INTRODUCTION TO ACCESSIBILITY

What are disabilities?

Web accessibility is a measure of how effectively all people, including those with disabilities, are **able to access and use web pages and web media**.

A “disability” is any **continuing condition** that restricts everyday activities.

According to a 2015 ABS survey:

- **4,290,100** (18.3%) of Australians have some sort of disability.
- **3,392,600** (14.5%) of Australians have a disability that restricts daily activities.

http://www.abs.gov.au/ausstats/abs@.nsf/0/C258C88A7AA5A87ECA2568A9001393E8
Opendocument

In other words:

- at least **1 in 5 Australians** has some sort of disability
- almost **1 in 7 Australians** has a disability that restricts daily activities.

These ratios also increase with age.

- Around **2 in 5 Australians**, 65 years or older, have some sort of disability.

Types of disability

Disabilities are often broken down into four broad categories:

- **visual**
- **auditory**
- **motor skill**
- **cognitive**

1. Visual

Vision disabilities include:

Low Vision (vision loss that cannot be corrected with glasses)

- Macular degeneration
- Glaucoma
- Diabetic retinopathy
- Cataract

Colour-blindness

- Protanopia (red deficiencies)
- Deuteranopia (green deficiencies)
- Tritanopia (blue deficiencies)
- Rod monochromacy (no colour)

Blindness

Which includes very little to no vision.

2. Auditory

Auditory disabilities include:

Mild hearing loss is defined as the inability to hear sounds below 30 decibels.

For people with mild hearing loss, **speech can be difficult to understand**, especially if background noises are present.

Moderate hearing loss is defined as the inability to hear sounds below 50 decibels.

For people with moderate hearing loss, **a hearing aid may be required**.

Severe hearing loss is defined as the inability to hear sounds below 80 decibels.

Profound hearing loss is defined as the inability to hear sounds below 95 decibels, or any sound in some cases.

For people with severe hearing or profound loss, **communication may be done through sign language**; others rely on lip-reading techniques.

3. Motor skill

Motor-skill disabilities include:

Traumatic Injuries

- Spinal cord injury
- Loss or permanent damage to limb(s)

Diseases & Congenital Conditions

- Cerebral palsy
- Muscular dystrophy
- Multiple sclerosis
- Spina bifida
- ALS (Lou Gehrig’s Disease)
- Arthritis
- Parkinson’s disease

4. Cognitive

Cognitive disabilities include various intellectual or cognitive deficits.

In simple terms, a person who has a cognitive disability **has trouble performing mental tasks** that the average person would be able to do.

This category includes:

- intellectual disability
- developmental delay
- developmental disability
- learning disabilities such as Dyslexia and ADHD.

It can also include conditions that cause **cognitive impairment**:

- acquired brain injuries
- genetic disability such as Down syndrome, Autism, and Dementia

Accessibility barriers

As well as long-term disabilities, people can experience **situational or short-term barriers** that affect their ability to interact with websites and web content.

Vision barriers could include eye fatigue, blurred vision or even trying to look at a mobile screen in bright sunlight.

Auditory barriers could include hearing issues while in a room with loud music, or short-term hearing loss from exposure to loud noise.

Motor-skill barriers could include trying to perform a task while holding a baby, or with a broken arm

Cognitive barriers could including suffering from concussion or recovering from short-term memory loss.

Other barriers that are not technically disabilities but can have a major impact on peoples lives include **literacy and language**.

According to a 2009 ABS survey:

- **7.3 million** (44%) of Australians had literacy skills at Levels 1 or 2
- **6.4 million** (39%) at Level 3
- **2.7 million** (17%) at Level 4/5

http://www.abs.gov.au/ausstats/abs@.nsf/Lookup/4228.0main+features992011-2012

A large percentage of Australians with lower levels of literacy are non-native English speakers. This group is often referred to as **“English as a Second Language”** (ESL).

Exercise 01a:
Using keyboard-only

Before using any screen reader, it is important to understand how to navigate websites and applications **using the keyboard only**.

We'll start by opening a
demonstration page to practice on.

Open **exercise01-keyboard-only/exercise1.html**

Moving forwards and backwards

The TAB keystroke moves focus to the **next focusable element** on the page.

The `SHIFT TAB` keystroke moves focus to the **previous focusable element** on the page.

Select menus

The `DOWN ARROW` keystroke will move focus to the **next option in dropdown menu**.

The UP ARROW keystroke will move focus to the **previous option in dropdown menu**.

The ENTER and SPACEBAR keystrokes will **select the option** that is currently in focus.

The ESC keystroke will **close the dropdown menu**.

Radio buttons

As long as all radio buttons have a matching name value, **they will act as a radio button group**.

This means you can **only select one radio button** from within the group at a time.

The TAB will move focus **into and out of a radio button group**.

The SPACEBAR keystroke will **select the current radio button**.

The DOWN ARROW keystroke will **move focus to next radio button and select it**.

The UP ARROW keystroke will **move to focus to the previous radio button and select it**.

When a radio buttons has been selected from within a radio group, **it is impossible to uncheck radio buttons from within this group**. The selection can be changed, but not unselected.

Checkboxes

Unlike radio buttons within a group, checkboxes are **always treated as individual form controls**.

The TAB will move focus **into and out of each checkbox**.

The SPACEBAR keystroke will **select and unselect the current checkbox**.

Assistive technologies

Assistive technologies are products, equipment and systems that enhance activities for people with disabilities.

For digital accessibility, Assistive Technologies are often broken down into two categories:

- **Input devices**
- **Output devices**

Input devices

Input devices aid people **when interacting with** websites and applications.

An example would be where a user **has to fill in a form**.

However, it also includes simple activities such as **using keyboard functions to navigate** around a web page or web application.

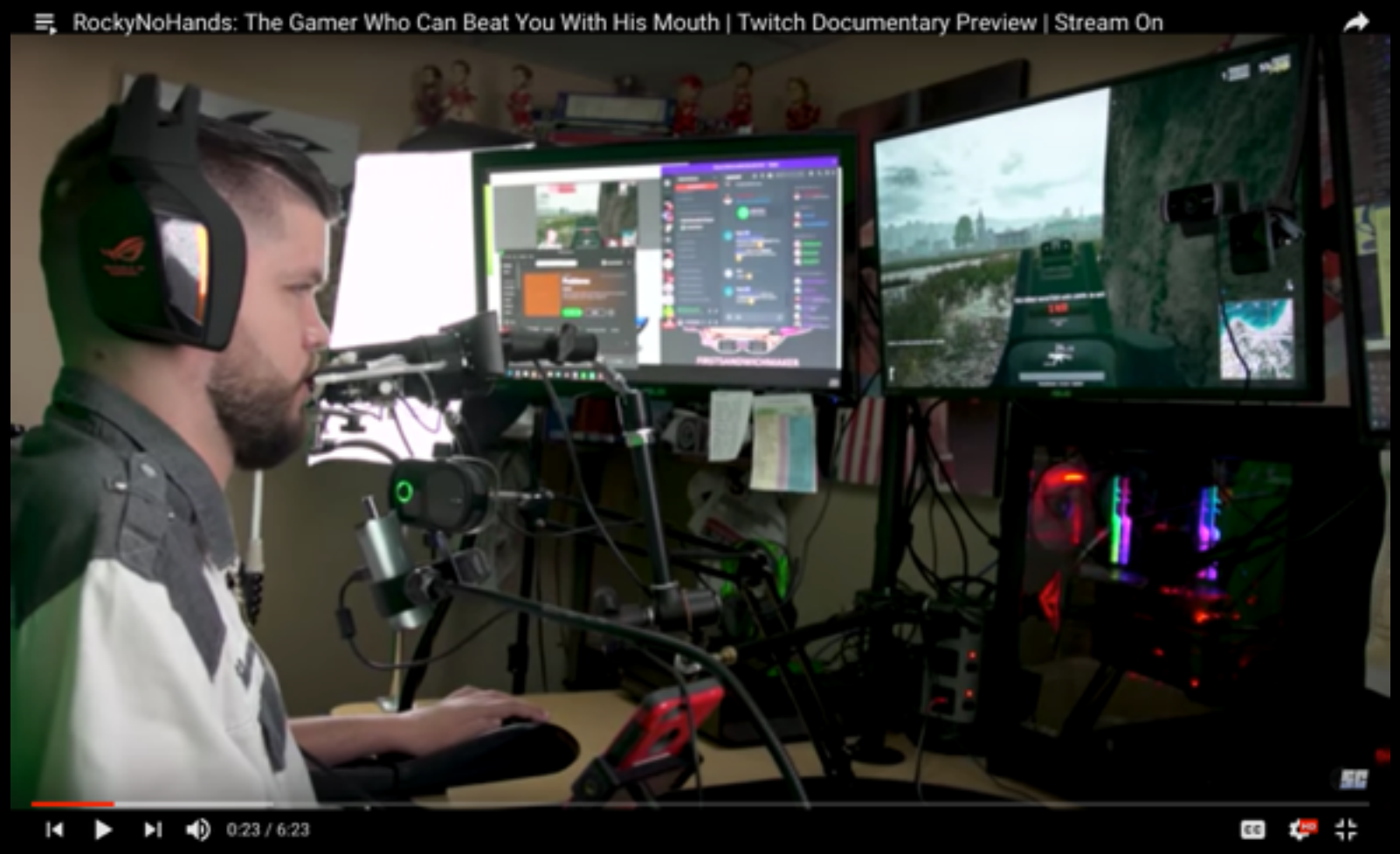
Input devices include:

Accessible keyboards, Track pads, Head wands, Mouth pieces, Puffers, Switches, Touch screens, Eye-trackers, Voice activation software, etc.

Judith: Cerebral Palsy

https://www.youtube.com/watch?v=CBlaiBV_yJs

Rocky: Tetraplegic due to spinal injury



https://www.youtube.com/watch?v=ZMvikz2cA-8

Output devices

Output devices aid people **when presenting information** from websites and applications.

Output devices include:

Magnifiers, Screen Readers, Refreshable Braille Devices etc.

Kim: Screen magnifier

https://www.youtube.com/watch?v=0_M1nEIaOBM

Bruce: Blind/Partially deaf

Exercise 01b: VoiceOver

Open **exercise01-keyboard-only/exercise1.html**

VO keys

VoiceOver uses **“VO” keys for control**. The default VO keys are the CONTROL + OPTION keystrokes.

The + symbol indicates that these keys are **used together**.

These two keys can be **changed in VoiceOver settings** as needed.

Starting and stopping

The COMMAND + F5 keystrokes will **start VoiceOver**.

Alternatively, **VoiceOver can be started manually via**:

System Preferences > Accessibility > VoiceOver > Enable VoiceOver

The COMMAND + F5 keystrokes will also **quit VoiceOver**.

Alternatively, VoiceOver can be **quit manually** by clicking the “X” icon in the top left corner of the VoiceOver panel.

Reading

The VO + A keystrokes will trigger VoiceOver to **start reading**.

The `CONTROL` keystroke will trigger VoiceOver to **stop reading**.

The VO + RIGHT ARROW keystrokes will **read the next item**.

The VO + LEFT ARROW keystrokes will **read the previous item**.

The VO + B keystrokes will **read from top of the page to the current location**.

Navigating

The VO + COMMAND + L keystrokes will **take you to the next link**.

The VO + COMMAND + H keystrokes will **take you to the next heading**.

The VO + COMMAND + J keystrokes will **take you to the next form control**.

The VO + COMMAND + X keystrokes will **take you to the next list**.

The VO + COMMAND + T keystrokes will **take you to the next table**.

The VO + COMMAND + U will **display the Rotar**.

Exercise 01c: NVDA

Open **exercise01-keyboard-only/exercise1.html**

NVDA key

The **NVDA key** is set to the `INSERT` key by default, but it can be changed to the `Caps lock` key when installing NVDA for the first time.

If you want to **change your NVDA key preferences**, press CTRL + NVDA + K.

Starting and stopping

The NVDA application **needs to be manually opened** in order to begin reading.

The INSERT + Q keystrokes will **quit NVDA**.

Reading

The INSERT + DOWN ARROW keystrokes will trigger NVDA to **start reading continuously from this point on**.

The `CONTROL` keystroke will trigger NVDA to **stop reading**.

The DOWN ARROW keystrokes will **read the next item**.

The UP ARROW keystrokes will **read the previous item**.

Navigating

The K keystrokes will **take you to the next link**.

The L keystrokes will **take you to the next list**.

The H keystrokes will **take you to the next heading**.

The T keystrokes will **take you to the next table**.

What is WCAG?

The W3C

The **World Wide Web Consortium** or the W3C is an international community that develops the open standards for the Web.

The W3C produces **specifications** on a wide range of web-related topics including HTML, CSS and Accessibility.

W3C technical specifications have **four levels of maturity**:

1. Working Draft (WD)

A document that W3C has published for review by the community, including W3C Members, the public, and other technical organizations.

2. Candidate Recommendation (CR)

A document that satisfies the Working Group's technical requirements, and has already received a comprehensive review.

3. Proposed Recommendation (PR)

A document that has been accepted by the W3C Director as of sufficient quality to become a W3C Recommendation.

4. W3C Recommendation (REC)

A specification or set of guidelines that, after extensive consensus-building, has received the endorsement of W3C Members and the Director.

WAI

Within the W3C, there is a sub-group called the **Web Accessibility Initiative** (WAI) Working Group.

The WAI Working Group has been responsible for developing the **Web Content Accessibility Guidelines** (WCAG).

WCAG

The WCAG guidelines **provide a standard for web content accessibility**.

WCAG 1.0 became a **W3C Recommendation** in May 1999.

https://www.w3.org/TR/WAI-WEBCONTENT/

WCAG 2.0 became a **W3C Recommendation** in December 2008.

http://www.w3.org/TR/WCAG20/

WCAG 2.1 became a **W3C Recommendation** in June 2018.

https://www.w3.org/TR/WCAG21/

WCAG Structure

Principles

- **4 Principles**
  - 13 Guidelines
    - 78 Success Criteria
      - Sufficient Techniques
        - Advisory Techniques
      - Failures

Four key “POUR” principles

- **Perceivable**
- **Operable**
- **Understandable**
- **Robust**

Perceivable: Information and user interface components must be presentable to users in ways they can perceive - it can’t be invisible to all of their senses.

Operable: User interface components and navigation must be operable - the interface cannot require types of interaction that a user cannot perform.

Understandable: Information and the operation of user interface must be understandable - the content or operation cannot be beyond their understanding.

Robust: Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies - even as technologies and user agents evolve.

Guidelines

- 4 Principles
  - **13 Guidelines**
    - 78 Success Criteria
      - Sufficient Techniques
        - Advisory Techniques
      - Failures

PERCIEVABLE

Guideline 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.

PERCIEVABLE

Guideline 1.2 – Time-based Media

Provide alternatives for time-based media.

PERCIEVABLE

Guideline 1.3 – Adaptable

Create content that can be presented in different ways (for example simpler layout) without losing information or structure.

PERCIEVABLE

Guideline 1.4 – Distinguishable

Make it easier for users to see and hear content including separating foreground from background.

OPERABLE

Guideline 2.1 – Keyboard Accessible

Make all functionality available from a keyboard.

OPERABLE

Guideline 2.2 – Enough Time

Provide users enough time to read and use content.

OPERABLE

Guideline 2.3 – Seizures and Physical Reactions

Do not design content in a way that is known to cause seizures.

OPERABLE

Guideline 2.4 – Navigable

Provide ways to help users navigate, find content, and determine where they are.

OPERABLE

Guideline 2.5 – Input Modalities

Make it easier for users to operate functionality through various inputs beyond keyboard.

UNDERSTANDABLE

Guideline 3.1 – Readable

Make text content readable and understandable.

UNDERSTANDABLE

Guideline 3.2 – Predictable

Make Web pages appear and operate in predictable ways.

UNDERSTANDABLE

Guideline 3.3 – Input Assistance

Help users avoid and correct mistakes.

ROBUST

Guideline 4.1 – Compatible

Maximize compatibility with current and future user agents, including assistive technologies.

Success criteria

- 4 Principles
  - 13 Guidelines
    - **78 Success Criteria**
      - Sufficient Techniques
        - Advisory Techniques
      - Failures

The **78 Success Criteria** are a checklist that can be used to determine if a website/application conforms to WCAG 2.0 guidelines.

Each of the **78 success criteria** is defined as either A, AA or AAA compliance.

Level A: satisfies all the Level A Success Criteria.

Level AA: satisfies all the Level A & Level AA Success Criteria.

Level AAA: satisfies all the Level A, Level AA & Level AAA Success Criteria.

Sufficiency techniques

- 4 Principles
  - 13 Guidelines
    - 78 Success Criteria
      - **Sufficient Techniques**
        - Advisory Techniques
      - Failures

Sufficient techniques are **reliable ways to meet the success criteria** from an author's perspective and from an evaluator's perspective.

Advisory techniques

- 4 Principles
  - 13 Guidelines
    - 78 Success Criteria
      - Sufficient Techniques
        - **Advisory Techniques**
      - Failures

Advisory techniques are **suggested ways to improve accessibility**. They are often very helpful to some users, and may be the only way that some users can access some types of content.

Failures

- 4 Principles
  - 13 Guidelines
    - 78 Success Criteria
      - Sufficient Techniques
        - Advisory Techniques
      - **Failures**

Failures are things that **cause accessibility barriers and fail specific success criteria**. The documented failures are useful for authors and evaluators.

Some common terms

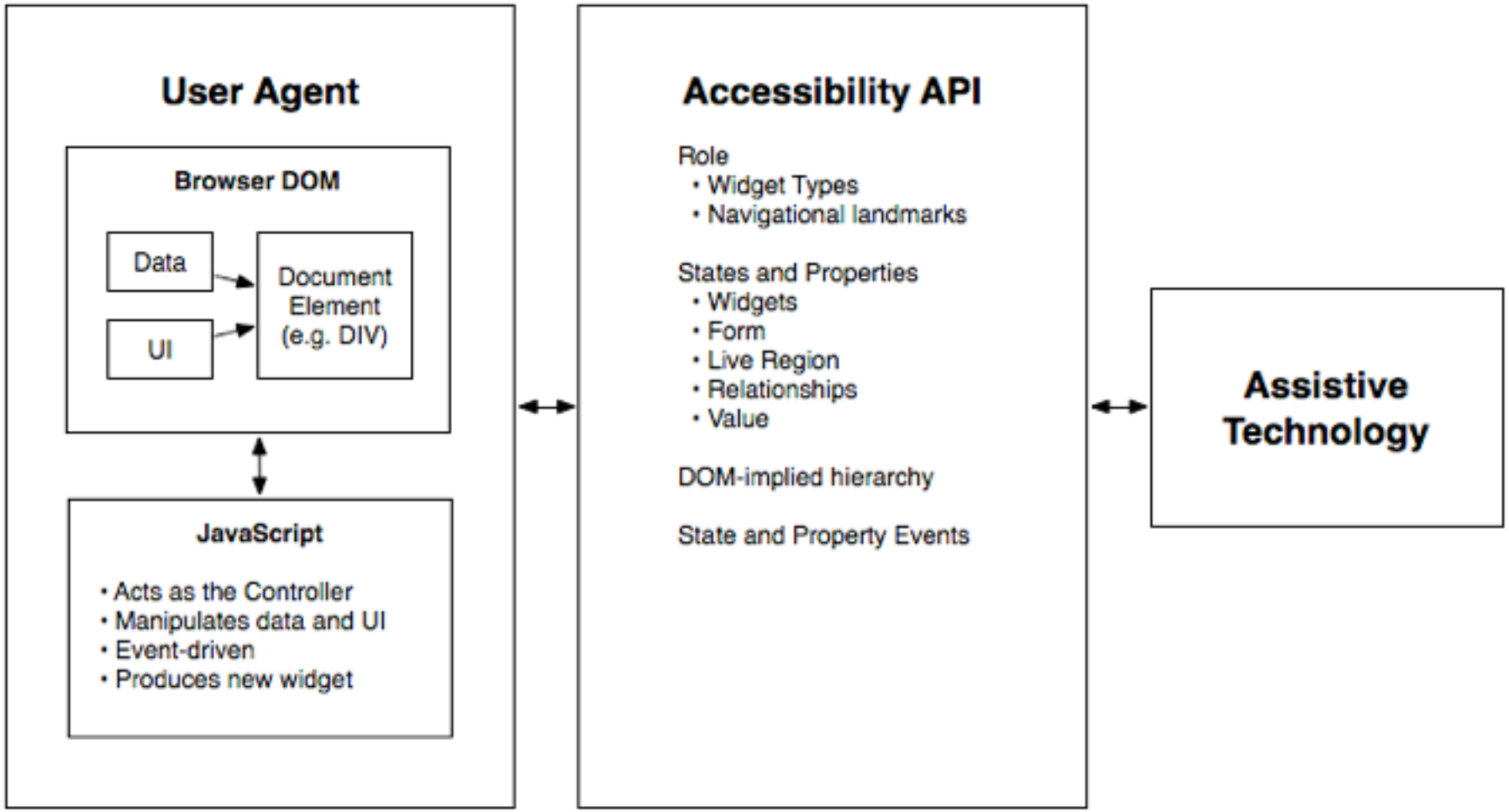
Here are some **common terms** that are relevant to accessibility.

Accessibility API

Accessibility application programming interfaces (APIs) are used to **communicate semantic information about the user interface** to Assistive Technologies.

“Accessibility APIs constitute a **contract between applications and assistive technologies**, to enable them to access the appropriate semantics needed to produce a usable alternative to interactive applications.”

For example, the Accessibility API **helps screen reading software** determine whether a particular UI widget is a menu, button, text field, list box, etc.


User Agent
Browser DOM
Data
UI
Document Element (e.g. DIV)
JavaScript
• Acts as the Controller
• Manipulates data and UI
• Event-driven
• Produces new widget
Accessibility API
Role
• Widget Types
• Navigational landmarks
States and Properties
• Widgets
• Form
• Live Region
• Relationships
• Value
DOM-implied hierarchy
State and Property Events
Assistive Technology

Accessibility APIs **expose information about each object** within the application such as:

1. The object's **role** (e.g. a menu, a button, an input, an image).

2. A **name that identifies the object within the interface** (e.g. a visible label or a name that has been encoded directly in the object).

3. The object's **state** (e.g. selected, unselected, checked, unchecked).

More than one API?

In OS X Safari and Chrome support **NSAccessibility**.

In iOS Safari and Chrome support **UIAccessibility**.

Some browsers support **one or more of the available accessibility APIs** for the platform they're running on.

In Windows, Firefox and Chrome support **MSAA/IAccessible** and **IAccessible2**.

And Internet Explorer supports **MSAA/IAccessible** and **UIAExpress**.

This is why you should always test against **more than one** Browser/ Assistive Technology combination.

Windows

IE: JAWS & NVDA
FireFox: JAWS & NVDA
Chrome: JAWS & NVDA

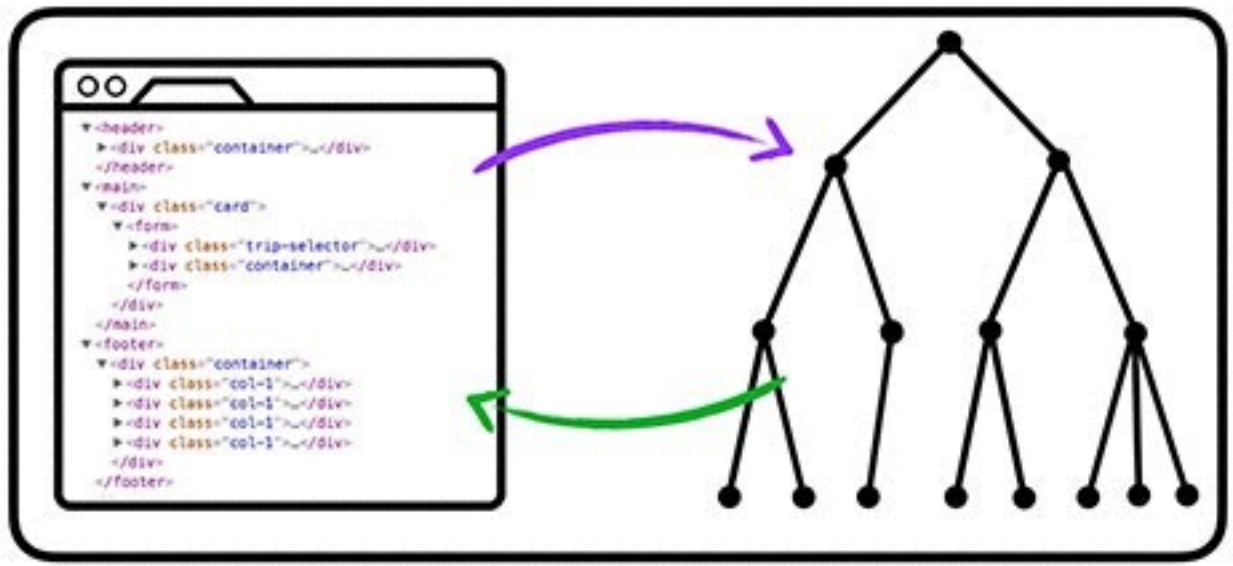
OSX

Safari: VoiceOver
FF: VoiceOver
Chrome: VoiceOver

Accessibility Tree

Browsers take the DOM tree **and modify it**, to turn it into a form that is useful for assistive technologies.

This modified tree, is referred to as **the accessibility tree** - a subset of of the DOM tree.


▼<header>
▶ …
</header>
▼<main>
▼
▼<form>
▶ …
▶ …
</form>

</main>
▼<footer>
▼
▶ …
▶ …
▶ …
▶ …

</footer>
DOM
accessibility tree

The accessibility tree contains **only “Accessible objects”**. These are nodes that have states, properties or events.

All other DOM nodes (that do not have states, properties or events) **are not presented in the accessibility tree**.

For example, **a section within the DOM tree** could be:

```
<div class="container">
  <form action="#">
    <div class="form-container">
      <label for="name">Name</label>
      <input id="name" type="text">
    </div>
    <div class="form-container">
      <button type="submit">Submit</button>
    </div>
  </form>
</div>
```

The Accessibility tree would **only present the following**:

```
<form action="#">

    <label for="name">Name</label>
    <input id="name" type="text">


    <button type="submit">Submit</button>

</form>
```

Each browser could potentially present a **slightly different accessibility tree**.

Widgets

Within the various WAI ARIA specifications, there are multiple references to **“widgets”**.

A widget is a component that **enables a user to perform a function** or access a service such as a dropdown menu, a modal or a tooltip.