

Helsingin malli



Accessibility: Implementation in Code, part 2/2

Tero Pesonen / Siteimprove



Contents

- Web Forms
 - Structure
 - Error handling
 - Composite elements
- Aria live announcements
- Aria current status

Additional material

- Web Forms
 - HDS:
 - Form guidelines
 - Form implementation with error checking
 - Demo page:
 - Form error checking and focus management examples
- Aria live & Aria current
 - Demo page:
 - For a list of example patterns, see Information → Contents
 - Static Live Regions: Notification, Log, Combobox
 - Non-static live regions: Spinner

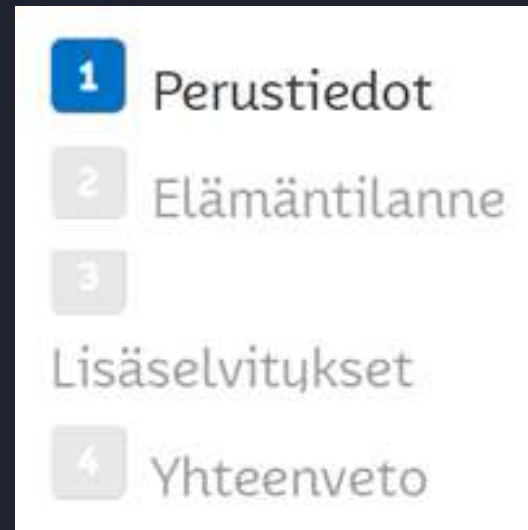
Tailored HTML elements

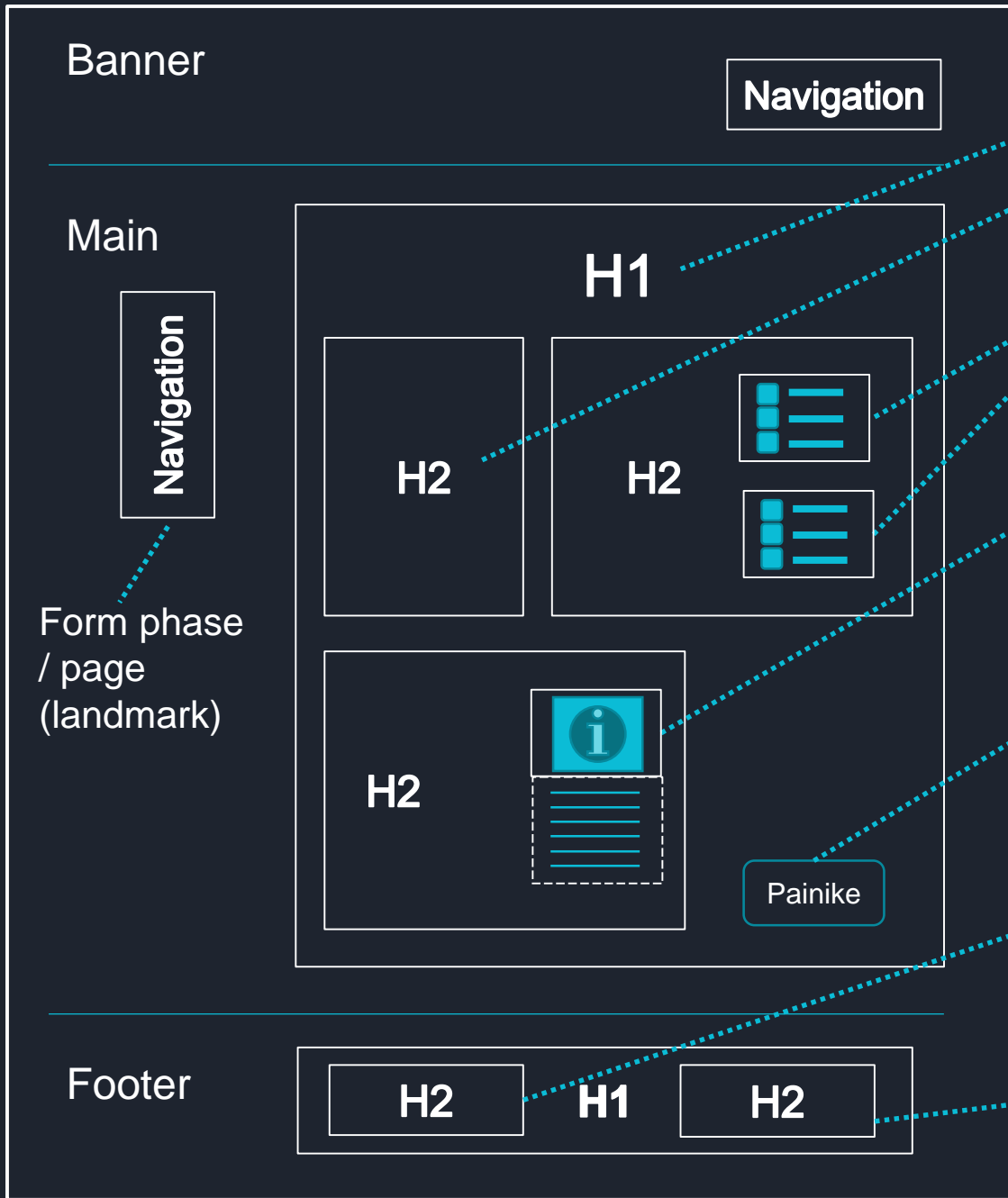
When implementing non-standard elements, developers must ensure that the element:

1. Describes its **name**, **role** and **value/state** to assistive technology
2. Is keyboard focusable, if interactive, and
3. Can be activated and interacted with by keyboard
4. Presents either the system default focus indicator or one that meets WCAG 1.4.11 & 2.4.7 contrast requirements.
5. Remember, too: Should the element trigger dynamic content changes (pop-up, menu button), or be a complex element with sub parts (calendar, navigation menu), its internal focus and navigation order is set according to the service design specification, that is, is not random.

Forms: Structure

- A long form when divided into pages or steps/phases is often easier work with for all users
 - One category or theme per page/phase
- Show steps and progress
 - Specify how the progress is described to assistive technology
- Structure each form page with headings
 - Headings can be more numerous than on normal content pages
 - Heading per topic → Adds also visual documentation
- Make use of HTML Fieldset sections and legend titles when building control groups
- Submit/next page button should be the last element of the form





Banner

Navigation

Main

Navigation

H1

H2

H2

H2



Submit/Next button (not in the footer!)

Form phase / page (landmark)

Footer

H2

H1

H2

Form page heading

Form section heading ("Personal information")

Fieldset + Legend
Group of controls

Icon as a button



Expandable dynamic content
Role=region
Aria-labelledby=icon

Pop-up -help icon

Submit/Next button (not in the footer!)

Contact info

- Address:
- Phone num: ...
- Email: ...

Visit us in social media

- Facebook
- Twitter
- Instagram

Forms: Navigation order

- Typical focus order
 - Form navigation or progress → Form segment → H1 → Help section heading → Help section → Form section X heading → Control groups, from left to right, top to bottom; help resources available before the control → Next/submit button → Previous (and other) buttons.
- Pay attention to situations where content is added or removed dynamically from a form page
 - For instance, checking “I am currently employed full or part time” brings additional questions to a form.
 - New content should be added ahead of the present focus point
- Content that is part of the form should normally not be placed after the submit button
 - “I have read the...” checkbox before the button
 - Important “small print” also before the button or linked at from within the form

Tablist Dem 1

2 Buttons	Form	Linkboxes	5 Tablist
---------------------------	-------------	-----------	---------------------------

Form With Error Handling 6

7 1. Personal info
2. Radiobuttons
3. Third page
10 4. Summary

Page 2 of 4: Radiogroups 11

ARIA-radios 12

Test Group 13

- First Option
- Second Option
- Third Option 16

No pre-selection 17

- Radio 1
- Radio 2

Form With Error Handling

1. Personal info
2. Radiobuttons
3. Third page
4. Summary

Page 1 of 4: Personal information

Input fields

Name:

Email address:

Email address is invalid

Combobox

City of residence:

Helsinki

Next Page

The combobox is a widget that merges an input field with a drop-down list box. The list opens dynamically to offer similar or matching candidates to the one being typed in the field. The user can then either type the value directly or select one from the list shown. In this particular combobox, the values are cities in English; typing

Elämäntilanne—lomakesivun avaaminen:

Kohdistus voidaan siirtää suoraan Elämäntilanne—
otsikkoon koko sivun alun sijasta

1 Perustiedot

2 Elämäntilanne

3 Lisäselvitykset

4 Yhteenveto

Elämäntilanne

 Ongelmia lomakkeen toiminnassa?

Ohjeet kenttien täyttöön

Sivu 2/4

Pakollinen kenttä on merkitty punaisella tähdellä: *

Elämäntilanne

Hakija

Elämäntilanne *

Lyhyt kuvaus nykyisestä elämäntilanteesta

Maksimi 50 merkkiä

Puoliso

Elämäntilanne *

Lyhyt kuvaus nykyisestä elämäntilanteesta


Maksimi 50 merkkiä

Kohdistettu elementti poistetaan: Mihin selainkohdistus tulisi siirtää roskakalattikko—painikkeen aktivoinnin jälkeen?

Kotona asuvien alaikäisten lasten nimet ja henkilötunnukset


Etunimi * Sukunimi * Henkilötunnus *



Etunimi Sukunimi Henkilötunnus 



Muiden samassa asunnossa asuvien nimet ja henkilötunnukset

Etunimi Sukunimi Henkilötunnus 



Error checking

- Demo form: <http://tpesonen.net/Demo>
- WCAG 3.3.1 & 3.3.3: Error identification and Error suggestions
- If the form identifies errors in user input, the user must be informed
 - Where in the form the error is (a control, a group of interacting choices, etc.)
 - The error type or what the error situation entails (invalid value, missing value, etc.)
- The error must be observable non-visually
 - Use of colour alone insufficient
 - Asterisk (*) without explanation insufficient
- The error message should not disappear unexpectedly
 - E.g. a pop-up that only briefly displays the error location is difficult to spot with assistive technology.

Error checking

- Error checking can be
 - Static
 - Dynamic
 - Hybrid of static and dynamic
- Static: Errors are reported on a submission/page transition attempt → The user is precluded from advancing with the form till the errors are fixed
- Dynamic: Form controls check their input value immediately after or even during input and produce an error message → Submission possible only after no errors remain.
- Hybrid: Some errors may be checked in a static way while other issues, such as formatting, are checked immediately, dynamically

Static error checking

- Error report / board printed in the beginning of the form/page
 - Includes all the errors found in the form section
 - Described with a heading + possibly a region
- Focus is transferred **automatically** from the button to the error report after a failed submission attempt
 - Target: Heading
- Errors are listed and written out clearly
- Each error is coupled with a link that points at the control or group whose input is causing the error
 - I.e. internal link

Form With Error Handling

1. Personal info
2. Radiobuttons
3. Third page
4. Summary

Page 1 of 4: Personal information

Input fields

Name:

Email address:

Combobox

City of residence:

[Next Page](#)

1. Personal info

2. Radiobuttons

3. Third page

4. Summary

Page 1 of 4: Personal information

The form contains errors:

- Error 1: [Name must not be empty.](#)
- Error 2: [Email address is invalid.](#)
- Error 3: [Invalid city of residence.](#)

Input fields

Name:

Email address:

Combobox

City of residence:

Next Page

Dynamic error checking

- User input is checked a) during or b) immediately after the input is completed
- Implementors need to be informed whether A or B applies.
 - A and B approaches call for different ARIA implementation techniques and navigation order issues that need to be accounted for.
- Error messages are printed in the control's immediate vicinity
 - Recommendation: Immediately after but before the next control
- Error messages remain visible till the control is re-focused or the error is amended
- Aria-live and role="alert" techniques should be applied
 - SC users receive a notification even as the error is not focused

Form With Error Handling

1. Personal info
2. Radiobuttons
3. Third page
4. Summary

Page 1 of 4: Personal information

Input fields

Name:

Name must not be empty.

Email address:

Email address is invalid.

Combobox 

City of residence:

Next Page

Combining the error checking modes

- A form may provide both static and dynamic checking
- Example: Some of the inputs are checked dynamically
 - Date format, social security number format, postal codes, etc.
 - But more complex dependencies between inputs (one input requires that another one is provided, too, or is in a specific format) are evaluated only on submission
 - Benefit: Aids the user in inputting correct data, and limits the number of static error messages as basic formatting issues need not be fixed later.
 - Benefits most forms

Aria—live

- Changes to a live region will be announced by a screen reader even when the region is not focused
- Aria live region is attributed a HTML tag (div, span, li, ...)
 - Assistive technology will monitor the region for changes
 - Developers can specify what type of changes the region records
 - The DOM can host multiple Live regions at a given time
- WCAG requirements (WCAG 4.1.3)
 - Only a region to which focus is **not transferred** needs a live region (is e.g. not a modal)
 - Live regions should only monitor **relevant** content changes
 - Routine/known content updates if made live regions will unnecessarily burden the user (News headlines carousel, “weathar now” frame, etc.)

Aria—live: Politeness

- Sets up a live region
- Permanent, but has to be set prior to the first change to the region
- `Aria-live="politeness"`
 - **Polite:** AT will notify the user when its synthesizer is free from other tasks → queued
 - **Assertive:** AT will interrupt other tasks and speak out the announcement without delay. Use only when necessary.
 - **Off:** Live region inactivated → Can be turned back on

```
<div class="MessageArea">  
  <ul class="MessageList" aria-live="polite"></ul>  
</div>
```

```
<div id="log" class="Log" role="region" aria-labelledby="h2_log"
aria-live="polite">div</div>
```

Buttons

Standard button

Toggle attributes

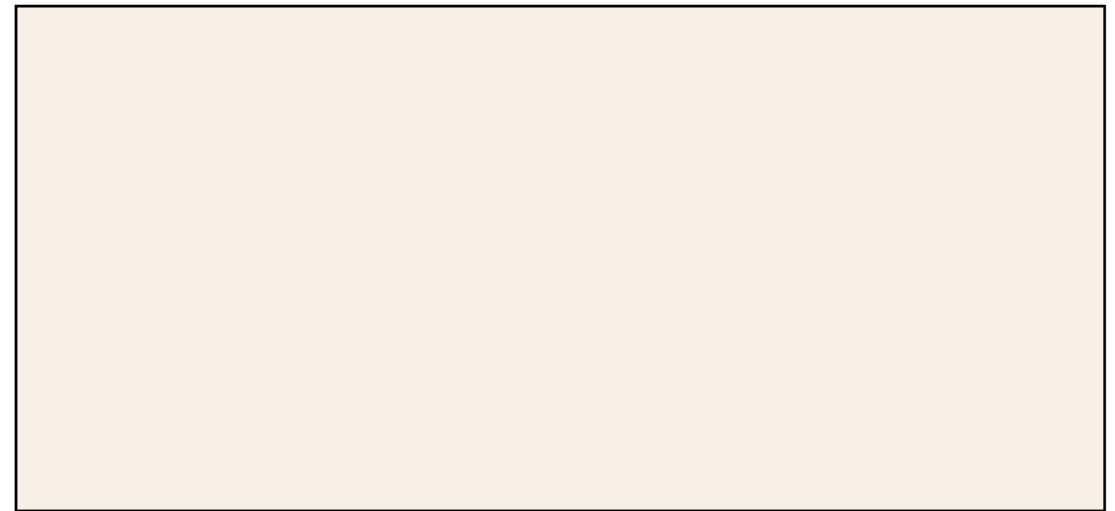
Accordion Button +

Toggle attributes

Menu Button ≡

Log

Announce log entries to SC



Aria—live: Attributes

- Aria-relevant=[parameter list]:What changes should be heeded?
 - Additions: Content that is added to the region
 - Removals: Content that is removed from the region
 - Text: Respond to textContent changes only
 - All: Any kind of change
 - Note: By default, aria live regions have aria-relevant="additions, text", so the attribute usually need not be set
- Aria-atomic=true/false: Is the entire region announced when a change occurs, or only the changed part of the region?
 - By default, aria-atomic="false"

Example: Should only the part of the string which indicates the number of results change (50), a screen reader would only announce "50", "35", etc. if `aria-atomic="false"` or `aria-relevant="additions"`. Using `aria-atomic="true"`, or combining the result + number into a single string, solves the problem.

```
<div class="..." aria-live="polite" aria-atomic="true">  
<span>50</span><span>hakutulosta</span></div>
```

satu

Yksikkö ▼ Nimike ▼ Tieteenala ▼

50 hakutulosta

JÄRJESTÄ: Relevanssi ▼

SU SATU UURINMÄKI
HENKILÖSTÖKOORDINAATTORI
HENKILÖSTÖPALVELUT

SP SATU PAAVOLA
Tieteenalat: Psykologia

The combobox hosts a live region that is visually hidden. Benefits:

- Combobox can provide SC with additional information that need not be exposed visually
- Example: "5 results found", "no results found", "Helsinki selected."
- Combobox creates and removes the live region on focus and blur events.

```
<div  
  id="combobox1_live_region"  
  aria-live="assertive"  
  class="LiveRegion">  
    2 Results found.  
</div>
```

Form With Error Handling

1. Personal info
2. Radiobuttons
3. Third page
4. Summary

Page 1 of 4: Personal information

Input fields

Name:

Email address:

Combobox

City of residence:

Caracas
Cardiff

Next Page

Aria-current (WCAG 1.3.1)

- When a UI element signals by its visual style that it corresponds to a present or active status, aria-current can pass the same information to assistive technology (WCAG 1.3.1 Info and Relationships)
- Aria-current=
 - "page": breadcrumb current page, navigation menu current page, ...
 - "step": presently open/active phase in a process. E.g. form step, wizard step, etc.
 - "date": current date (not necessarily currently **selected** date)
 - "time": current time
 - "true"/"false": generic "current" status, or none.





Aika ja päivämäärä

<< toukokuu kesäkuu 2019 heinäkuu >>

	ma	ti	ke	to	pe	la	su
22	27	28	29	30	31	1	2
23	3	4	5	6	7	8	9
24	10	11	12	13	14	15	16
25	17	18	19	20	21	22	23
26	24	25	26	27	28	29	30

Tämä päivä Valittu päivä Vapaita aikoja

Etusivu > Päivystys > Päivystyspisteet

Päivystyspisteet

Arvioi tilanteesi kiireellisyys

Asiointi päivystyksessä

Asiakkaille ja läheisille

Päivystyspisteet

Turun alueen yhteispäivystys

Loimaan päivystys

Turku >
Tyks, T-sairaala, Savitehtaankatu 1, Turku
Puh. 02 313 8800

Loimaa >
Seppälänkatu 15-17, 32200 Loimaa
Puh. 02 313 8800

Salon >
Sairaalantie 9, 24130 Salo

Aria-current="date"

Aria-pressed="true"

Aria yhteenveto: Valitun tilan osoittaminen

- Aria-selected
 - Role="tab" (sisältyy role="tablist"—elementtiin)
 - Role="option" (sisältyy role="listbox"—elementtiin, joka vastaa HTML Select—tägiä)
- Aria-checked
 - Role="checkbox"
 - Role="radio"
- Aria-current
 - Vallitseva sivukonteksti
- Aria-pressed
 - Role="button" (kytkinpainike, toggle—button)
- Aria-expanded
 - Role="button" (haitari, valikko)
- Aria-label="Vapaamuotoinen kuvaus, jos vakiomuotoinen tila-attribuutti ei sovellu"