TEKNINEN LIITE

Saavutettavuusauditointi — https://swappie.com/fi/
Mikko Tarkiainen, saavutettavuusasiantuntija | 31.1.2026

Työkalut: Google Lighthouse · WAVE · axe DevTools · MacOS VoiceOver · Colour Contrast Analyser (CCA) 3.5.5

Yksityiskohtaiset korjauskoodit käsittelyohjeineen

1. Semanttinen HTML — Tuotevalinnat

Tuotevalinnat on toteutettu -elementeillä. on inline-elementti ilman sisäänrakennettua rooliä — se ei saa näppäimistöfokusta ja ruudunlukija ei tunnista sitä valintapainikkeeksi.

1.1 Nykyinen virheellinen koodi

```
<!-- ✗ NYKYINEN: ei toimi ruudunlukijalla tai näppäimistöllä -->
<div class="selector-container">
<span class="option-title">256 GB</span>
</div>
```

1.2 Korjattu koodi

HTML — Radio button + Label

```
<!-- ✓ KORJATTU: natiivi radio + label -->
<div class="selector-item">
<input
type="radio"
id="storage-256"
name="storage"
value="256"
class="visually-hidden"
>
<label for="storage-256" class="selector-button">
<span class="title">256 GB</span>
<span class="price-diff">+50 €</span>
</label>
</div>
```

CSS — Visually-hidden + Focus

```
/* Piilotetaan input visuaalisesti, jätetään ruudunlukijalle */
.visually-hidden {
position: absolute;
width: 1px;
height: 1px;
padding: 0;
margin: -1px;
overflow: hidden;
clip: rect(0, 0, 0, 0);
border: 0;
}

/* Focus näkyy labelissa kun input saa fokuksen */
.visually-hidden:focus + .selector-button {
outline: 3px solid #000;
outline-offset: 2px;
}
```

1.3 Miksi tämä toimii

- <input type="radio"> saa näppäimistöfokuksen automaattisesti (Tab + nuolinäppäimet)
- <label for="..."> yhdistää visuaalisen elementin inputtiin — kliikkaus labelissa aktivoi inputin
- Ruudunlukija ilmoittaa: "256 GB, radio button, valitsematon" — selkeä ja informatiivinen
- visually-hidden piilottaa inputin näkymästä, mutta säilyttää sen ruudunlukijalle

2. Dynaaminen sisältö — aria-live

Kun käyttäjä vaihtaa tuotteen ominaisuuksia, hinta päivittyy visuaalisesti, mutta ruudunlukija ei ilmoita muutoksesta.

2.1 Korjattu koodi (React / Next.js)

```
// ✓ aria-live on suoraan muuttuvassa <span>-elementissä
// (ei ympäröivässä <div>-kerroksessa)
<div className="price-container">
<span
className="current-price"
aria-live="polite"
aria-atomic="true"
>
{productPrice} €
</span>
</div>
```

2.2 Selitys

- **aria-live="polite"** kertoo ruudunlukijalle, että elementin sisältö muuttuu dynaamically — ruudunlukija ilmoittaa muutoksesta nykyisen puheen jälkeen.
- **aria-atomic="true"** pakottaa ruudunlukijan lukemaan *koko* elementin sisällön uudelleen (esim. "149 €"), ei vain muuttunutta osuetta.
- aria-live on **suoraan** muuttuvassa -elementissä, ei ympäröivässä kerroksessa — tämä takaa luotettavan ilmoituksen kaikissa ruudunlukijoissa (VoiceOver, NVDA, JAWS).

3. SPA-navigoinnin fokuksen hallinta

Swappie käyttää Next.js-pohjaista SPA-arkkitehtuuria. Kun käyttäjä klikkaa linkkiä, URL muuttuu, mutta fokus palaa sivun alkuun.

3.1 Korjattu koodi (React + Next.js)

```
import { useEffect } from 'react';
import { usePathname } from 'next/navigation';

function FocusManager() {
const pathname = usePathname();

useEffect(() => {
const h1 = document.querySelector('h1');
if (h1) {
h1.setAttribute('tabindex', '-1');
h1.focus();

// Poista tabindex focuksen jälkeen
// jotta H1 ei jää tab-järjestykseen
h1.addEventListener('blur', () => {
h1.removeAttribute('tabindex');
}, { once: true });
}
}, [pathname]);

return null;
}
```

3.2 Selitys

- **usePathname()** palauttaa nykyisen URL-polun — useEffect laukaistaan aina, kun polku muuttuu (eli kun käyttäjä navigoi uudelle sivulle).
- **tabindex="-1"** lisätään H1-elementtiin väliaikaista fokusointia varten. Ilman sitä <h1> ei ole ohjelmatisesti fokusoitava.
- **h1.focus()** siirtää fokuksen suoraan sivun otsikkoon — käyttäjä kuulee, millä sivulla on, eikä joudu käymään läpi navigaatiota alusta.
- **blur-tapahtuma** poistaa tabindex="-1" focuksen jälkeen, jotta H1 ei jää tab-järjestykseen. { once: true } takaa, että listener poistetaan automaattisesti.

4. "Siirry sisältöön"-linkki (Skip Link)

Sivulta puuttuu skip-link. Käyttäjä joutuu joka sivulla käymään läpi koko navigaation päästäkseen sisältöön.

4.1 HTML

```
<!-- Sivun ensimmäinen elementti <body>:n jälkeen -->
<a href="#main-content" class="skip-link">
Siirry sisältöön
</a>

<main id="main-content" tabindex="-1">
<!-- Sivun sisältö -->
</main>
```

4.2 CSS

```
.skip-link {
position: absolute;
top: -40px; /* Piilotettu näkymästä */
left: 0;
padding: 8px;
background: #000;
color: #fff;
z-index: 100;
}

.skip-link:focus {
top: 0; /* Tulee näkyviin Tab-painalluksella */
outline: 3px solid #000;
outline-offset: 2px;
}
```

4.3 Selitys

- Linkki on näkymätön oletukseen (top: -40px)
- Tulee näkyviin kun käyttäjä painaa Tab ensimmäisen kerran
- Hyppää suoraan <main id="main-content">-elementtiin
- tabindex="-1" mahdollistaa programmaattisen fokuksen <main>-elementtiin

5. Korjauksien yhteenveto

Korjaus	Teknologia	WCAG	Aikaa
1. Tuotevalinnat	HTML: input[type=radio] + label + CSS	4.1.2	~1 päivä
2. Hinnat	JSX: aria-live + aria-atomic on	4.1.3	~2 h
3. SPA fokus	React: useEffect + usePathname + blur	2.4.1	~2 h
4. Skip link	HTML + CSS position absolute	2.4.1	~1 h