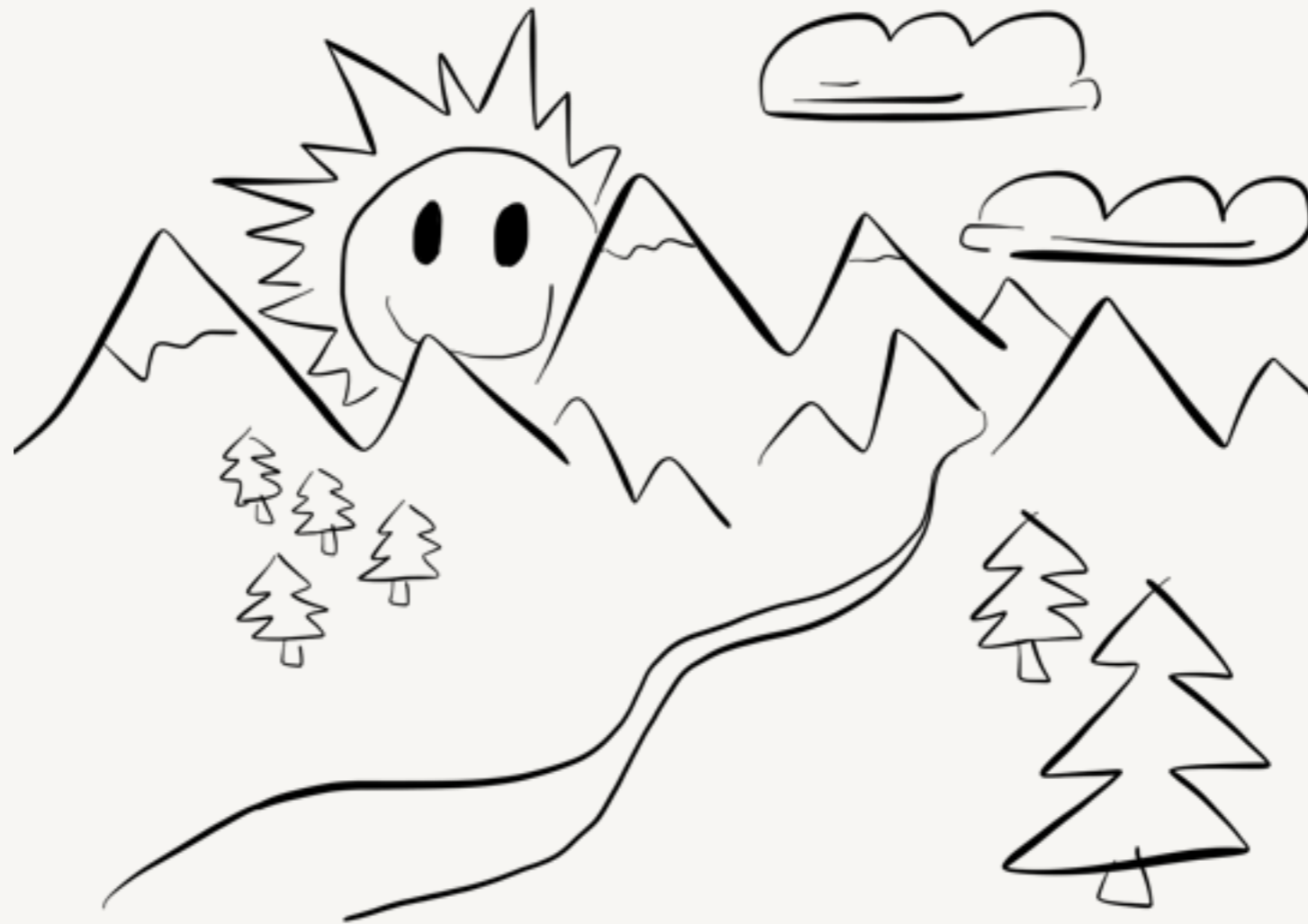
Fireworks

Wallpapers

I called the customer to establish what exactly the problem was, and together we navigated the home page using a screen reader. It was at that point I realized that, while all of the traditional ingredients of an accessible page were in place —

<http://www.smashingmagazine.com/2015/02/23/bbc-iplayer-accessibility-case-study/>





Makes me *hopeful* and dreamy

Reality



**This is your team**

Developers

Designers

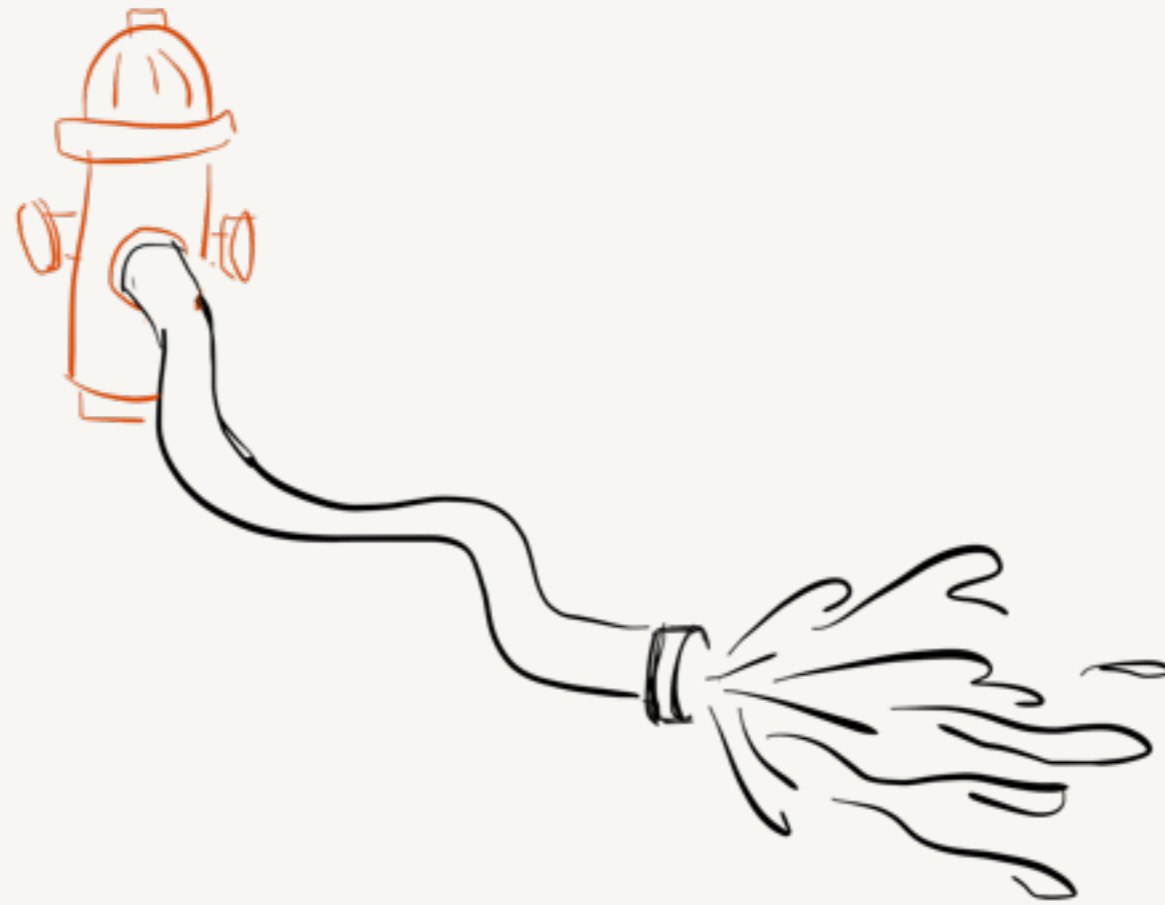
Dev..signers?

Desilopers?

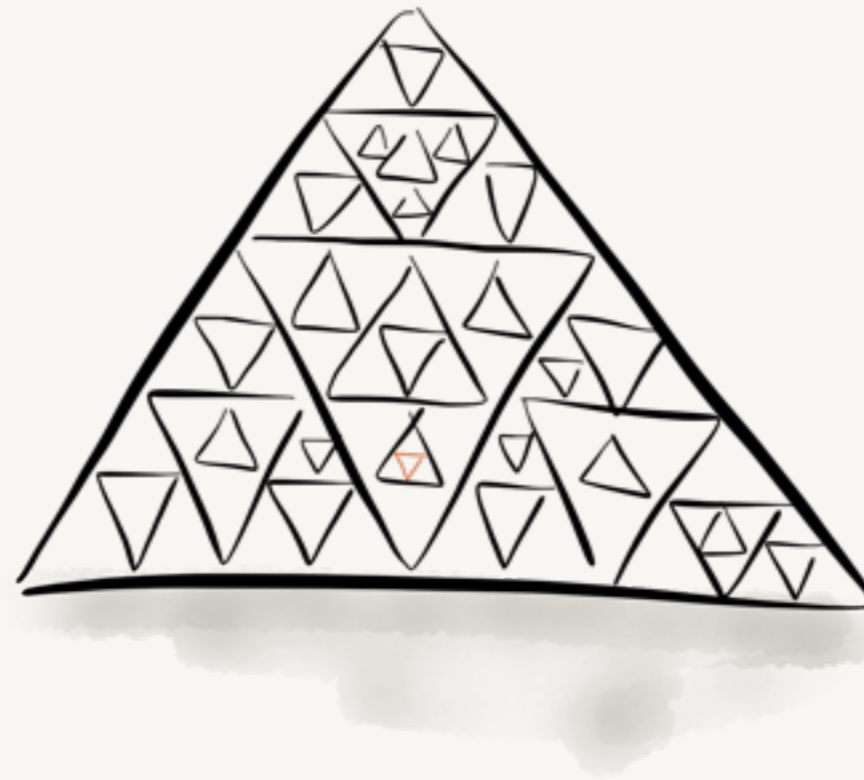
A team of people who build the web



**They're pretty busy**



**Their world looks like this**



Building the web is *complex*



Too much *pressure* and too many *details*





Leads to: burnout, apathy



Despair



But, again, here's your team



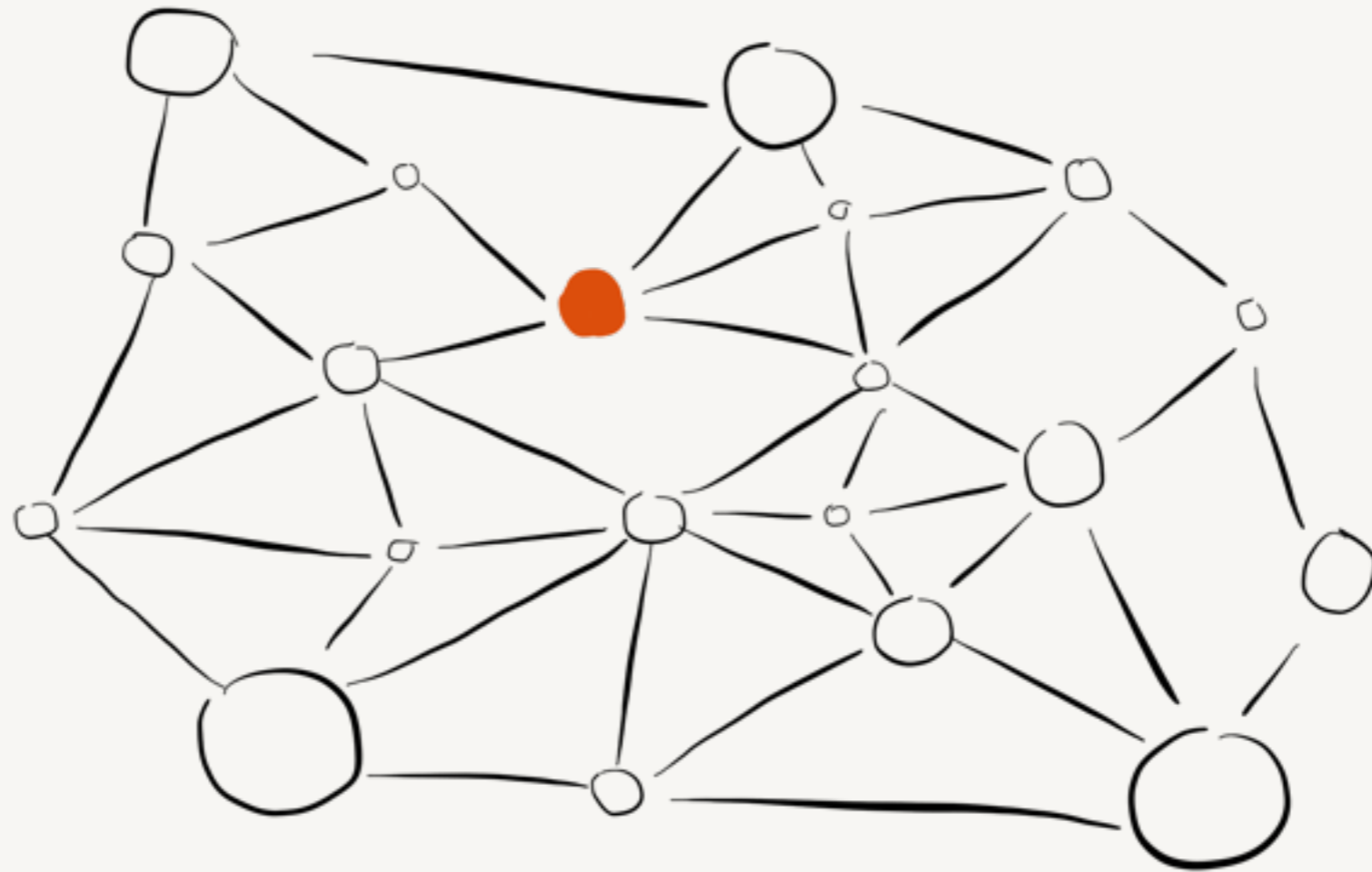
I believe in the people on your teams



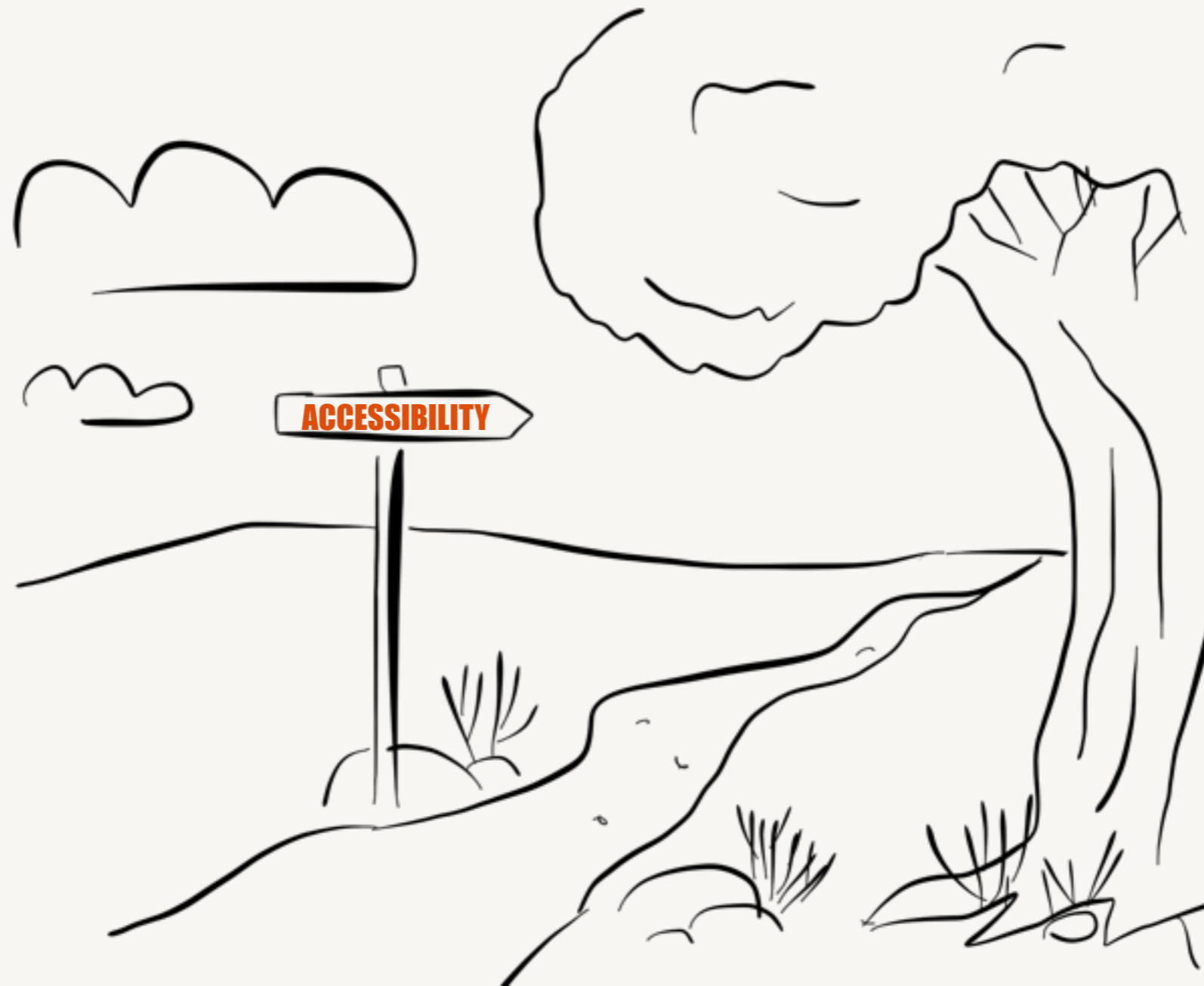
**I believe humans are compassionate**



And that builders desire ***excellence***



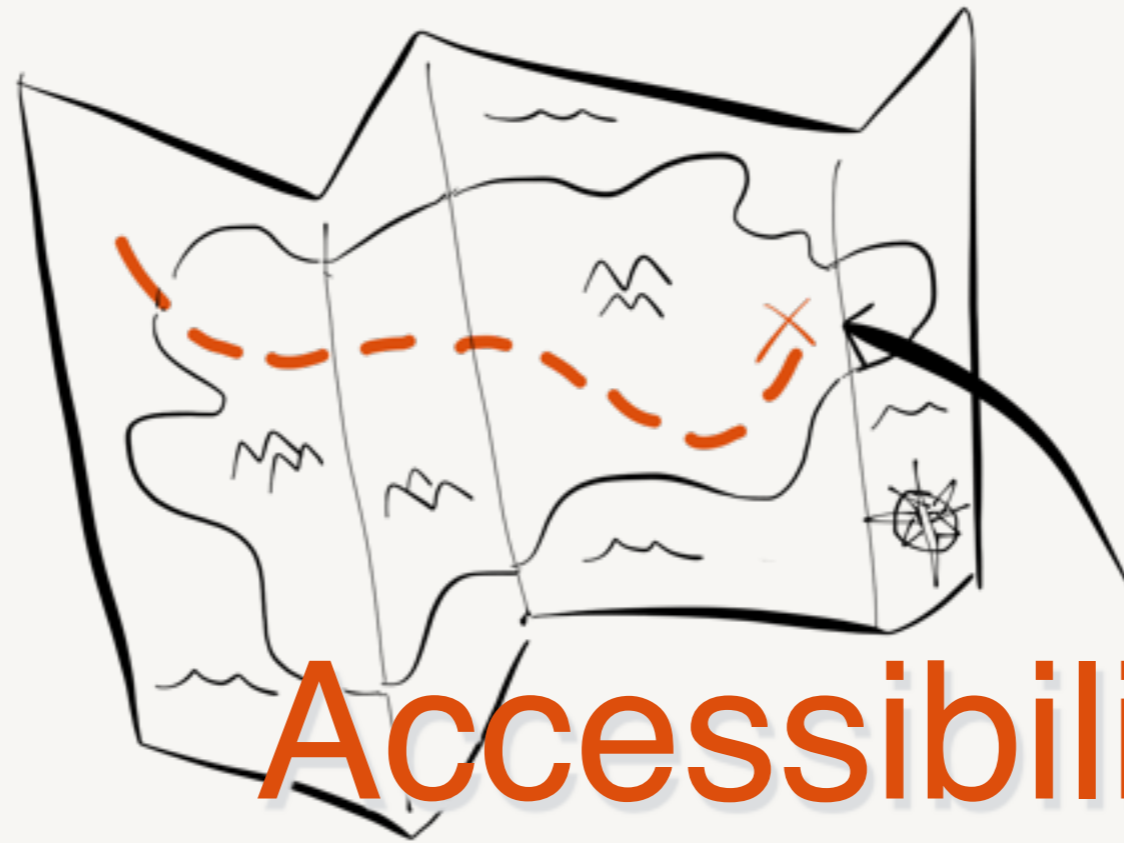
Accessibility is but ***one piece*** of the complex fabric of web construction



We need to help *lead* the way



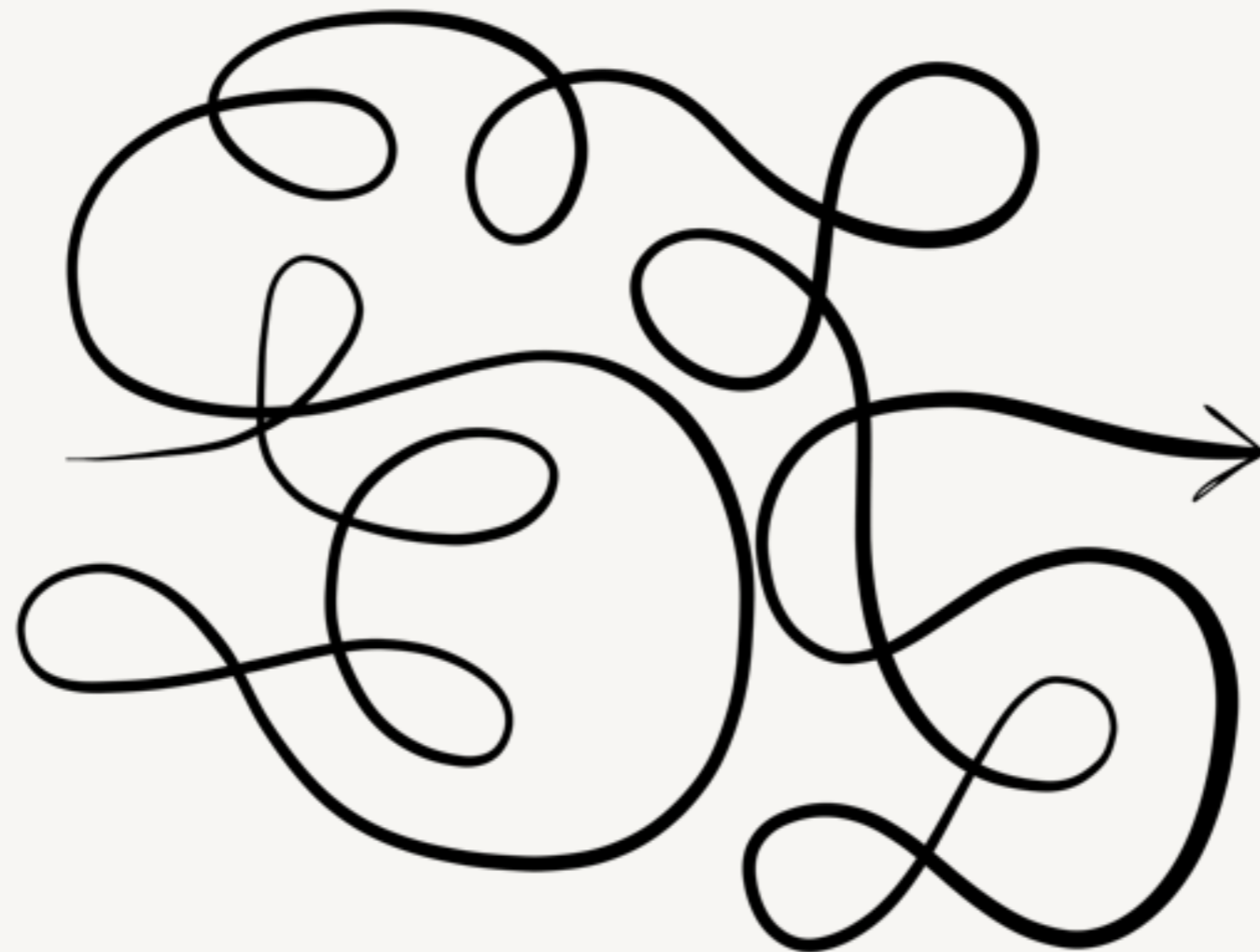
Finding a path



# Accessibility bliss

Our ideal path

Let's go!



Many *paths* to and through the web

# Many paths for the web

***Technical Context***

***Surrounding Processes***

***Individual Perspective***

# Many paths for the web

*Technical Content Surrounding Processes and Individual Perspectives*

# Many paths for the web

*Technical Context* Surrounding Processes and Individual Perspectives

# Many paths for the web

***Technical Context***





# Many paths for the web

***Technical Context*** “the nature of the Web”

generalist’s skillset

broad knowledge needed

inconsistent tech

changing standards

firehose of information

# Many paths for the web

***Technical Context*** Surrounding Processes Individual Perspective

“the nature of the Web”

generalist’s skillset

broad knowledge needed

inconsistent tech

changing standards

firehose of information

# Many paths for the web

**Technical Context** *Surrounding Processes* **Individual Perspective**

“the nature of the Web”

generalist’s skillset

broad knowledge needed

inconsistent tech

changing standards

firehose of information

# Many paths for the web

*Surrounding Processes*



# Many paths for the web

*Surrounding Processes* project requirements  
constraints  
deadlines  
communication woes  
development cycles  
client priorities

# Many paths for the web

## Technical Context *Surrounding Processes* Individual Perspective

“the nature of the Web”

generalist’s skillset

broad knowledge needed

inconsistent tech

changing standards

firehose of information

project requirements

requirements

constraints

communication woes

development cycles

client priorities

# Many paths for the web

## Technical Context Surrounding Processes *Individual Perspective*

“the nature of the Web”

generalist’s skillset

broad knowledge needed

inconsistent tech

changing standards

firehose of information

project requirements

requirements

constraints

communication woes

development cycles

client priorities

# Many paths for the web

*Individual Perspective*





# Many paths for the web

## *Individual Perspective*

priorities

technical decisions

skill level

best practices

strategic thinking

finding info

# Many paths for the web

## Technical Context   Surrounding Processes   *Individual Perspective*

“the nature of the Web”

generalist’s skillset

broad knowledge needed

inconsistent tech

changing standards

firehose of information

project requirements

requirements

constraints

communication woes

development cycles

client priorities

priorities

technical decisions

skill level

best practices

strategic thinking

finding info

# Many paths for building

## Technical Context

“the nature of the Web”

generalism required

broad knowledge needed

stuff changing all the time

firehose of information

priorities

## External Processes

project requirements

constraints

deadlines

communication woes

development cycles

client priorities

## Individual Practices

technical priorities

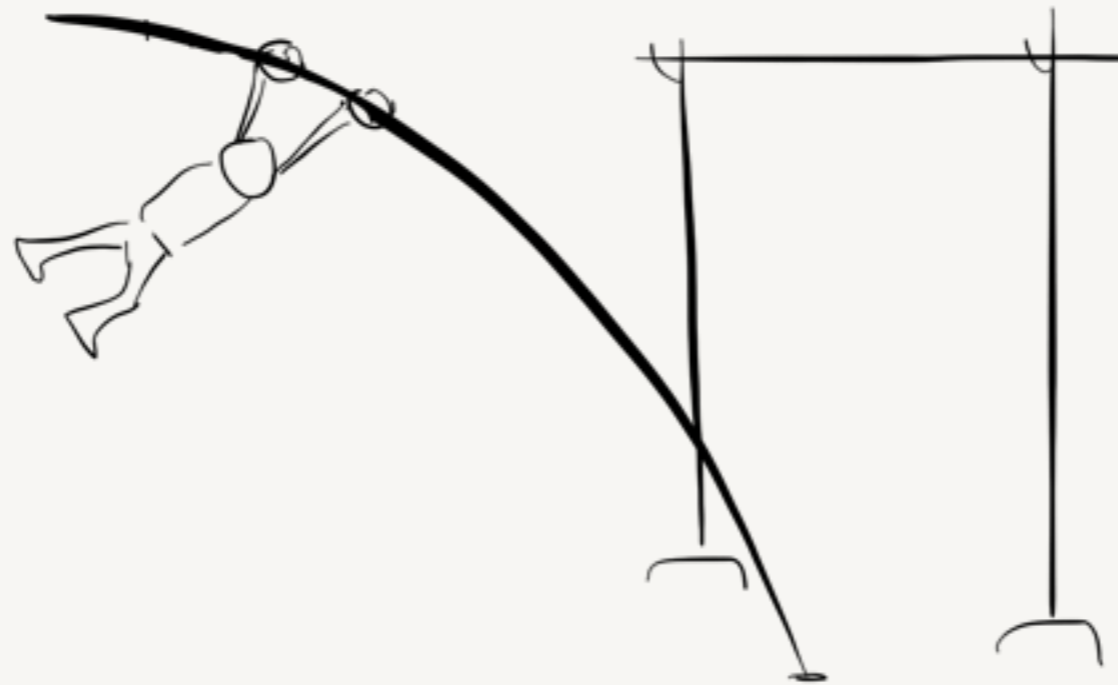
design/dev decisions

skill level

best practices

strategic thinking

finding info



We have ***hurdles*** to get past

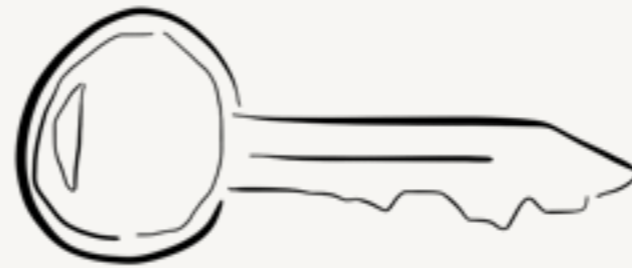


***Technical Context***

***Surrounding Processes***

***Individual Perspective***

This is our ***quest*** today



Two *keys* to this problem



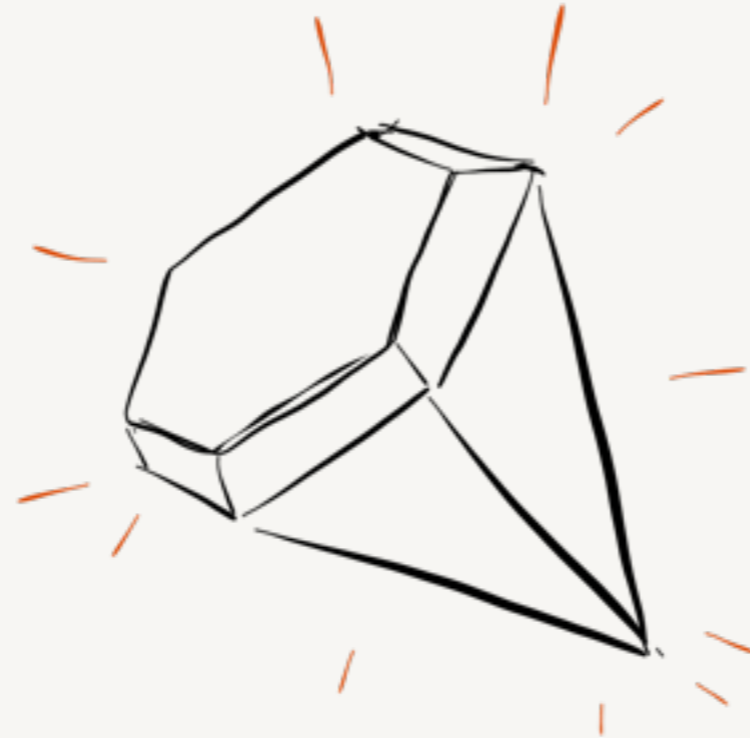
**Synthesis**



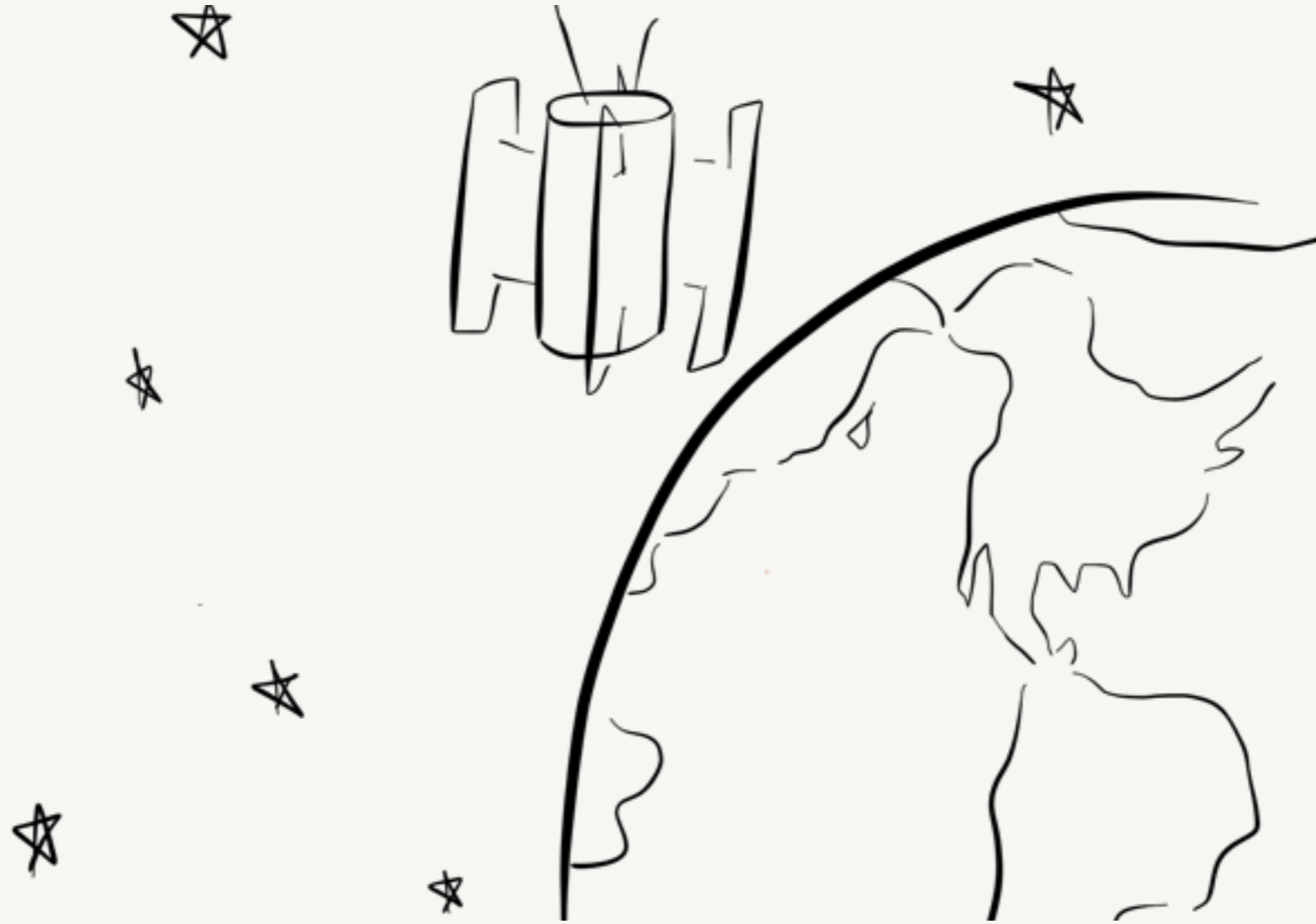
**Motivation**



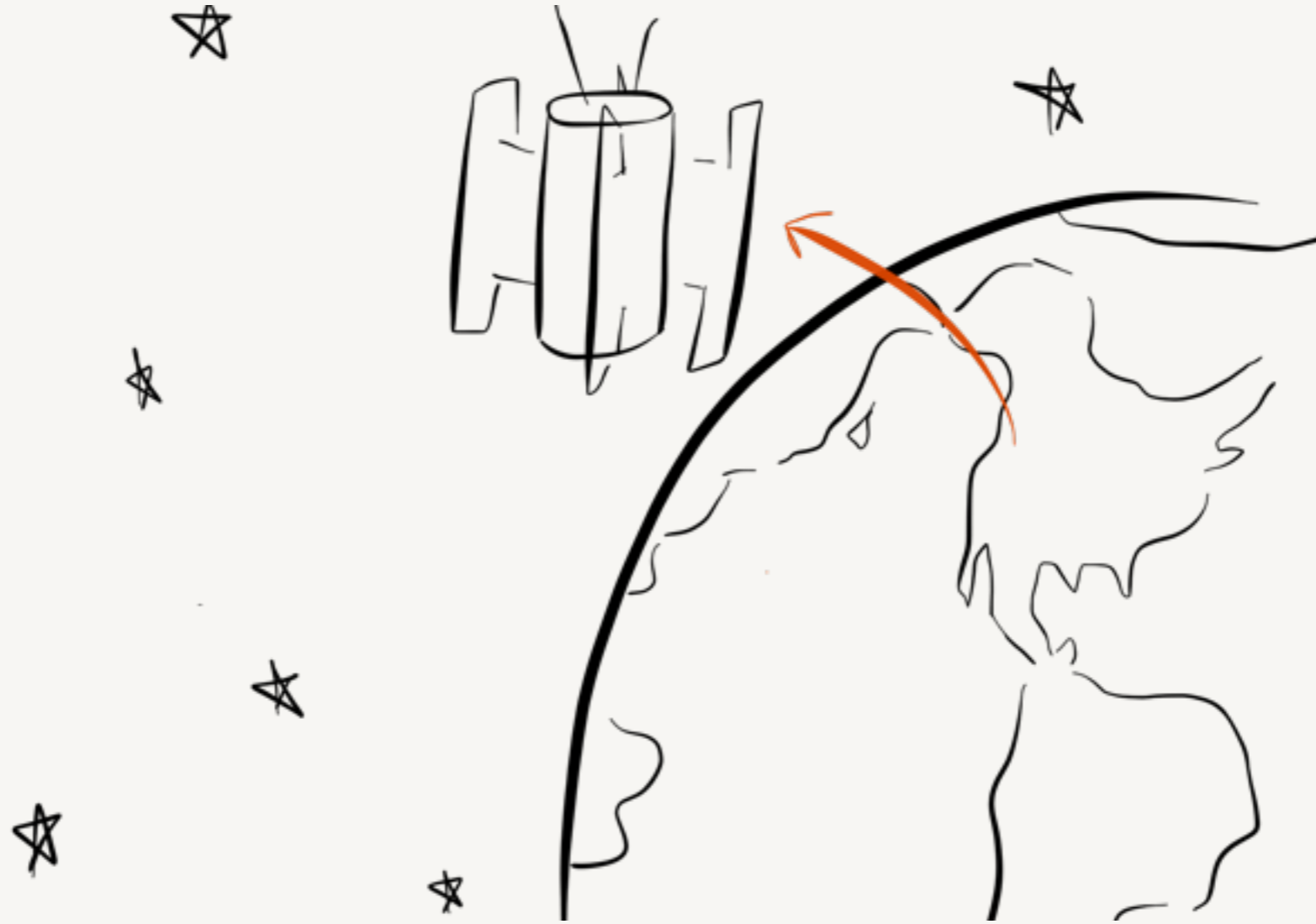
# The meaning of accessibility



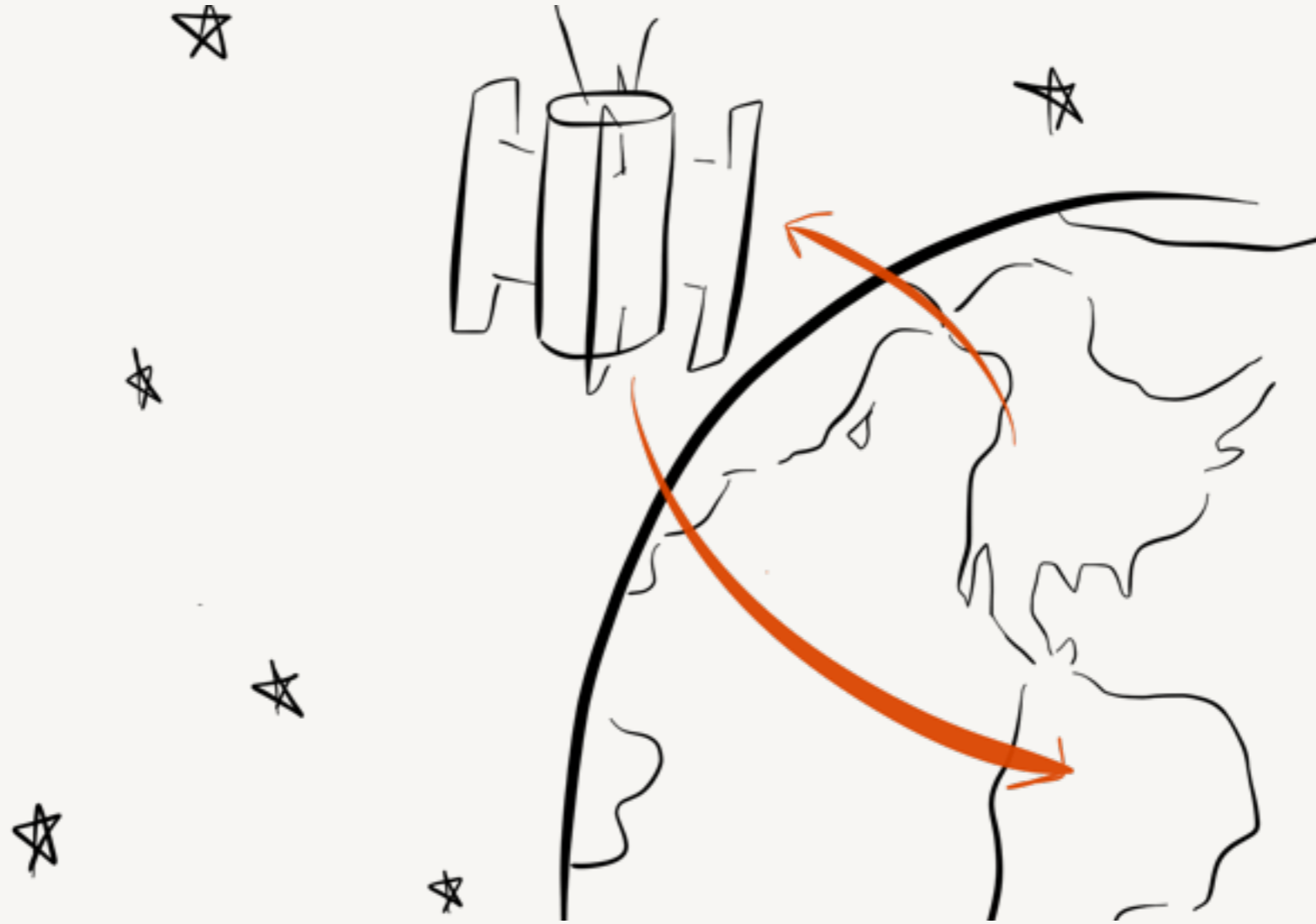
“***accessibility***” from another angle



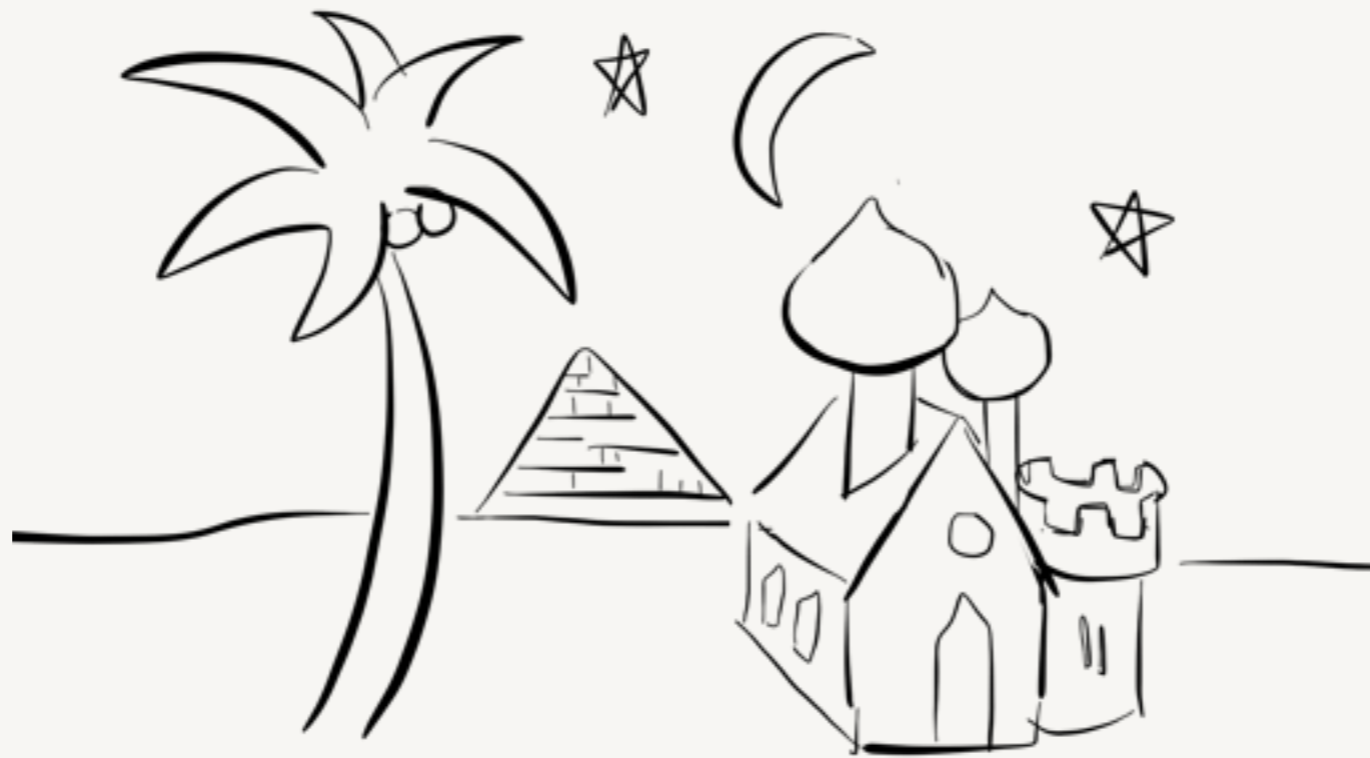
Feeling separated from the impact



Feeling separated from the impact



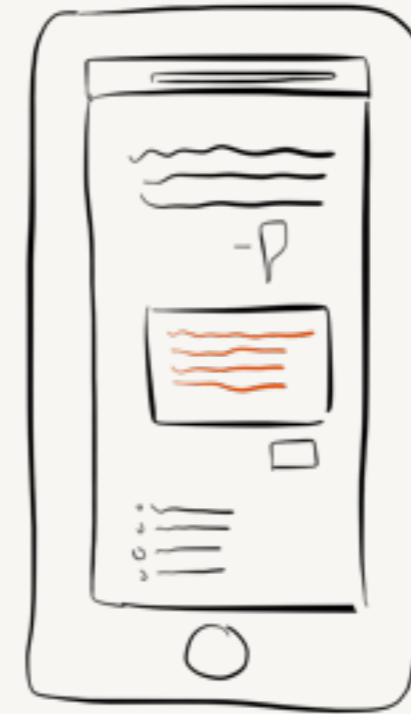
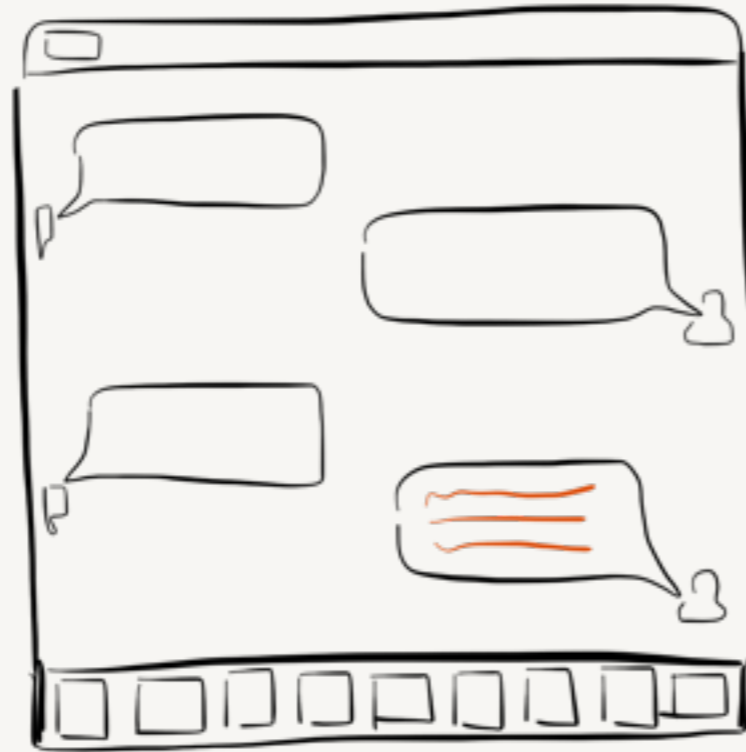
Feeling separated from the impact



Feels ***exotic***, exceptional, specific



Making things ***work*** for inputs



Making things *work* in lots of places

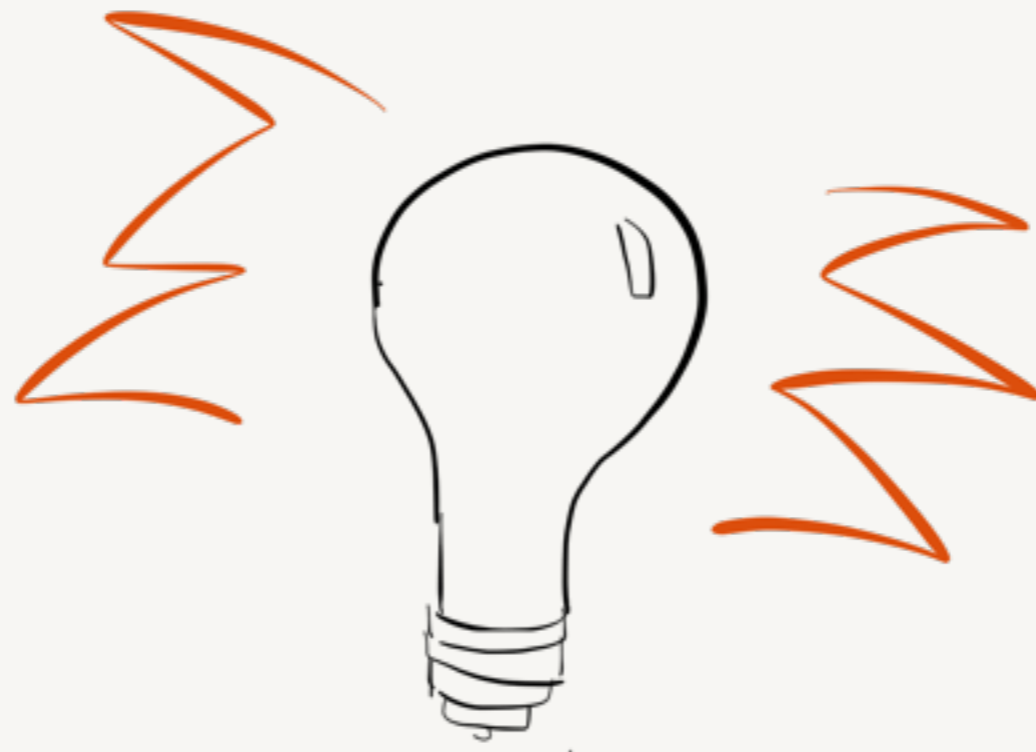




Making things *flexible*



We're ***adventurers!***

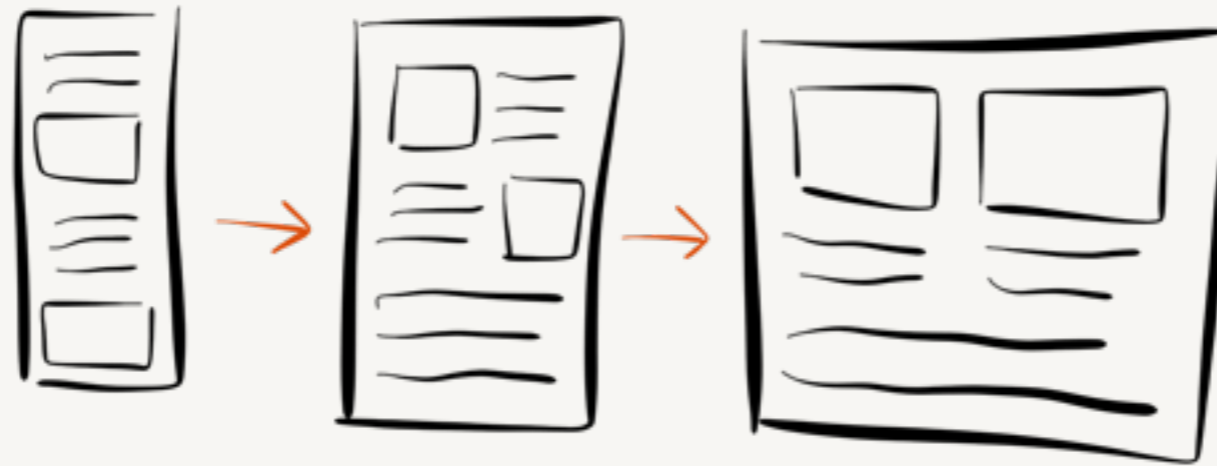


*Responsive Web Design!*  
*Progressive Enhancement!*

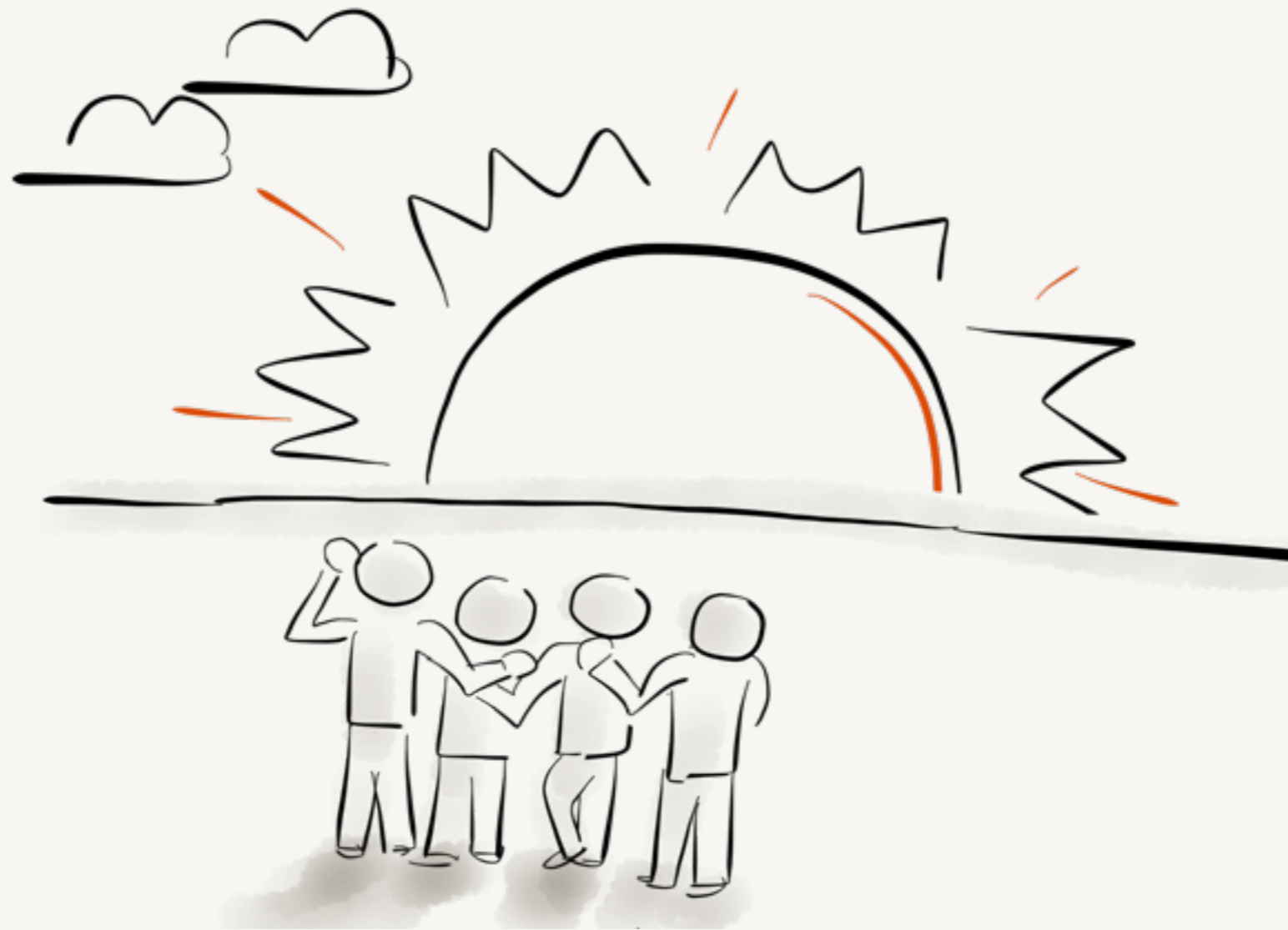
Sound familiar?



***Parallels*** with accessibility



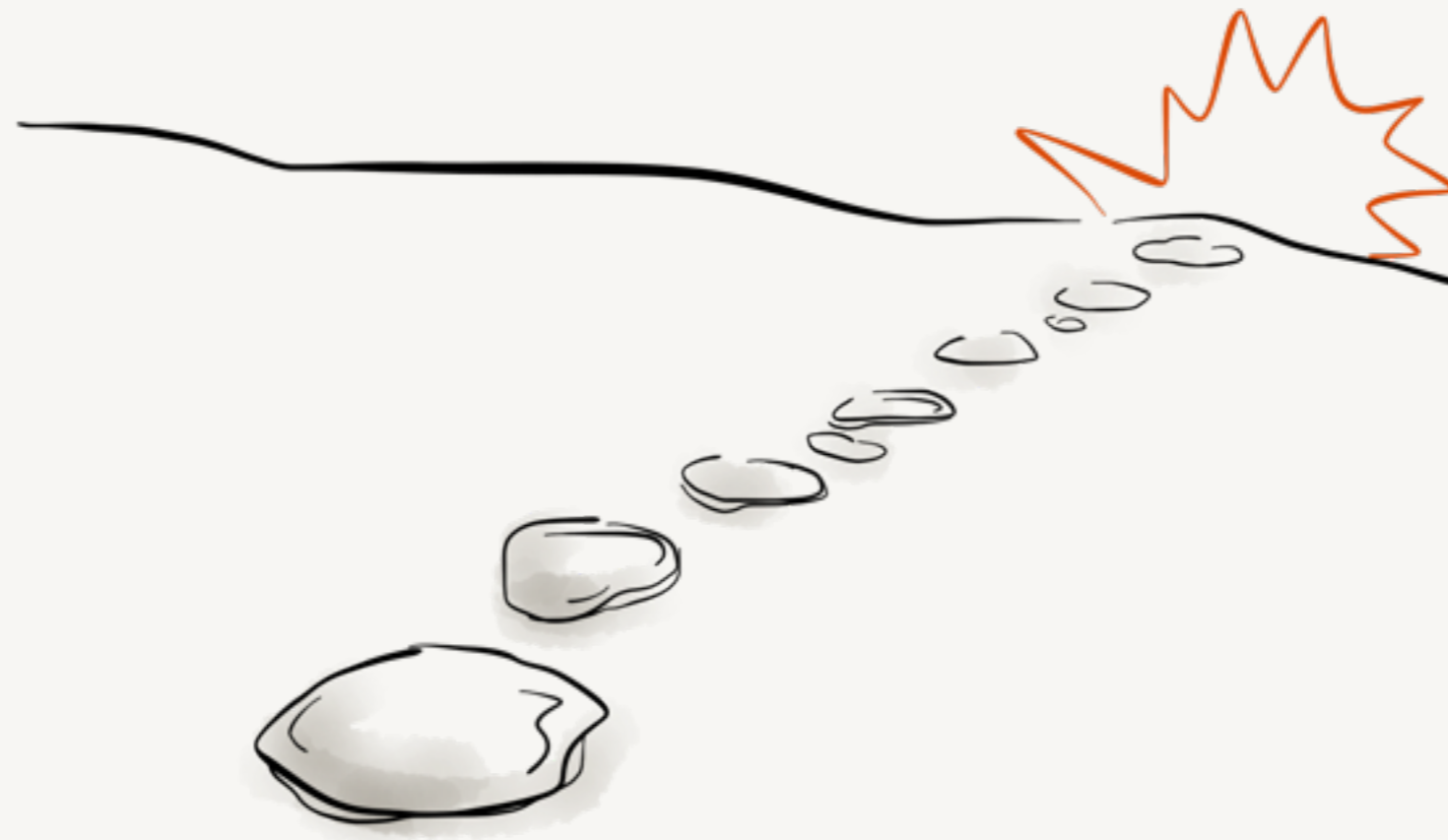
Fewer ***assumptions***, relinquishing control



***Connecting*** meaning to values



Putting it in ***context*** to build ***motivation***



**Building our path**



Doing what we already  
do, but better



Time to make stuff *happen*

# Two kinds of *doing*

Doing what we do now, better

***VS.***

Doing new, different things

Two kinds of *doing*

Doing what we do now, better

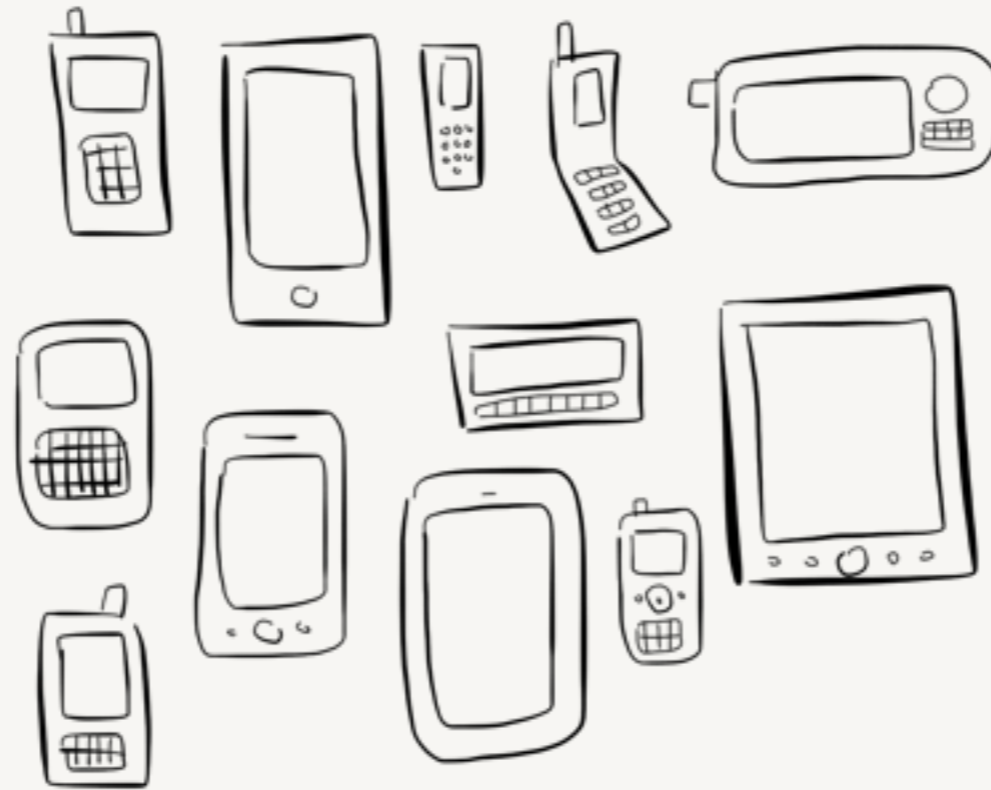
*VS.*

Doing new, different things

HTML is pretty accessible



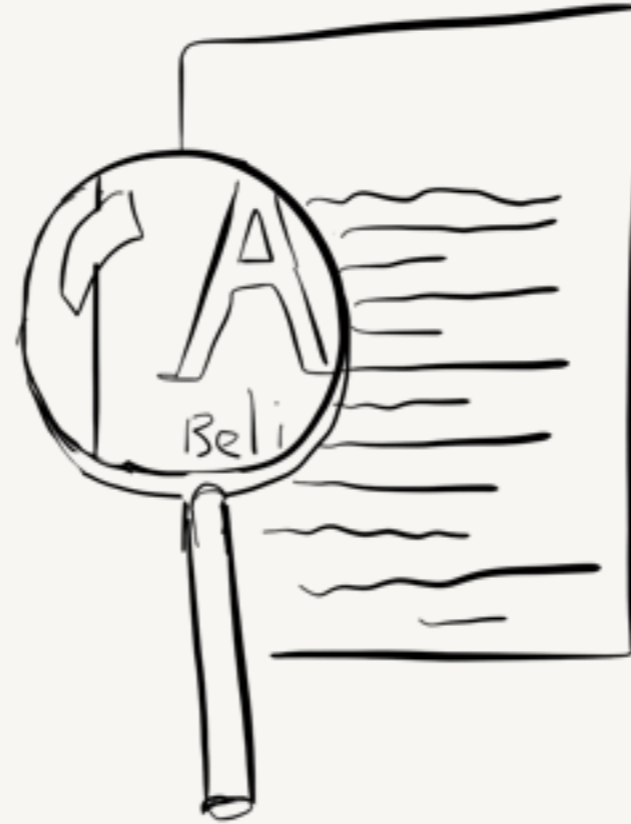
**News flash!**



Arrival of devices—*recalibrating* our thinking

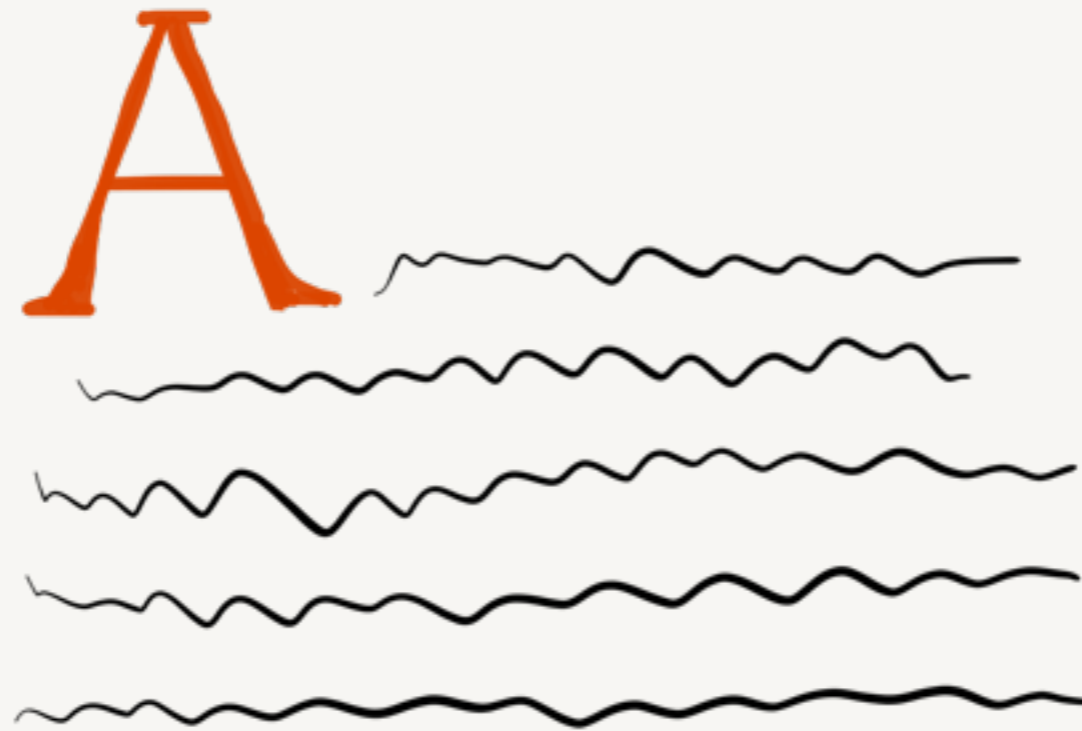


**Content is king**

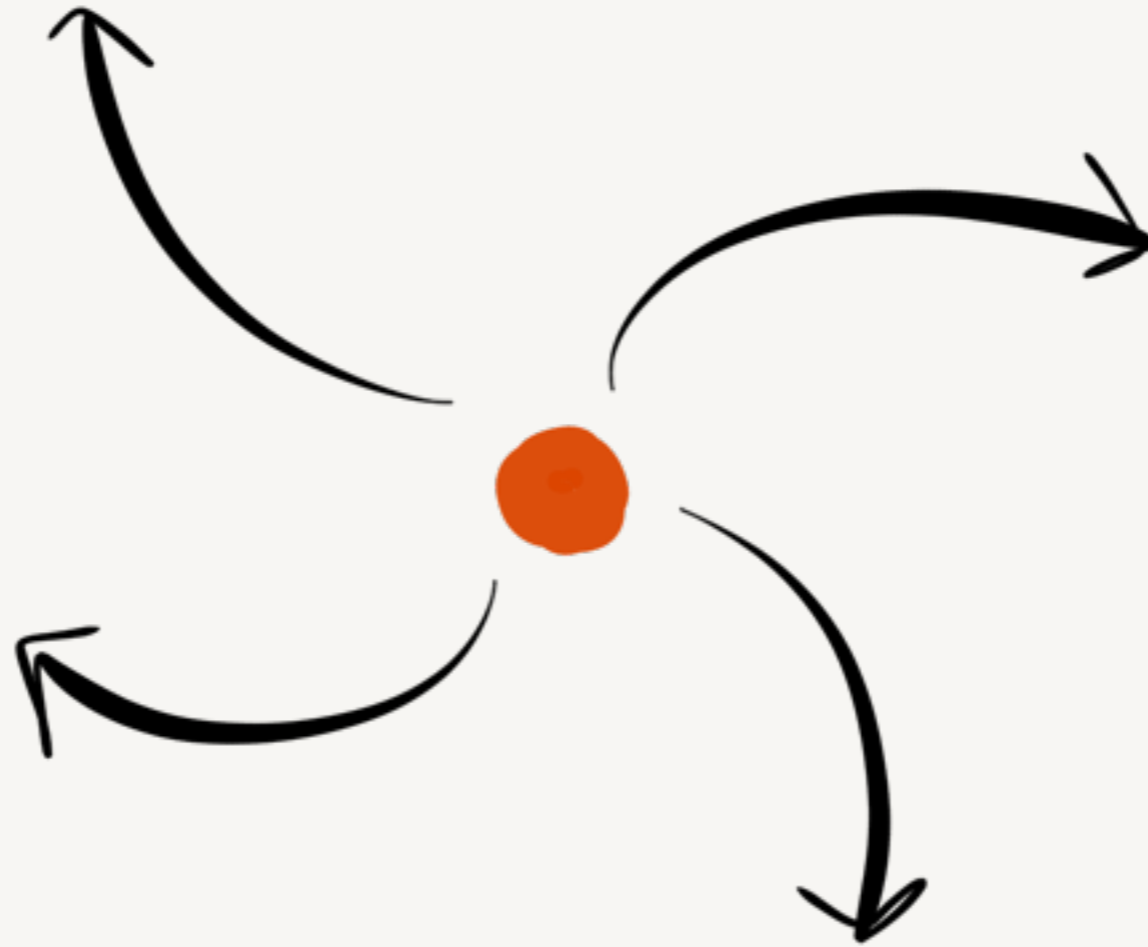


**Which shows in the HTML results**





Respect for content leads to good code



Baseline-first



Taking deliberate ***steps***

# Advocate mastery of authorship

1. ***Content*** obsession
2. Alternate/textual representations and ***fallbacks***
3. ***Semantic*** HTML5 and ***hierarchical*** expression
4. ***Best practices*** in element, attribute use
5. ***Light DOM*** structure
6. Mindfulness of ***source order***



**Semantic HTML is powerful**



Lovely code leads to accidental *experts*



Emphasize ***specifics***

# Ex: HTML lang attribute

language-dependent styling

appropriate hyphenation rules

```
<html lang="en">
```

search results relevance

aids translation software

accessibility

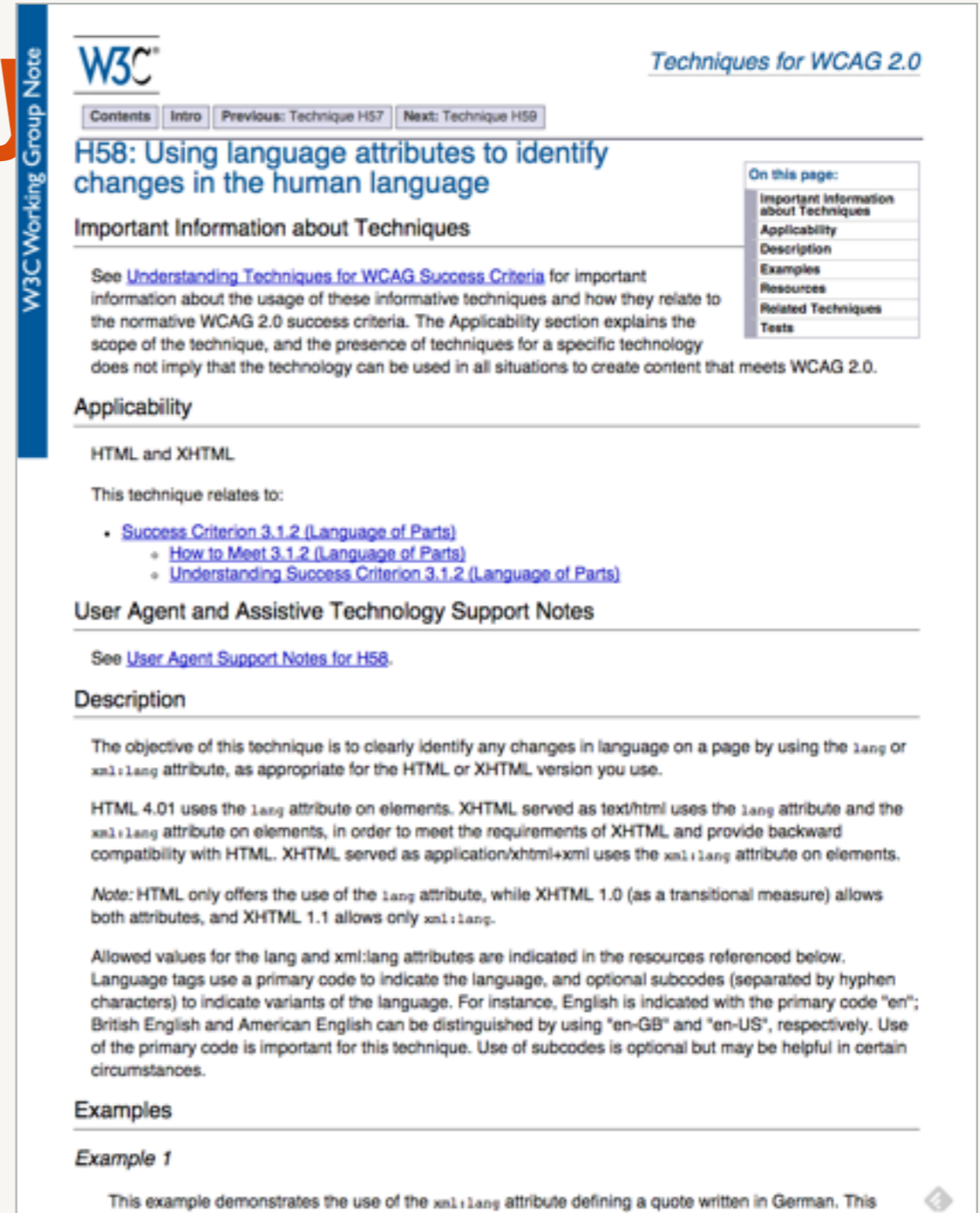


# Ex: <html> lang attribute

## accessibility

helps assistive technologies (AT) identify what language mode to use and how to handle content

required to meet the W3C Web Content Accessibility Guidelines (WCAG) 2.0



The screenshot shows a W3C Working Group Note page titled "H58: Using language attributes to identify changes in the human language". The page includes a navigation bar with "Contents", "Intro", "Previous: Technique H57", and "Next: Technique H59". A sidebar on the right lists "On this page:" with links to "Important Information about Techniques", "Applicability", "Description", "Examples", "Resources", "Related Techniques", and "Tests". The main content is divided into sections: "Important Information about Techniques" (with a link to "Understanding Techniques for WCAG Success Criteria"), "Applicability" (covering HTML and XHTML), "User Agent and Assistive Technology Support Notes" (with a link to "User Agent Support Notes for H58"), "Description" (explaining the objective and usage of the lang and xml:lang attributes), and "Examples" (starting with "Example 1").

# Establishing good for the *future*

“ The *usefulness of language tagging has increased* over recent years, as technology has progressed, and it will continue to increase as we go forward. In many cases, these applications may not be things you see as important when first developing your content, but may *grow in value* as time progresses.



**Make connections**

Avoiding the valley of  
despair



**Beware the Valley of Despair**



For more gains, more ***effort***



It's ***complicated***: now what?



**Moment of doom?**



W3C Home

Web Accessibility Initiative (WAI) Home
Getting Started
Designing for Inclusion
Guidelines & Techniques
Web Content (WCAG)
Authoring Tool (ATAG)
User Agent (UAAG)
» <b>WAI-ARIA (Rich Applications)</b>
• <a href="#">FAQ</a>
• <a href="#">WAI-ARIA 1.0</a>
• <a href="#">Primer</a>
• <a href="#">Authoring Practices</a>
• <a href="#">User Agent Implementation</a>
• <a href="#">Roadmap</a>
• <a href="#">CR Implementation</a>
Indie UI
Evaluation Language (EARL)
Research Topics
Technical Papers
Referencing & Linking
Development Process
Planning & Implementing
Evaluating Accessibility
Presentations & Tutorials
Getting Involved with WAI

*Discover new resources for people with disabilities, policy makers, managers, and you!*

Translations

## WAI-ARIA Overview

Quick links: [WAI-ARIA](#), [User Agent Implementation Guide](#), [FAQ](#)

See also [FAQ: What is the current status of WAI-ARIA development?](#)

### Introduction

WAI-ARIA, the Accessible Rich Internet Applications Suite, defines a way to make Web content and Web applications more accessible to people with disabilities. It especially helps with dynamic content and advanced user interface controls developed with Ajax, HTML, JavaScript, and related technologies. Currently certain functionality used in Web sites is not available to some users with disabilities, especially people who rely on screen readers and people who cannot use a mouse. WAI-ARIA addresses these accessibility challenges, for example, by defining new ways for functionality to be provided to assistive technology. With WAI-ARIA, make advanced Web applications accessible and usable to people with disabilities.

This page describes the problems that WAI-ARIA addresses, and introduces the WAI-ARIA suite of technical documents. Many of the terms used in this page—*Web content*, *user agents*, and *assistive technology*—are described in [Introduction to Web Accessibility](#) and [Essential Components of Web Accessibility](#). Additional information is available in:

- Blog post [Accessible Rich Internet Applications \(WAI-ARIA\) 1.0 is a W3C Recommendation](#)
- Press release: [W3C's Accessible Rich Internet Applications \(WAI-ARIA\) 1.0 Expands Accessibility of the Open Web Platform](#)
- [WAI-ARIA FAQ](#) answers questions such as: "What happens in current and older browsers when WAI-ARIA is implemented?" and "As a Web content developer, should I do with WAI-ARIA now?"

### Making Ajax and Related Technologies Accessible

Web sites are increasingly using more advanced and complex user interface controls, such as tree controls for Web site navigation like the example in Figure 1. To provide an accessible user experience to people with disabilities, assistive technologies need to be able to interact with these controls. However, the information that the assistive technologies need is not available with most current Web technologies.

Another example of an accessibility barrier is drag-and-drop functionality that is not available to users who use a keyboard only and cannot use a mouse. Even relatively simple Web sites can be difficult if they require an extensive amount of keystrokes to navigate with only a keyboard.

Many Web applications developed with Ajax (also known as AJAX), DHTML, and other technologies pose additional accessibility challenges. For example, if the content of a Web page changes in response to user actions or time- or event-based updates, that new content may not be available to some people, such as people who are blind or people with cognitive disabilities who use a screen reader.

WAI-ARIA addresses these accessibility challenges by defining how information about this functionality can be provided to assistive technology. With WAI-ARIA, an advanced Web application can be made accessible and usable to people with disabilities.

### Technical Solutions

More specifically, WAI-ARIA provides a framework for adding attributes to identify features for user interaction, how they relate to each other, and their current state. WAI-ARIA describes new navigation techniques to mark regions and common Web structures as menus, primary content, secondary content, banner information, and other types of Web structures. For example, with WAI-ARIA, developers can identify regions of pages and enable keyboard users to easily move among regions, rather than having to press Tab many times.

Page Contents

- ▾ [Introduction](#)
- ▾ [Making Ajax and Related Technologies Accessible](#)
- ▾ [The WAI-ARIA Suite Documents](#)
- ▾ [Versions 1.0, 1.1, future](#)
- ▾ [Who develops WAI-ARIA](#)

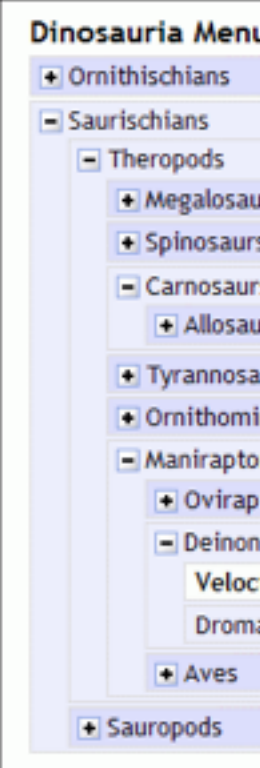


Figure 1: Tree control

## How to Meet WCAG 2.0

A customizable quick reference to Web Content Accessibility Guidelines 2.0 requirements (success criteria) and techniques

### Introduction

[\[Hide Introduction\]](#)

This web page can be used as a checklist for WCAG 2.0. It provides:

- **All of the requirements (called "success criteria")** from [Web Content Accessibility Guidelines \(WCAG\) 2.0](#)
- **Techniques** to meet the requirements, which are linked to pages with descriptions, code examples, browser and assistive technology support notes, and tests.
- **Failures** to avoid, which are linked to pages with descriptions, examples, and tests.
- **"Understanding" links** to pages that explain the intent of the guideline or success criterion, how it helps people with different disabilities, key terms, and resources.

You can customize what is included in this page by selecting from the [Customize this Quick Reference](#) section which Technologies, Levels of success criteria, and Sections of techniques you want to include.

For an introduction to WCAG, Techniques, and Understanding documents, see the [WCAG Overview](#).

Note that even content that conforms at the highest level (AAA) will not be accessible to individuals with all types, degrees, or combinations of disability, particularly in the cognitive language and learning areas. Authors are encouraged to seek relevant advice about current best practice to ensure that Web content is accessible, as far as possible, to this community.

### About the Techniques

For important information about the techniques, please see the [Understanding Techniques for WCAG Success Criteria](#) section of [Understanding WCAG 2.0](#).

**Note:** The basis for determining conformance to WCAG 2.0 is the success criteria, not the techniques. (The success criteria have 3-level numbering (0.0.0) and in this page they are followed by a link "Understanding Success Criterion".) All techniques are informative; that means they are not required. There may be [other techniques](#) besides the ones listed here.

### New Techniques and Comments

The *Techniques for WCAG 2.0* document is updated periodically, and anyone can [submit techniques](#) that will be considered for inclusion in an update. Please submit corrections, updates, or new information related to techniques, failures, or other WCAG documentation to the WCAG Working Group, per the [instructions for commenting](#).

---

## Table of Contents

[WCAG 2.0 Quick Reference List](#)

- 1.1 **Text Alternatives:** [Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, braille, speech, symbols or simpler language.](#)
- 1.2 **Time-based Media:** [Provide alternatives for time-based media.](#)
- 1.3 **Adaptable:** [Create content that can be presented in different ways \(for example simpler layout\) without losing information or structure.](#)
- 1.4 **Distinguishable:** [Make it easier for users to see and hear content including separating foreground from background.](#)

### Customize this Quick Reference

#### Technologies:

- Show HTML techniques and failures
- Show CSS techniques and failures
- Show SMIL techniques and failures
- Show Client-side Scripting techniques and failures
- Show Server-side Scripting techniques and failures
- Show Flash techniques and failures
- Show PDF techniques and failures
- Show Silverlight techniques and failures
- Show WAI-ARIA techniques and failures

#### Levels:

- Show Level A Success Criteria
- Show Level AA Success Criteria
- Show Level AAA Success Criteria

#### Sections:

- Show Sufficient Techniques and Failures
- Show Advisory Techniques

#### Save Settings Option:

- Save these settings (requires cookies)

[Customize with Settings Above](#)

## 6. Supported States and Properties

This section is *normative*.

### 6.1. Clarification of States versus Properties

The terms "states" and "properties" refer to similar features. Both provide specific information about an *object*, and both form part of the definition of the nature of *roles*. In this document, states and properties are both treated as aria-prefixed markup *attributes*. However, they are maintained conceptually distinct to clarify subtle differences in their meaning. One major difference is that the properties (such as [aria-labelledby](#)) are often less likely to change throughout the application life-cycle than the values of states (such as [aria-checked](#)) which may change frequently due to interaction. Note that the frequency of change difference is not a rule; a few properties, such as [aria-activedescendant](#), [aria-valuenow](#), and [aria-valuetext](#) are expected to change often. Because the distinction between states and properties is of little consequence to most web content authors, this specification refers to both "states" and "properties" simply as "attributes" whenever possible. See the definitions of *state* and *property* for more information.

### 6.2. Characteristics of States and Properties

States and properties have the characteristics described in the following sections.

#### 6.2.1. Related Concepts

Advisory information about features from this or other languages that correspond to this *state* or *property*. While the correspondence may not be exact, it is useful to help understand the intent of the state or property.

#### 6.2.2. Used in Roles

Advisory information about [roles](#) that use this *state* or *property*. This information is provided to help understand the appropriate usage of the state or property. Use of a given state or property is limited to the roles listed when used on roles other than those listed.

#### 6.2.3. Inherits into Roles

Advisory information about [roles](#) that inherit the *state* or *property* from an ancestor role.

#### 6.2.4. Value

Value type of the *state* or *property*. The value may be one of the following types:

**true/false**

Value representing either true or false, with a default "false" value.

**tristate**

Value representing true or false, with an intermediate "mixed" value. Default value is "false" unless otherwise specified.

**true/false/undefined**

Value representing true or false, with a default "undefined" value indicating the state or property is not relevant.

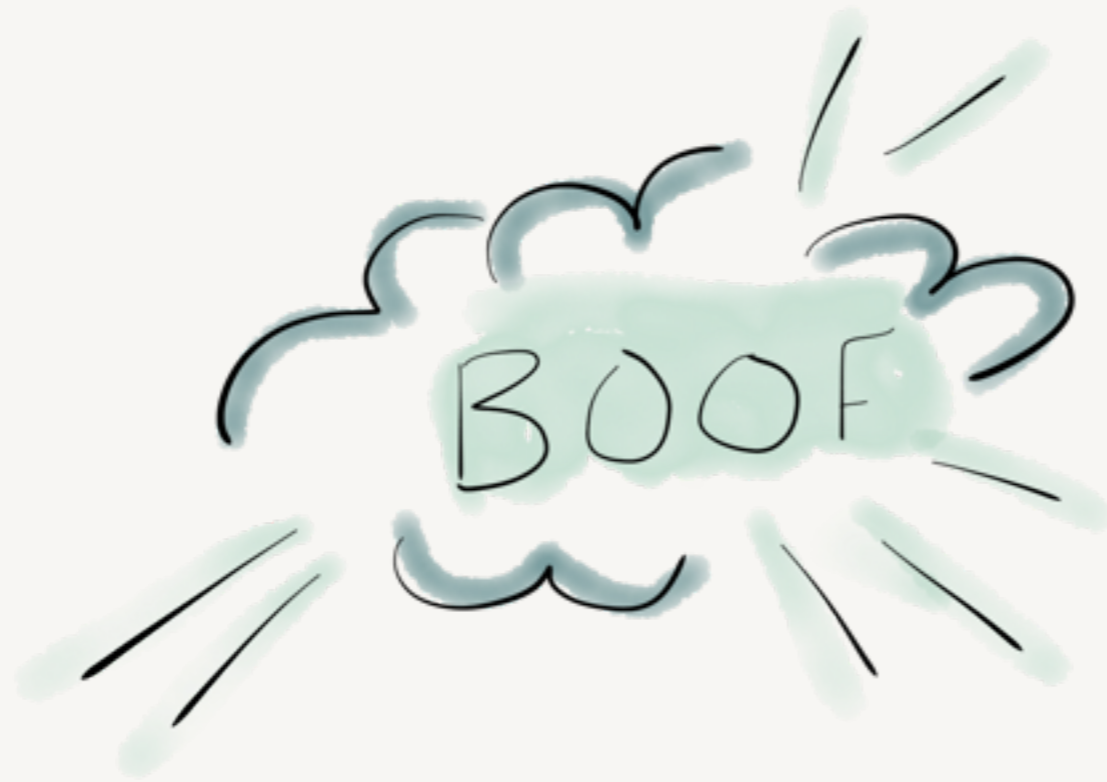
**ID reference**

Reference to the ID of another *element* in the same document

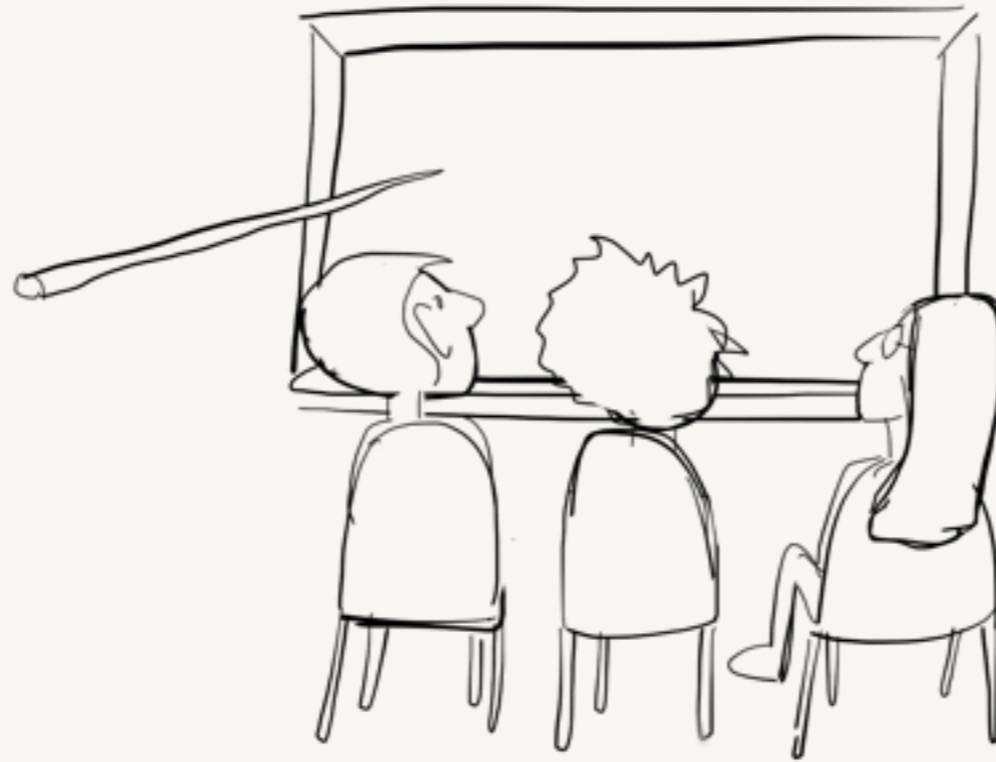
**ID reference list**

A list of one or more ID references.

**integer**



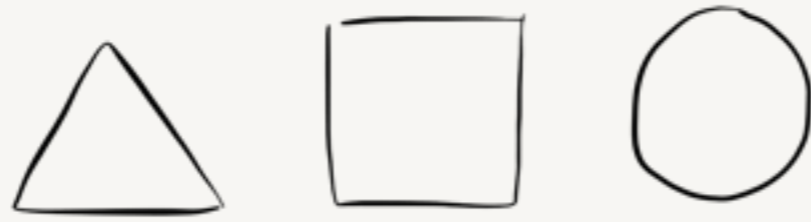
***Boom***...give up...



Our chance to ***educate***, advocate



Remaining mindful of ***constraints***



***Synthesize*** down to chunks



Not everyone is an accessibility *wizard*



Layering things on



Synthesizing... *what* now?

W3C

W3C

WAI-ARIA

Web Accessibility Initiative

Accessible Rich Interactive Applications Suite

Time to *break down* the problem



## Role Attribute 1.0

An attribute to support the role classification of elements

W3C Recommendation 28 March 2013

**This version:**

<http://www.w3.org/TR/2013/REC-role-attribute-20130328/>

**Latest published version:**

<http://www.w3.org/TR/role-attribute/>

**Latest editor's draft:**

<http://www.w3.org/WAI/PF/role-attribute/>

**Previous version:**

<http://www.w3.org/TR/2012/PR-role-attribute-20121213/>

**Editor:**

Shane McCarron, Applied Testing and Technology, Inc., [shane@aptest.com](mailto:shane@aptest.com)

**Authors:**

Ben Adida, [Creative Commons](#)  
Mark Birbeck, [webBackplane](#)  
Steven Pemberton, [CWI/W3C](#)  
T. V. Raman, [Google, Inc.](#)  
Richard Schwerdtfeger, [IBM Corporation](#)

Please refer to the [errata](#) for this document, which may include some normative corrections.

See also [translations](#).

The English version of this specification is the only normative version. Non-normative [translations](#) may also be available.

Copyright © 2006-2013 W3C® (MIT, ERCIM, Keio, Beihang). All Rights Reserved. W3C [liability](#), [trademark](#) and [document use](#) rules apply.

### Abstract

The Role Attribute defined in this specification allows the author to annotate markup languages with machine-extractable semantic information about the purpose of an element. Use cases include accessibility, device adaptation, server-side processing, and complex data description. This attribute can be integrated into any markup language. In particular, schema implementations are provided to facilitate with languages based upon XHTML Modularization [XHTML-MODULARIZATION11-2e].

The role attribute is necessary to support Accessible Rich Internet Applications (WAI-ARIA) [WAI-ARIA] to define roles in XML-based languages, when the languages do not define their own role attribute. Although this is the reason the Role Attribute is published by the Protocols and Formats Working Group, the attribute has more general use cases as well.

### Status of This Document

*This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the [W3C technical reports index](#) at <http://www.w3.org/TR/>.*

This is the Role Attribute [W3C Recommendation](#) by the [Protocols & Formats Working Group](#) of the [Web Accessibility Initiative](#). This document has been reviewed by W3C Members, by software developers, and by other W3C groups and interested parties, and is endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited from another document. W3C's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the

<http://www.w3.org/TR/role-attribute/>

# The HTML *role* attribute

“ The ***Role Attribute*** defined in this specification allows the author to ***annotate*** markup languages with machine-extractable ***semantic information about the purpose of an element***. Use cases include accessibility, device adaptation, server-side processing, and complex data description.

<http://www.w3.org/TR/role-attribute/>

## Why roles?

```
1 <header role="banner">  
2  
3 <div role="navigation">
```



Roles can assist anything that parses the DOM. Not just Assistive Technologies.

Roles have a larger...*role*

**role**



**role**

*abstract roles*

*document structure roles*

*landmark roles*

*widget roles*

# Breaking it down: *landmark roles*

abstract roles

document structure roles

*landmark roles*

widget roles

Landmarks help  
““ assistive technology  
(AT) users *orient*  
themselves to a page  
and help them  
*navigate easily to*  
*various sections* of a  
page.

# Breaking it down: *landmark roles*

abstract roles  
document structure roles  
*landmark roles*  
widget roles

application  
banner  
complementary  
contentinfo  
form  
main  
navigation  
search

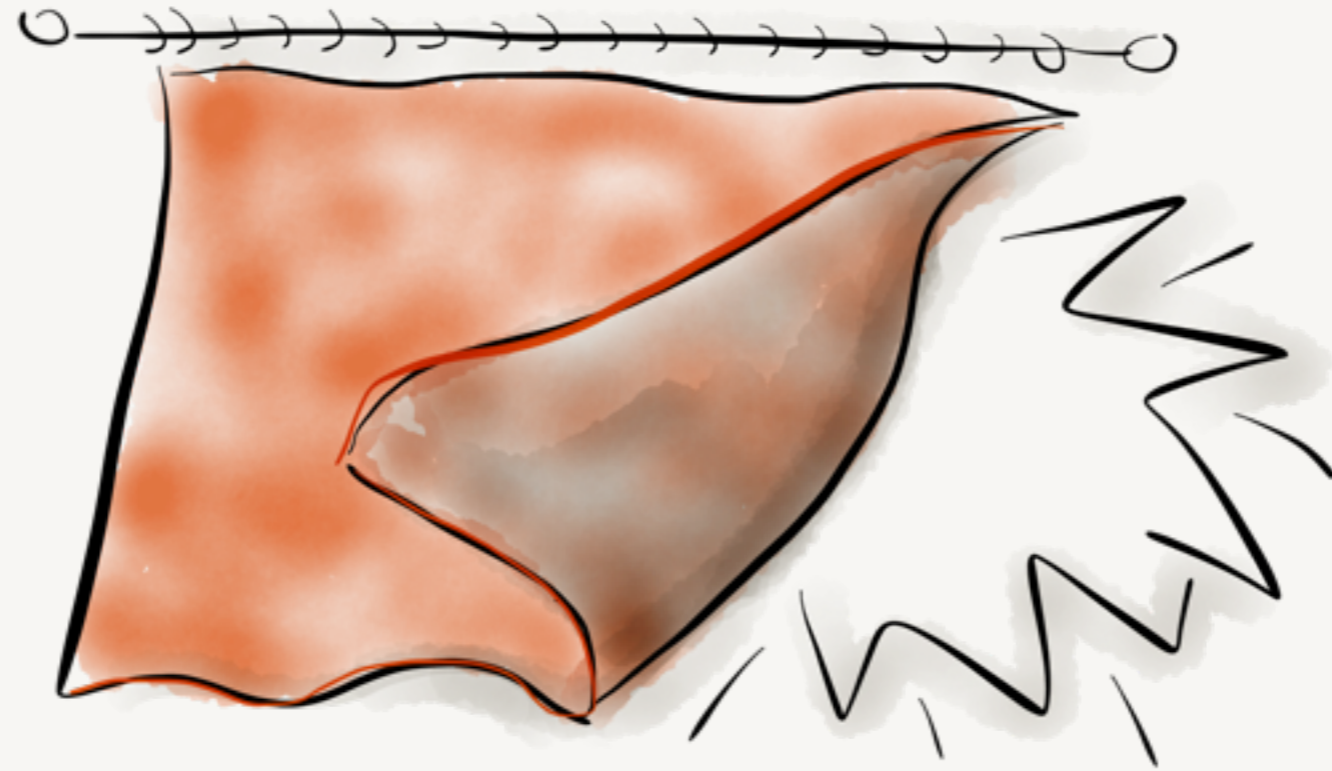
# Breaking it down: *landmark roles*



```
1 <header role="banner"></header>
2 <div role="navigation"></div>
3 <div role="main"></div>
4 <footer role="contentinfo"></footer>
5
```



Landmark roles: a small *step*



Pulling back the curtain on WAI-ARIA

**role**

*abstract roles*

*document structure roles*

*landmark roles*

*widget roles*



**role**

**role**  
**aria-\***

**aria-\***

*widget attributes*

*live region attributes*

*drag-and-drop attributes*

*relationship attributes*

**role**  
**aria-\***

WAI-ARIA *greatly oversimplified*

**role**

*abstract*

*document structure*

*landmark*

*widget*

**aria-\***

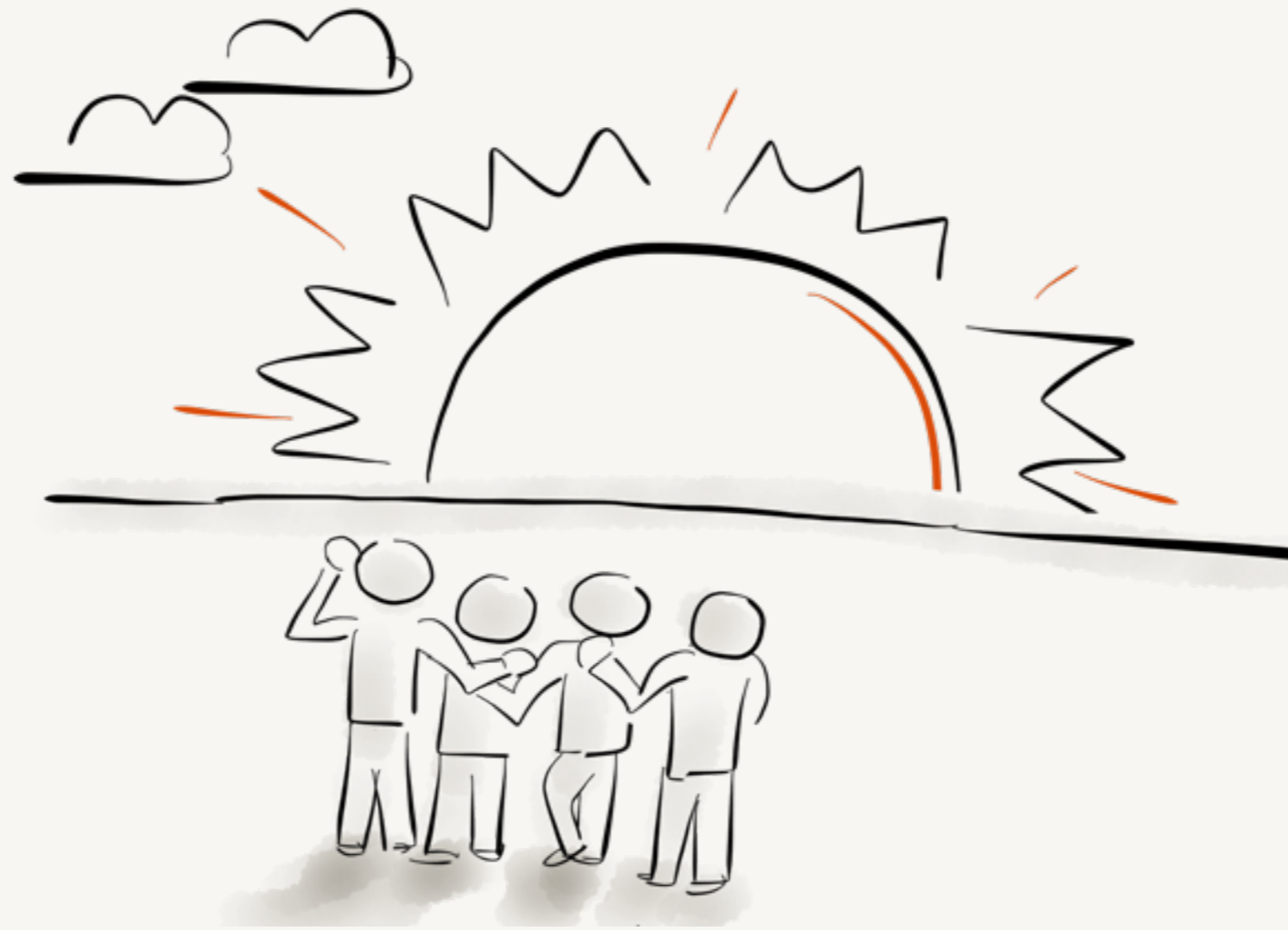
*widget*

*live region*

*drag-and-drop*

*relationship*

It takes a village



**Back to the bigger picture**

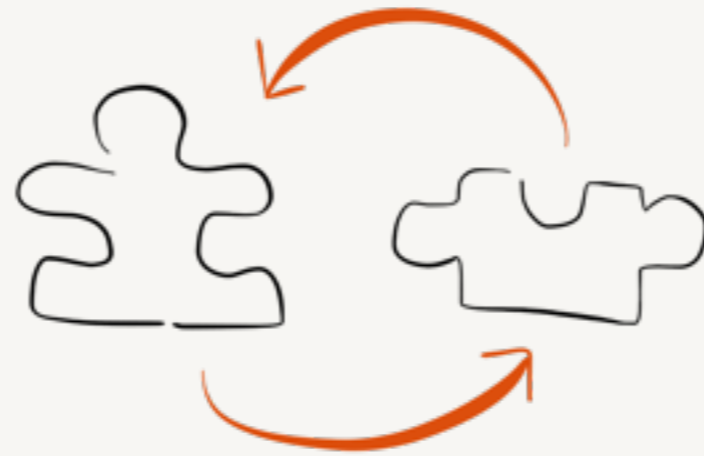


**Committing as a team**





Or else we lose our *path*



Accessibility must *fit in* with the plan



Accessibility must be a *part* of us



Has to be constantly reassessed



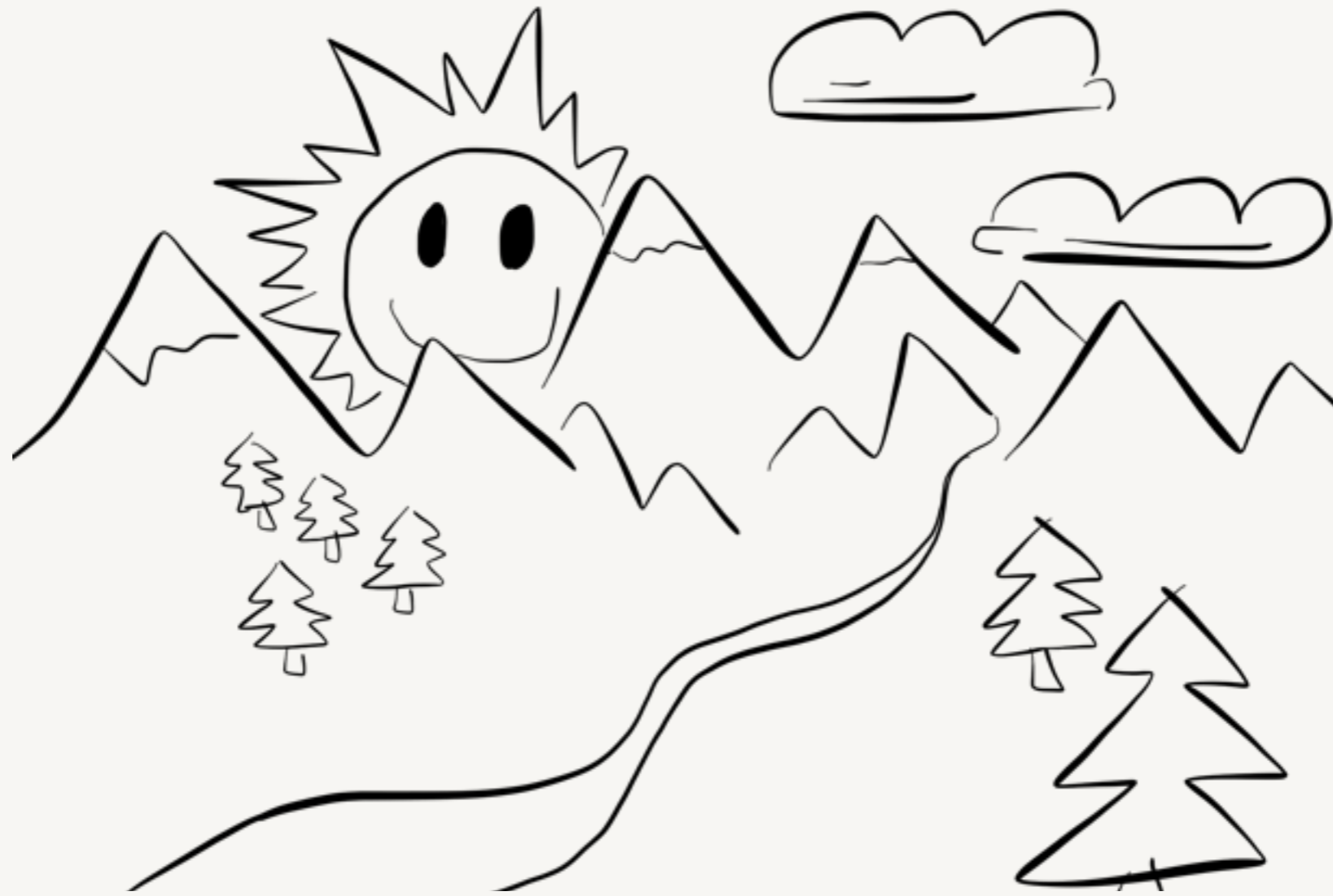
Teams need to ***connect*** about  
accessibility

# Presentation

```
1 [aria-expanded="true"] { }  
2  
3 [role="tablist"] { }
```

# Behavior

```
2 doSomething().then(function(result) {  
3     var myWidget = document.getElementById('my-widget');  
4     myWidget.setAttribute('aria-expanded', true);  
5 });  
6  
7 $(' [role="tablist"] ').doSomethingToChangeBehavior();
```



**Our side trip is complete**

# Gathering Steam



**role**  
**aria-\***

WAI-ARIA *greatly oversimplified*

**role**

*abstract*

*document structure*

*landmark*

*widget*

**aria-\***

*widget*

*live region*

*drag-and-drop*

*relationship*

Each attribute is a state *or* a property

**aria-\***

*widget*

*live region*

*drag-and-drop*

*relationship*

# From an HTML *element's viewpoint*

**role**

abstract  
document structure  
landmark  
widget

declares who I am

**aria-\***

widget  
live region  
drag-and-drop  
relationship

property describes a detail about me

state describes what I'm doing right now

# From an HTML *element's viewpoint*

**role**

abstract  
document structure  
landmark  
widget

never changes

**aria-\***

widget  
live region  
drag-and-drop  
relationship

property

changes rarely

state

may change often

```
1 <div class="dropdown">
2   <a id="dLabel" data-target="#" href="http://example.com"
3     data-toggle="dropdown" aria-haspopup="true"
4     role="button" aria-expanded="false">
5     Dropdown trigger
6   </a>
7
8   <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
9     ...
10  </ul>
11 </div>
```

<http://getbootstrap.com/javascript/#dropdowns>

# Bootstrap drop-down example

```
1 <a id="dLabel"  
2   href="http://example.com"  
3   data-target="#"  
4   data-toggle="dropdown"  
5   role="button"  
6   aria-haspopup="true"  
7   aria-expanded="false">  
8     Dropdown trigger  
9 </a>
```

Focus on the `<a>` element

**role**

category: widget

**attribute (property)**

category: widget

**attribute (state)**

category: widget

```
role="button"  
aria-haspopup="true"  
aria-expanded="false"
```

Zoom even closer



**role**  
category: widget

I am like a button!

**attribute (property)**  
category: widget

```
role="button"  
aria-haspopup="true"  
aria-expanded="false"
```

**attribute (state)**  
category: widget

Zoom even closer

**role**  
category: widget

**attribute (property)**  
category: widget

**attribute (state)**  
category: widget

I (unchangingly)  
have a popup!

```
role="button"  
aria-haspopup="true"  
aria-expanded="false"
```

Zoom even closer

**role**  
category: widget

**attribute (property)**  
category: widget

**attribute (state)**  
category: widget

Right now, I am  
not expanded.

```
role="button"  
aria-haspopup="true"  
aria-expanded="false"
```

Zoom even closer



**Making meaning clearer**

# Breaking it down: *widget roles*



# Breaking it down: *widget roles*

abstract roles  
document structure roles  
landmark roles  
*widget roles*

## *standalone* *composite*

alertdialog	combobox
button	grid
checkbox	listbox
dialog	menu
gridcell	menubar
link	radiogroup
log	tablist
marquee	tree
menuitem	treegrid
menuitemcheckbox	
menuitemradio	
option	
progressbar	
radio	
scrollbar	
slider	
spinbutton	
status	
tab	
tabpanel	
textbox	
timer	
tooltip	
treeitem	

# Breaking it down: *widget roles*

et roles  
re roles  
k roles  
*widget roles*

## *standalone*

alert	option
alertdialog	progressbar
button	radio
checkbox	scrollbar
dialog	slider
gridcell	spinbutton
link	status
log	tab
marquee	tabpanel
menuitem	textbox
menuitemcheckbox	timer
menuitemradio	tooltip
	treeitem

# Breaking it down: *widget roles*

et roles  
re roles  
k roles  
*widget roles*

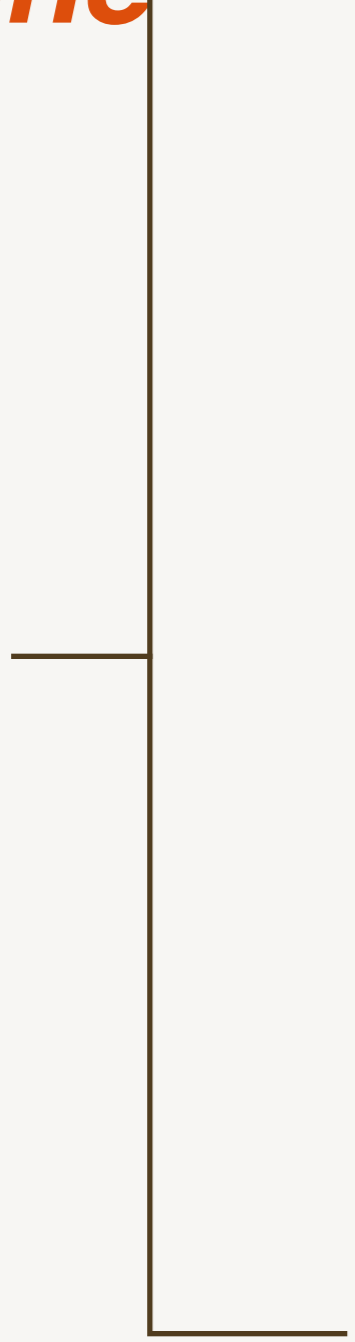
## *standalone*

alert	option
alertdialog	progressbar
<u>button</u>	radio
checkbox	scrollbar
dialog	slider
gridcell	spinbutton
link	status
log	tab
marquee	tabpanel
menuitem	textbox
menuitemcheckbox	timer
menuitemradio	tooltip
	treeitem



# Breaking it down: *widget roles*

*ndalone* — widget > command > *button*



# Breaking it down: *widget roles*

widget > command > *button*

supported

*aria-expanded (state)*

aria-pressed (state)

inherited

aria-atomic

aria-busy (state)

aria-controls

aria-describedby

aria-disabled (state)

aria-dropeffect

aria-flowto

aria-grabbed (state)

*aria-haspopup*

aria-hidden (state)

aria-invalid (state)

aria-label

aria-labelledby

aria-live

aria-owns

aria-relevant

*alone*

`<div role="form">`

`<a role="button">`

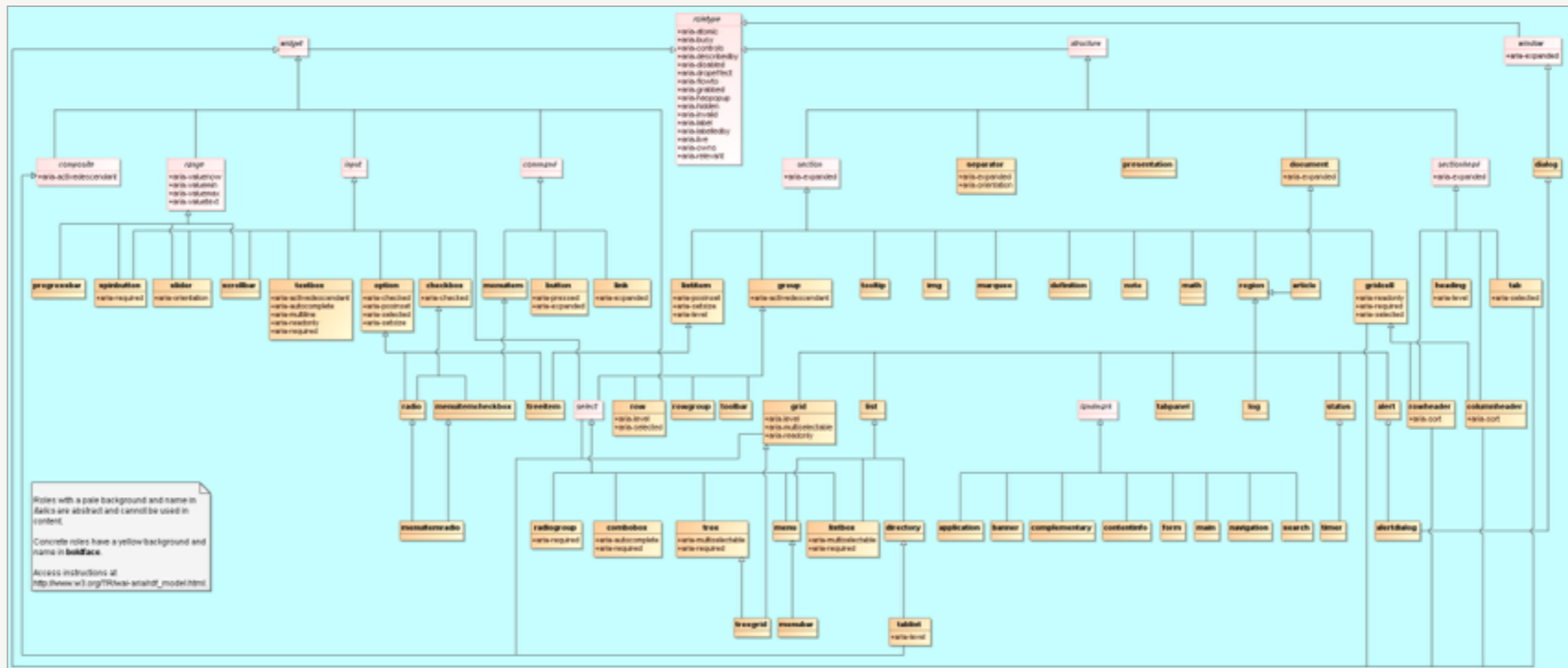
`<form>`

`<button>`

Or you could just use the correct element

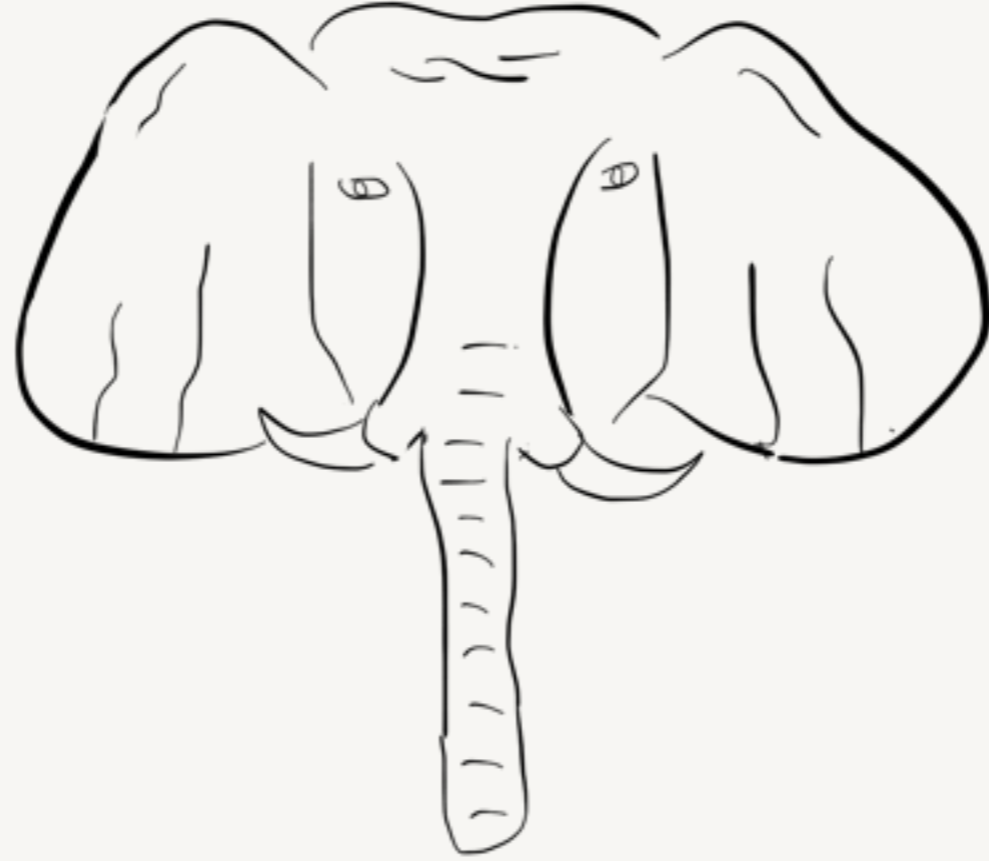
`<div role="tablist">`

`<div role="radiogroup">`



[http://www.w3.org/TR/wai-aria/rdf\\_model.png](http://www.w3.org/TR/wai-aria/rdf_model.png)

Widget role-attribute taxonomy complex



**It's like a wild safari!**

Entering  
the danger zone





**Lyza Danger Gardner**

@lyzadanger

Casual question for HTML builders: How thorough and consistent would you say your knowledge and use of the `role` attribute is?

I decided to ask some ***questions***



**Daniel Martínez**  
@Wakkos

@lyzadanger basic! Enough to say when to use “main, banner, contentinfo” and those common roles.



**Sam Nabi**  
@samnabi



Follow

@lyzadanger I know I should kinda use it on forms? Really not sure what it does though.



**Dan Mindru**  
@mindrudan



Follow

@lyzadanger not that thorough in my case. I am afraid of what it does and only use it with confidence as role="menu"



**Andy Davies**  
@AndyDavies

@lyzadanger I'm rubbish at it - could really do with a good guide to help me out



Is it that people *don't care?*



People are ***overwhelmed***



**Again at the valley of despair**

```
<progress value="0" max="100">0% complete</progress>
```

Making <progress>

```
var progressBar = document.getElementById("progress-bar");

// Check to see if the browser supports the HTML5 <progress> tag.
var supportsHTML5Progress = (typeof (HTMLProgressElement) !== "undefined");

function setupProgress() {
  if (!supportsHTML5Progress) {
    // HTML5 <progress> isn't supported in this browser, so we need to add
    // ARIA roles and states to the element.
    progressBar.setAttribute("role", "progressbar");
    progressBar.setAttribute("aria-valuemin", 0);
    progressBar.setAttribute("aria-valuemax", 100);
  }
}

function updateProgress(percentComplete) {
  if (!supportsHTML5Progress) {
    // HTML5 <progress> isn't supported by this browser,
    // so we need to update the aria-valuenow attribute
    progressBar.setAttribute("aria-valuenow", percentComplete);
  } else {
    // HTML5 <progress> is supported, so update the value attribute instead.
    progressBar.setAttribute("value", percentComplete);
  }

  progressBar.textContent = percentComplete + "% complete";
}

function initDemo() {
  setupProgress(); // Setup the progress bar.

  // Bind a click handler to the button, which will update the progress bar to 75%.
  document.getElementById("update-button").addEventListener("click", function (e) {
    updateProgress(75);
    e.preventDefault();
  }, false);
}

initDemo();
```

All this JS for a <progress> widget?

# Javascript Source Code

[Hide Javascript Source Code: alertdialog1.js](#)

```
<script type="text/javascript">
$(document).ready(function () {
  var guess1 = new guess(1, 10, 'guess1', 'guess1_text', 'guess1_check', 'guess1_again', 'alert1');
});

//
// keyCode is an object that defines keycodes for the key handlers
//
function keyCode () {
  // Define values for keycodes
  this.tab = 9;
  this.enter = 13;
  this.esc = 27;
  this.space = 32;
}

//
// alertDlg() is a class to implement a modal alert dialog
//
// @param (alert_id string) alert_id is the id of the dialog to create
//
// @param (game_id string) game_id is the id to attach the dialog to
//
// @return N/A
//
function alertDlg(alert_id, game_id) {
  var dlg = '<div id="' + alert_id + '" role="alertdialog" tabindex="-1" aria-hidden="true" aria-labeledby="' +
    alert_id + ' title"><p id="' + alert_id + ' title">Alert Box</p><p id="' +
    alert_id + ' message">No Message</p><input id="' +
    alert_id + ' close" type="button" value="Close" /></div>';

  // append the dialog to the document
  $('div#' + game_id).append(dlg);

  // Define the object properties
  this.$dlg = $('#' + alert_id); // the object pointer of the dialog
  this.$game = $('#' + game_id); // the object pointer of the containing div for the game
  this.$msg = $('#' + alert_id + ' message'); // the object pointer of the alert message area
  this.$button = $('#' + alert_id + ' close'); // the object pointer of the alert close button
  this.$focus; // the object pointer of a page element to give focus to on dialog dismissal

  this.keys = new keyCode();

  // bind handlers
  this.bindHandlers();
} // end alertDlg constructor

//
// showMsg() is a member function to set the message text of the alertDlg
//
// @param (msg string) msg is the message to display in the dialog box.
//
// @param (focusId string) focusId is the id of the element to give focus to when the dialog is dismissed.
//
// @return N/A
//
alertDlg.prototype.showMsg = function (msg, $focus) {

  // Store the focus ID
  this.$focus = $focus;

  // Set the message text
  this.$msg.html(msg);

  // Show the dialog
  this.showDlg();
} // end showMsg()

//
// bindHandlers() is a member function to bind event handlers to the modal alert dialog
//
// @return N/A
//
alertDlg.prototype.bindHandlers = function () {

  var thisObj = this; // store the this pointer

  // bind a keydown handler
  this.$dlg.keydown(function(e) {
    return thisObj.handleDlgKeyDown(e);
  });

  // bind a keypress handler
  this.$dlg.keypress(function(e) {
    return thisObj.handleDlgKeyPress(e);
  });

  // bind a click handler
  this.$button.click(function(e) {
    return thisObj.handleCloseClick(e);
  });
} // end bindHandlers()

//
// handleDlgKeyDown() is a member function to process keydown events for the alert dialog
```





# So many specs and abstract documents! [\[contents\]](#)

## WAI-ARIA 1.0 Authoring Practices

An author's guide to understanding and implementing Accessible Rich Internet Applications

W3C Working Draft 7 March 2013

**This version:**

<http://www.w3.org/TR/2013/WD-wai-aria-practices-20130307/>

**Latest version:**

<http://www.w3.org/TR/wai-aria-practices/>

**Previous version:**

<http://www.w3.org/TR/2010/WD-wai-aria-practices-20100916/>

**Editors:**

Joseph Scheuhammer, Invited Expert

Michael Cooper, W3C

**Previous Editors:**

Lisa Pappas, Society for Technical Communication

Richard Schwerdtfeger, IBM

Copyright © 2007 - 2013 W3C® (MIT, ERCIM, Keio, Beihang), All Rights Reserved. W3C [liability](#), [trademark](#) and [document use](#) rules apply.

## Abstract

This document provides readers with an understanding of how to use [WAI-ARIA \[ARIA\]](#) to create accessible rich internet applications. It describes considerations that might be addressed in a technical specification alone and recommends approaches to make widgets, navigation, and behaviors accessible using WAI-ARIA roles, states, and properties. This document is directed at authors of user agent and assistive technology. The guidance is also useful for user agent and assistive technology developers. This document is part of the WAI-ARIA suite described in the [WAI-ARIA Overview](#).

## Status of This Document

*This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest version of the [W3C technical reports index](#) at <http://www.w3.org/TR/>.*

This is a Public Working Draft by the [Protocols & Formats Working Group](#) of the [Web Accessibility Initiative](#). It supports the [Accessible Rich Internet Applications \(WAI-ARIA\)](#) specification. It includes examples beyond what would be appropriate to a technical specification but which are important to understand the specification. This version incorporates changes made in the previous version and issues identified by the Working Group in the course of its general work on WAI-ARIA. This snapshot is being published to bring the public Working Draft up to date with the prior publication, but it does not address all issues known to date. A [history of changes to WAI-ARIA 1.0 Authoring Practices](#) is available. Refer to the [summary of actions taken since the previous draft](#) and the [issue disposition report for the previous draft](#).

Feedback on the information provided here is essential to the ultimate success of Rich Internet Applications that afford full access to their information and operations. The PH

```

<script>
...
function optionKeyEvent(event)
{
var tb = event.target;
var buttonid;

DOM_VK_ENTER = 13;
// Partial sample code for processing arrow keys

if (event.type == "keydown") {
  if (event.altKey) {
    return true; // Browser should use this, the menu view doesn't need alt-modified keys
  }
  // XXX Implement circular keyboard navigation within the toolbar buttons

  if (event.keyCode == DOM_VK_ENTER) {
    ExecuteButtonAction(getCurrentButtonID()); // This is an author defined function
  }
  else if (event.keyCode == event.DOM_VK_RIGHT) {
    // Change the active toolbar button to the one to the right (circular) by
    var buttonid = getNextButtonID(); // This is an author defined function
    tb.setAttribute("aria-activedescendant", buttonid);
  }
  else if (event.keyCode == event.DOM_VK_LEFT) {
    // Change the active toolbar button to the one to the left (circular) by
    var buttonid = getPrevButtonID(); // This is an author defined function
    tb.setAttribute("aria-activedescendant", buttonid);
  }
  else {
    return true;
  }
  return false;
}
else if (event.type == "keypress") {
...
}
}
</script>

```





**Scott Jehl**

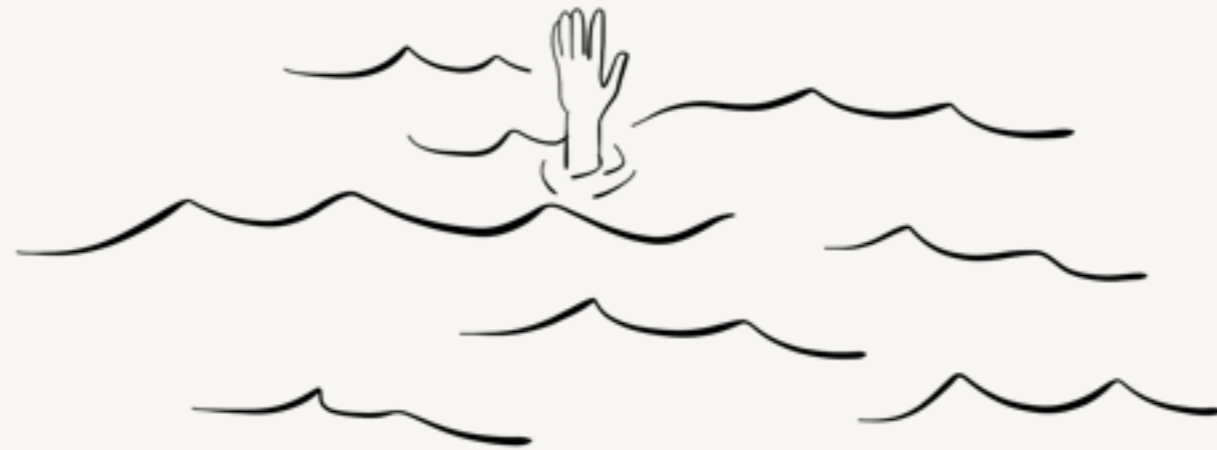
@scottjehl



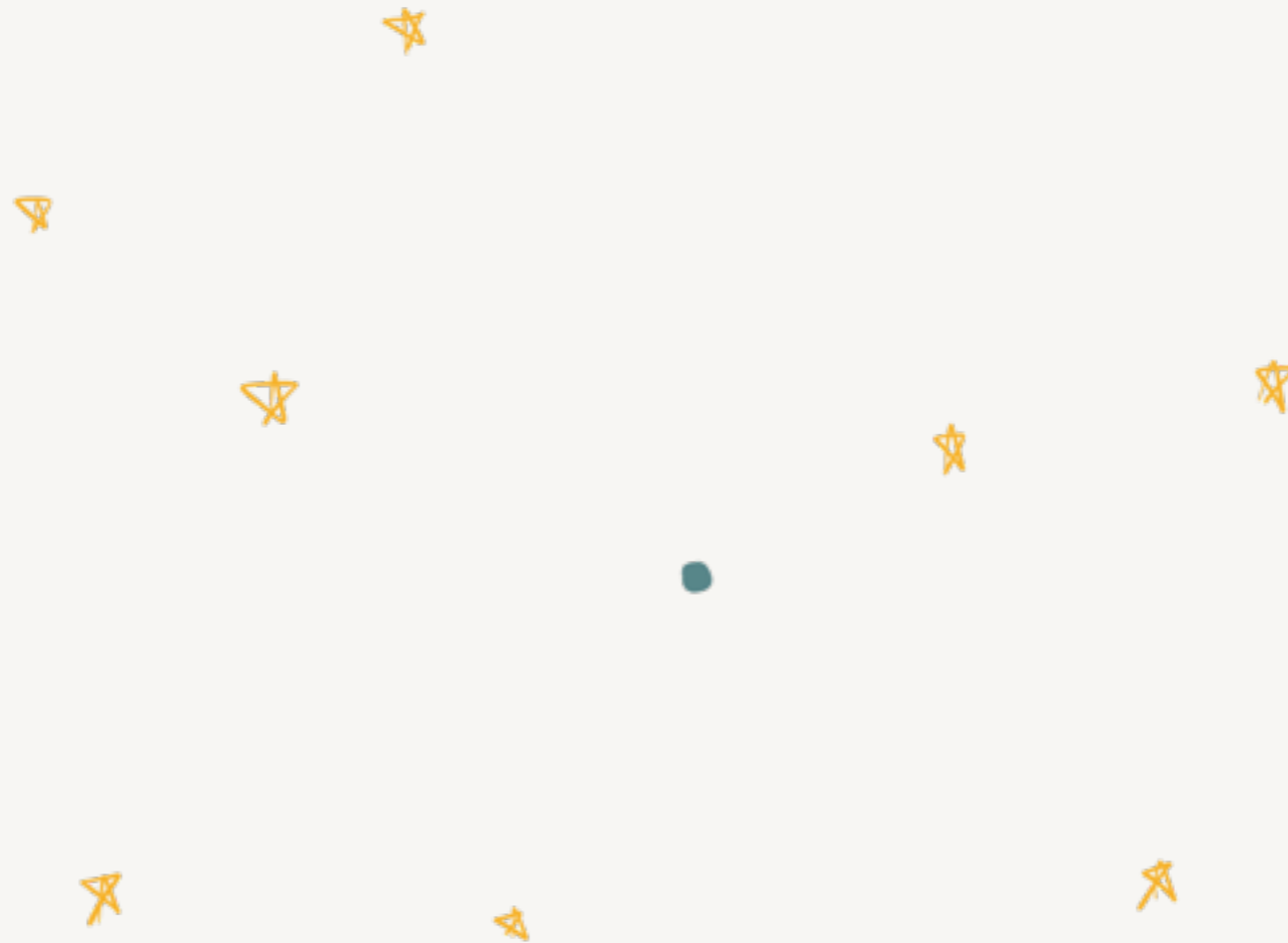
Following

@davatron5000 Every time I finish ARIA-ing a dynamic component I think... that was too hard to figure out and test. It's rough.  
+@lyzadanger

Oh, dear



**We're drowning**



There is a void



**Difficult decisions**



patrick h. lauke

@patrick\_h\_lauke



Following

@lyzadanger even just roles. adding role="menu" without the appropriate role="menuitem" children and proper kbd handling for instance

```
1 <div class="dropdown">
2   <a id="dLabel" data-
3     data-toggle="drop
4     role="button" aria-expanded="false"
5     Dropdown trigger
6   </a>
```

```
8 <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
9   ...
10 </ul>
```

```
11 </div>
```

No menuitems? You'll break it

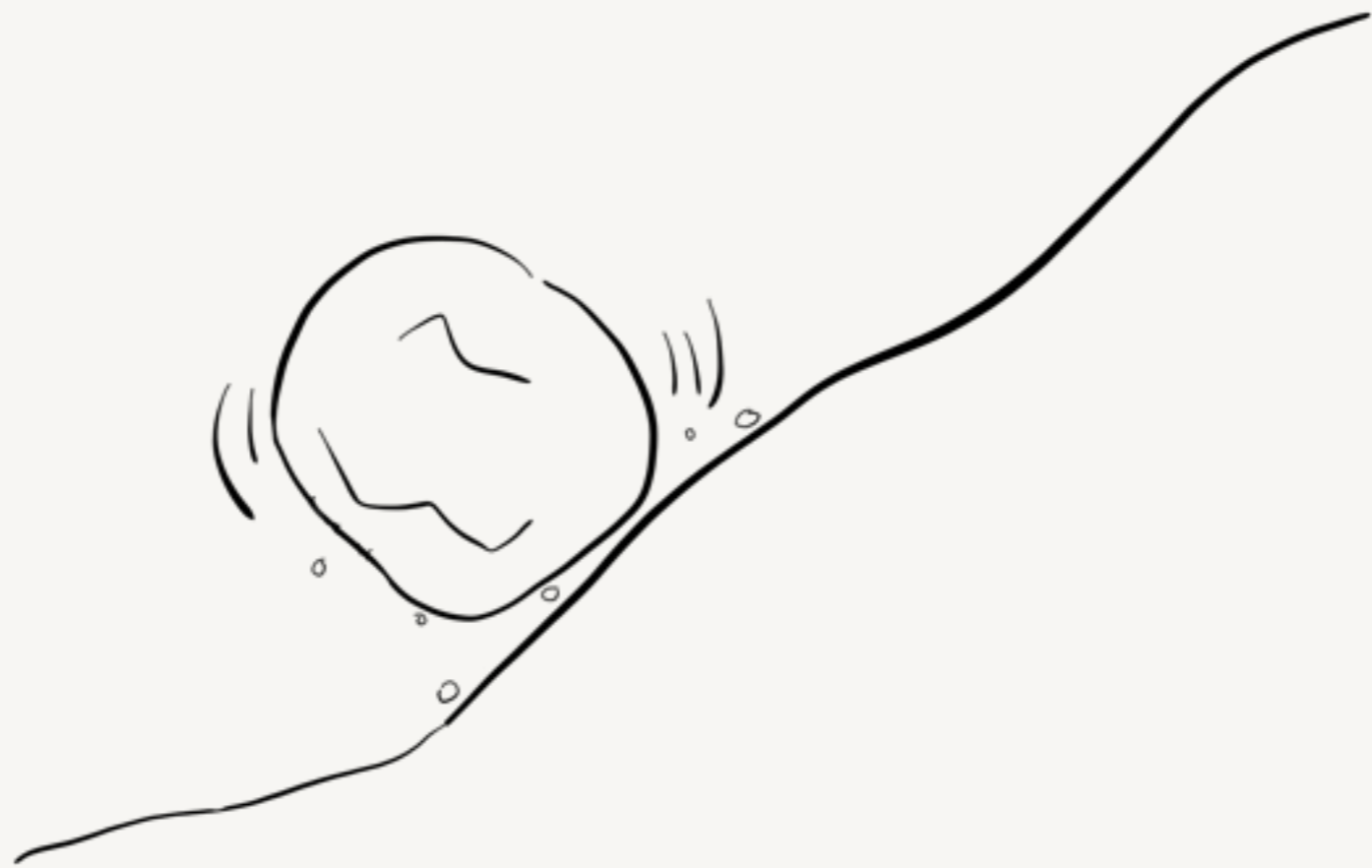
Oh, dear

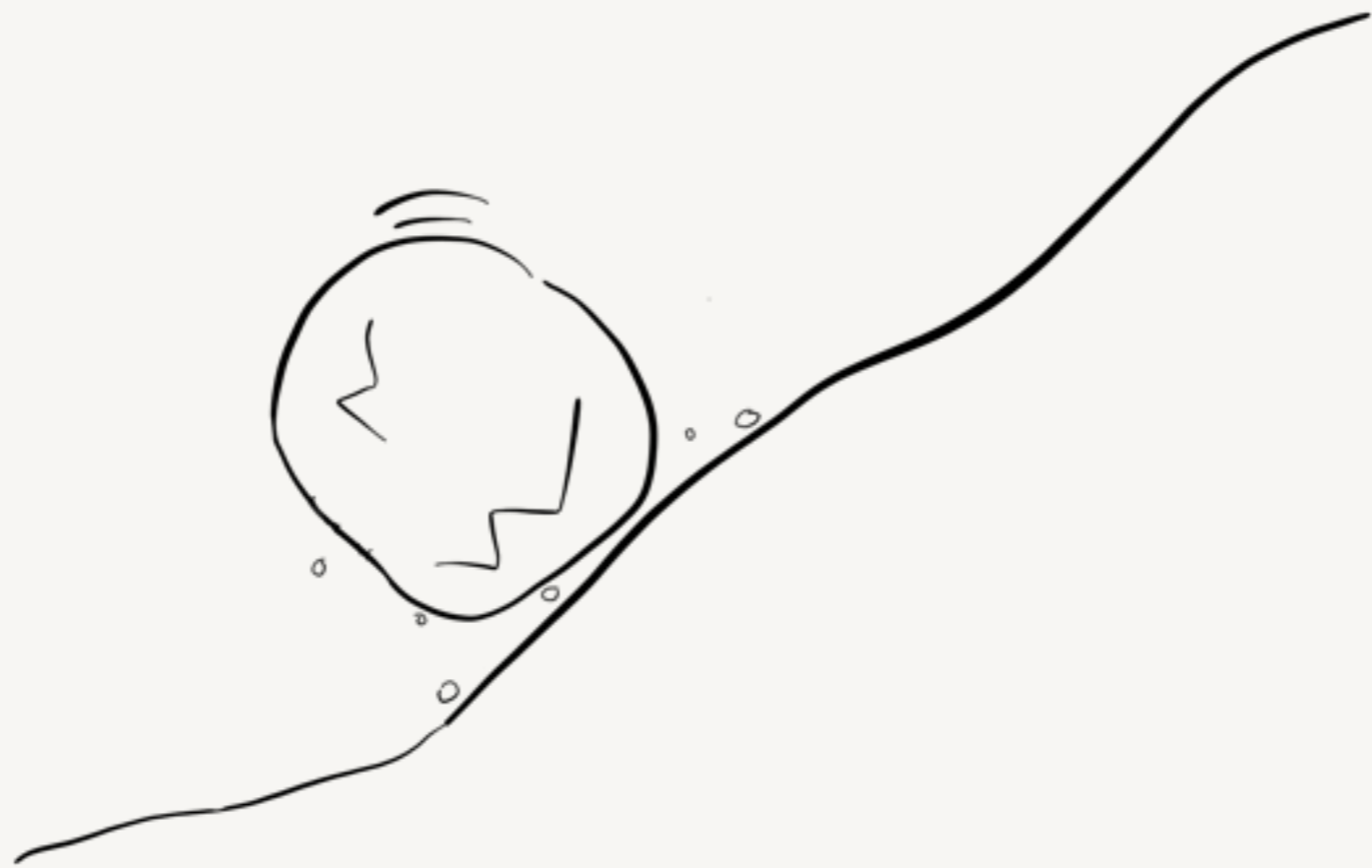


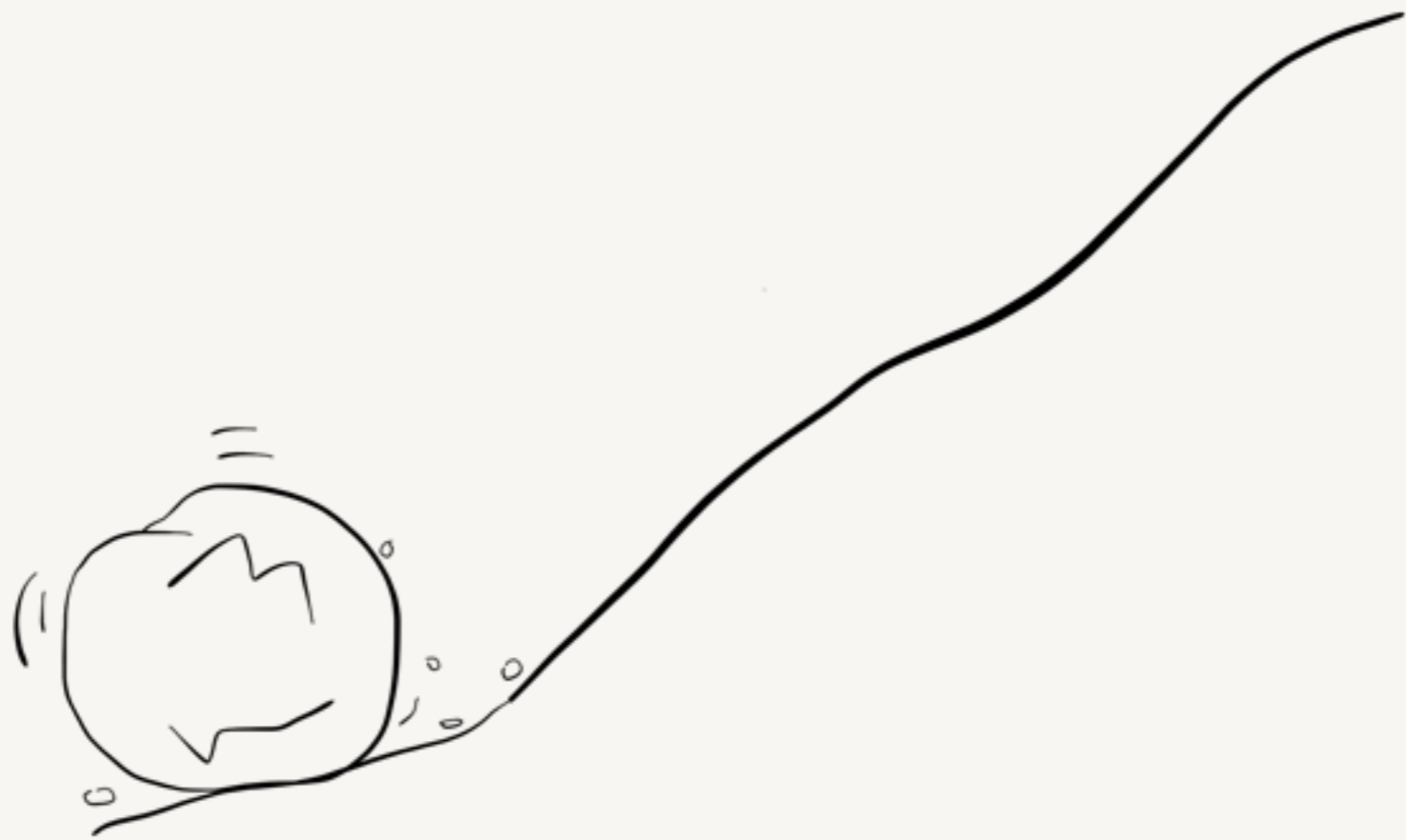


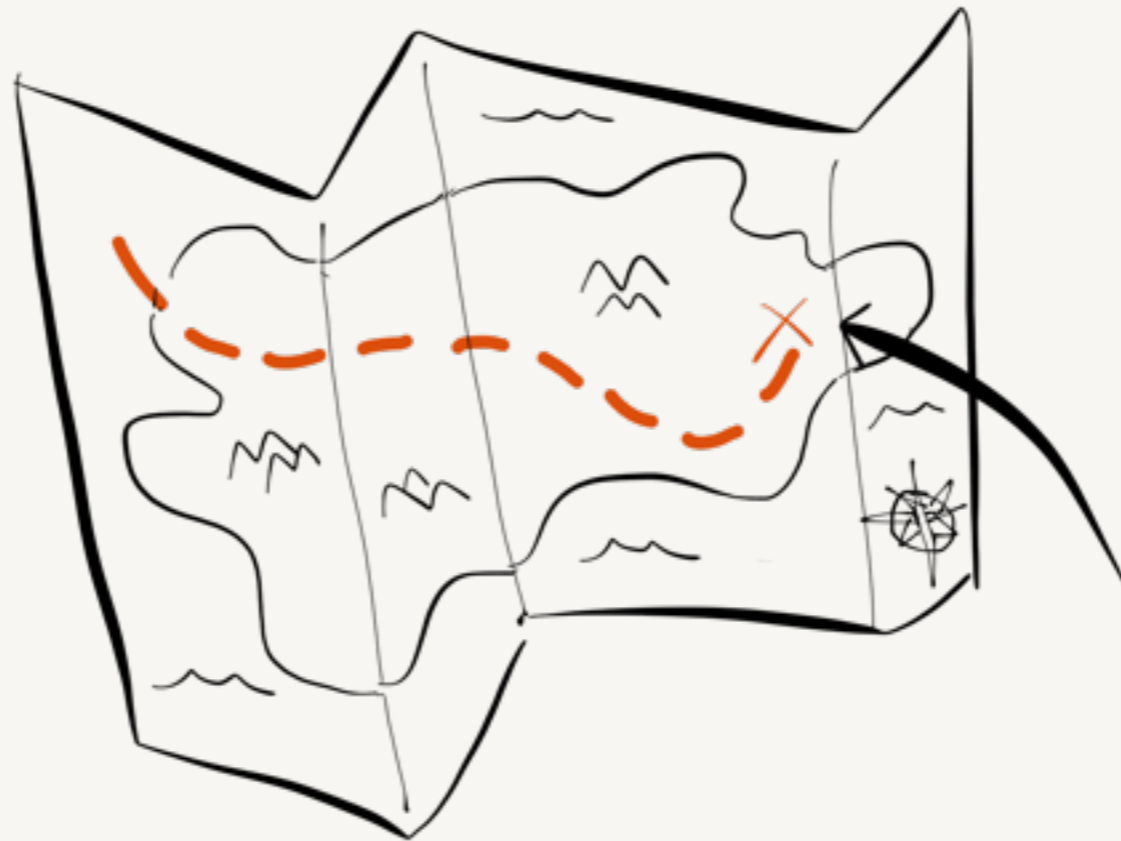
The challenge seems too hard





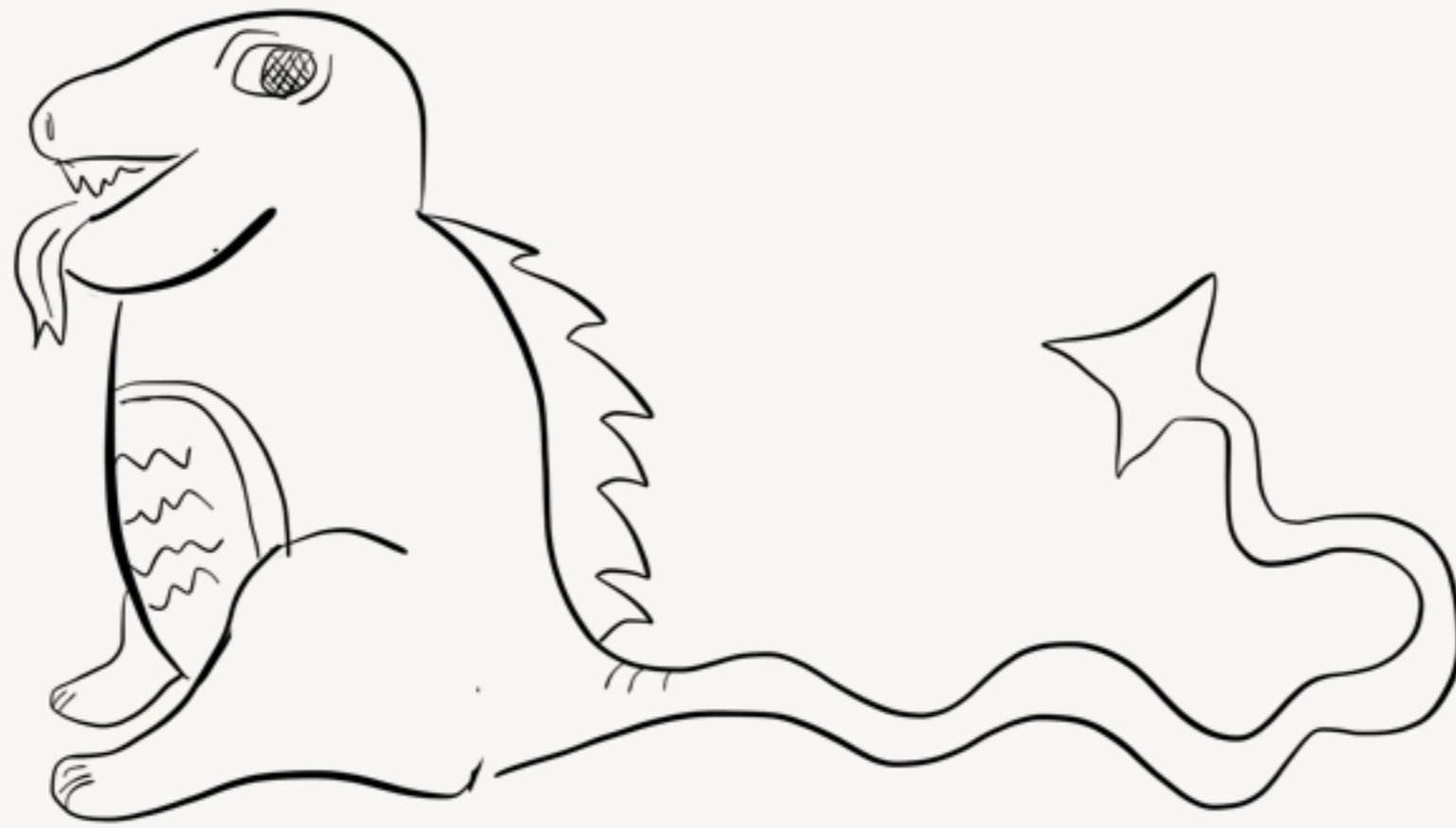






Should we give up?

Returning to safety

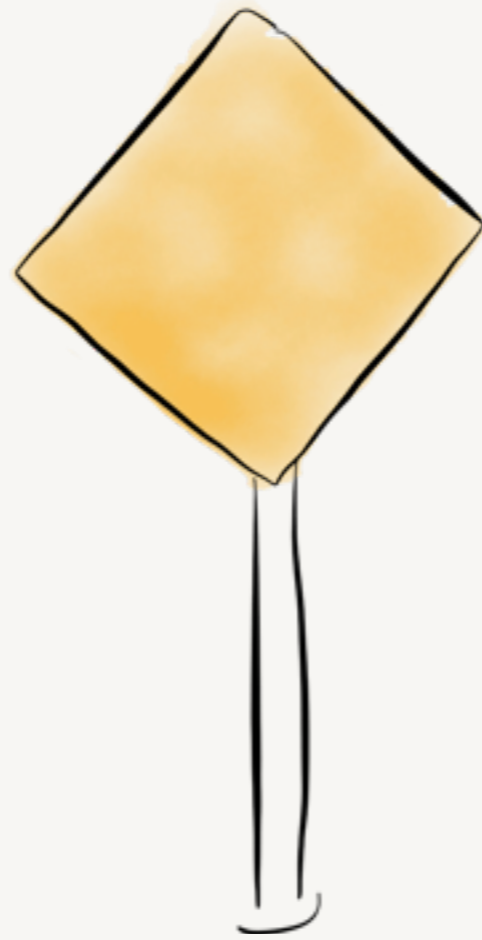


**WAI-ARIA can be scary**





**Making it less scary**



**Building a safety zone**

# Best practices to polish

Content and hierarchy

HTML authorship

Design considerations



# Safety zone

Content and hierarchy

HTML authorship

Design considerations

***Polishing best practices***

A diagram consisting of three text elements on the left: 'Content and hierarchy', 'HTML authorship', and 'Design considerations'. A horizontal line connects the right side of 'Content and hierarchy' to the right side of 'Design considerations'. A vertical line descends from the midpoint of this horizontal line to the right side of 'HTML authorship'. A horizontal line then extends from the right side of 'HTML authorship' to the text '***Polishing best practices***' on the right.

# Introducing concepts

ARIA landmark roles



**aria-\***

*widget attributes*

*live region attributes*

*drag-and-drop attributes*

*relationship attributes*

Good idea: ***relationship attributes***

widget attributes

live region attributes

***relationship attributes***

drag-and-drop attributes

aria-activedescendant

aria-controls

aria-describedby

aria-flowto

aria-labelledby

aria-owns

aria-posinset

aria-setsize

# Good idea: *relationship attributes*

Attributes

Attributes

*Attributes*

Attributes

aria-activedescendant

aria-controls

aria-describedby

aria-flowto

aria-labelledby

aria-owns

aria-posinset

aria-setsize



# Introducing concepts

ARIA landmark roles

ARIA relationship attributes



# More into the safety zone

Content and hierarchy

HTML authorship

Design considerations

***Polishing best practices***

ARIA landmark roles

ARIA relationship attributes

***Introducing concepts***

# Getting more sophisticated

keyboard navigation and focus

ARIA standalone widgets

ARIA composite widgets



# Stepping out of our comfort zone

Content and hierarchy

HTML authorship

Design considerations

***Polishing best practices***

ARIA landmark roles

ARIA relationship attributes

***Introducing concepts***

Keyboard navigation and focus

ARIA standalone widgets

ARIA composite widgets

***More sophistication***

# A careful progression

Content and hierarchy

HTML authorship

Design considerations

***Polishing best practices***

ARIA landmark roles

ARIA relationship attributes

***Introducing concepts***

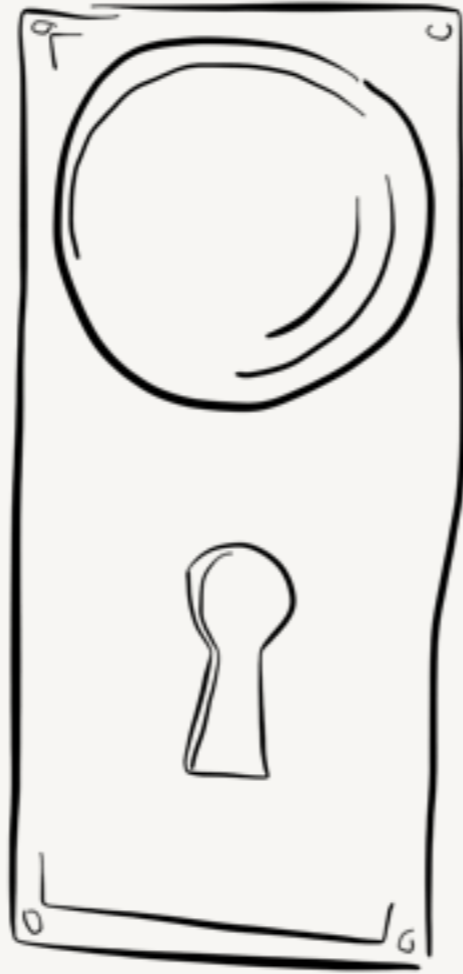
Keyboard navigation and focus

ARIA standalone widgets

ARIA composite widgets

***More sophistication***





A ***keyhole*** to the future

`<div role="article">`

`<div role="main">`

`<div role="navigation">`

Gradual phasing in of more semantics

`<article>`

`<main>`

`<nav>`

Gradual phasing in of more semantics



“ ***If you can use a native HTML element...*** or attribute with the semantics and behavior you require already build in... ***then do so.***

<http://w3c.github.io/aria-in-html/>

# First rule of ARIA

~~<form role="form">~~

You *can* overdo it

# A careful progression

Content and hierarchy

HTML authorship

Design considerations

***Polishing best practices***

ARIA landmark roles

ARIA relationship attributes

***Introducing concepts***

Keyboard navigation and focus

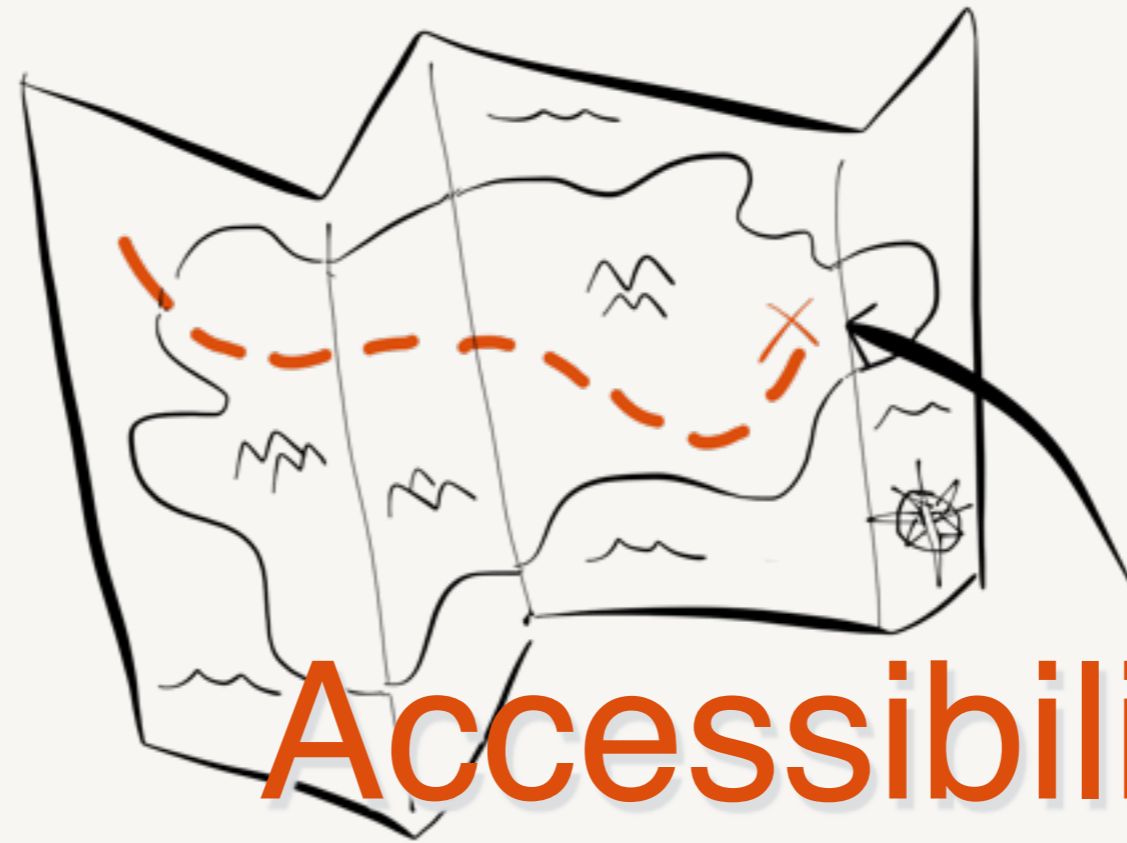
ARIA standalone widgets

ARIA composite widgets

***More sophistication***



# Synthesis



# Accessibility bliss

Our ideal path

# Co-existing realities

## Technical Context

“the nature of the Web”

generalism required

broad knowledge needed

stuff changing all the time

firehose of information

priorities

## External Processes

project requirements

constraints

deadlines

communication woes

development cycles

client priorities

## Individual Practices

technical priorities

design/dev decisions

skill level

best practices

strategic thinking

finding info

# Individual motivation is essential

## *Individual Perspective*

priorities  
technical decisions  
skill level  
best practices  
strategic thinking  
finding info



Rage quit



# A common goal is essential

*Surrounding Processes* project requirements  
constraints  
deadlines  
communication woes  
development cycles  
client priorities



Risk losing our *path* otherwise

# Chaos surrounds us

***Technical Context*** “the nature of the Web”  
generalist’s skillset  
broad knowledge needed  
inconsistent tech  
changing standards  
firehose of information



Things are too deep of a dive now

# We can do these things today

Content and hierarchy

HTML authorship

Design considerations

***Polishing best practices***

ARIA landmark roles

ARIA relationship attributes

***Introducing concepts***

Keyboard navigation and focus

ARIA standalone widgets

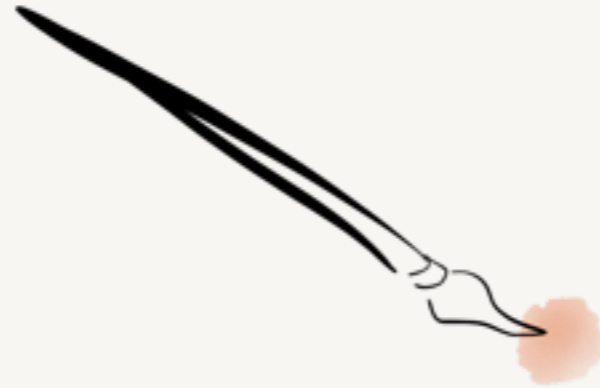
ARIA composite widgets

***More sophistication***





But when I think of the *future*...



We should do as *little* as possible

A11Y Project Patterns Checklist Resources About

# The Accessibility Project

A community-driven effort to make web accessibility easier.

[Learn more](#) [Contribute on Github](#)

## How-tos

- [How-to: Using Caption Services with HTML5 Video](#)  
How to implement captions on HTML5 video (and audio) elements.
- [How-to: Use Skip Navigation links](#)  
Use skip nav links to ease keyboard user fatigue and frustration.
- [How-to: Use TITLE attributes](#)  
Short answer: Avoid using title attributes except in a few special circumstances.
- [How-to: Use role="application"](#)  
**NEVER** use `role=application` on a widely containing element such as `<body>` if your page consists mostly of traditional widgets or page elements.
- [How-to: Future proof your accessibility efforts.](#)  
Ways to make your accessibility effort as future proof as possible.
- [Getting Started with OS X VoiceOver](#)

ARCHIVE CATEGORIES

- [How-tos](#)
- [Myths](#)
- [Quick tests](#)
- [Quick tips](#)
- [Basics](#)

<http://a11yproject.com/>

We need to fill in the *gaps*





We need to ***document*** clear steps

open source .com

DISCOVER AN OPEN SOURCE WORLD

redhat  
SUPPORTED BY RED HAT

open\* business education government health law life

## Why open source needs accessibility standards

Posted 19 Feb 2015 by [Shaun Gillies](#)  Rating: ★★★★★ (8 votes)



Image by : [opensource.com](#)

[Tweet](#) 144 [+ reddit this!](#) [Submit](#) [Like](#) 71 [g+](#) 20



My name is Shaun Gillies and I am a student of Information Technology at the Queensland University of Technology in Brisbane, Queensland. I also work as a web administrator and designer for a small company called La'Quar. I am an advocate of open source software, and like to contribute actively to the open source community.

[» More about me](#)

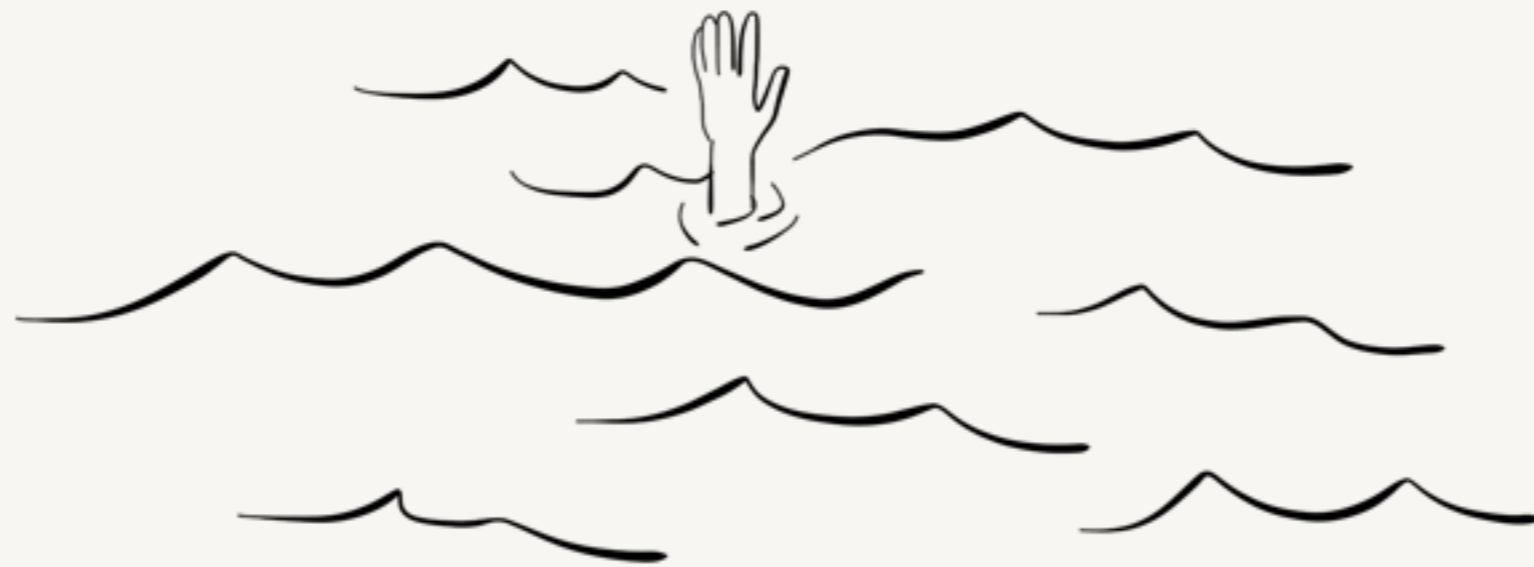
READER FAVORITE  
TINKERING WITH THE RASPBERRY PI A+

Most popular

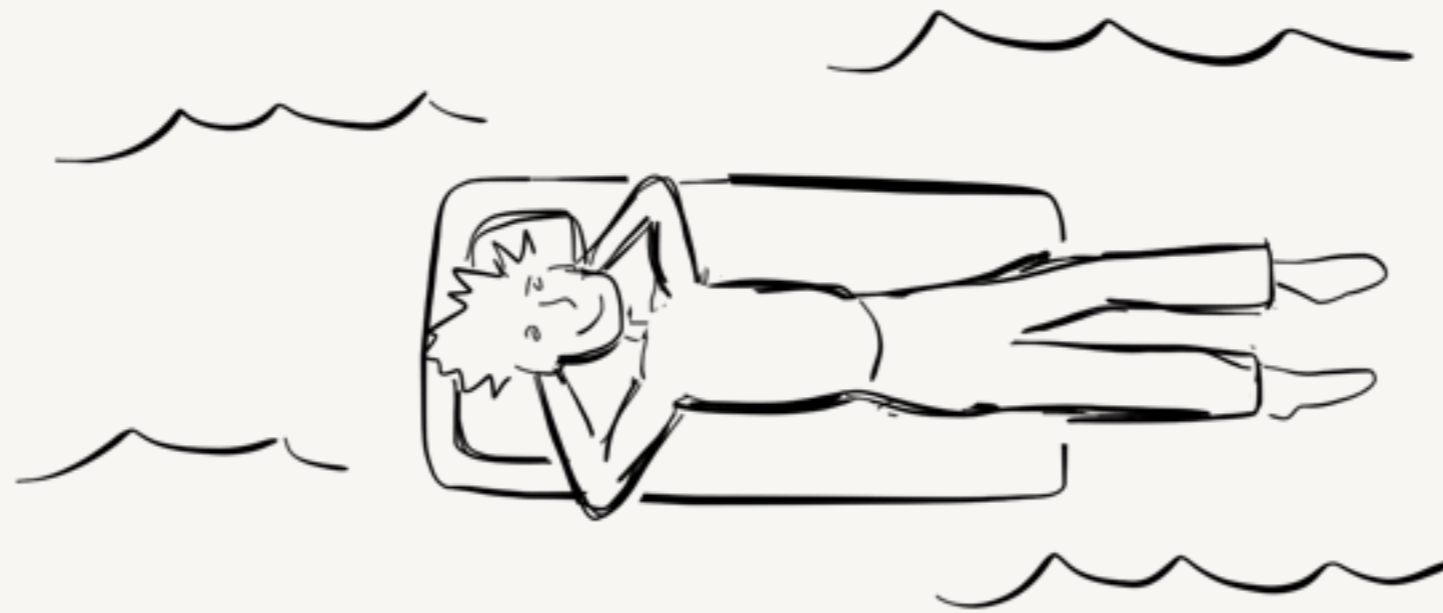
- Top 5 open source project management tools in 2015

<http://opensource.com/life/15/2/why-open-source-needs-accessibility-standards>

We need clarity and **standards**



To go from *this*



To *this*



To *this*



To *this*



To *this*



***Thank you***



# Accessibility for the Apathetic

*Lyza Danger Gardner*

Funka Accessibility Days | April, 2015 |

*[@lyzadanger](#)*