Inlämningsuppgift 1: CV i HTML

Introduktion

I den första uppgiften ska vi fokusera *enbart* på HTML.

- HTML är grundbulten i all webbutveckling.
- Det är *oerhört viktigt* att vi använder rätt taggar på rätt sätt för att:
  - Tillgänglighetsanpassa webbplatser
  - Sökmotoroptimera webbplatser
  - Framtidssäkra webbplatser
  - Göra utvecklarupplevelsen (DX = developer experience) bra
  - Kunna skriva optimal CSS senare
  - Kunna skriva optimal JavaScript senare

Tips

Ofta betraktar vi HTML som någonting "lätt" och enkelt, och det är väl bara att skriva en " `<div>` " och så lite CSS på det. Men i och med gällande EU-direktiv och lagkrav, så behöver vi förstå och använda HTML på rätt sätt.

▶ **Lär dig mer om digital tillgänglighet**

Uppgiftsbeskrivning

I den här uppgiften ska du skapa ett CV med enbart HTML. CV:t ska inte innehålla någon CSS eller JavaScript, endast HTML.

🚨 **Viktigt!**

Du startar (alla) inlämningsuppgifter via länken du hittar under "Länkar"-sidan.

Översikt arbetsmoment

1. Skapa wireframe
2. Gör din första commit
3. Koda upp CV:t med HTML
4. Sökmotoroptimera
5. Validera

▶ **1. Skapa wireframe**

▶ **2. Commit:a**

▶ **3. Koda upp sidan**

▶ **4. Undvik vanliga misstag**

▶ **5. Sökmotoroptimera**

▶ **6. Validera**

▶ **Arbetsflöde – exempelvideo**

Moment som examineras från kursplanen

Kunskaper

- Syntaxmässigt korrekta HTML-dokument, utifrån gällande rekommendationer för såväl struktur, tillgänglighet som sökmotoroptimering.
- Kunna särskilja på strikt och mindre strikt HTML.
- Kunna använda mallar och ramverk för ~~CSS och~~ HTML där det är lämpligt.
- Vad versionshantering är för något och vad det används för.

Färdigheter

- Genomföra tillgänglighetsanpassningar av webbplatser.
- Genomföra validering av webbplatser avseende såväl tillgänglighet som sökmotoroptimering.

Betygskriterier

För G

- Du har använt syntaxmässigt korrekta taggar för att märka upp innehållet
- Du har gjort en tillgänglighetsanalys på webbplatsen och bifogat rapporten som PDF i inlämningsuppgiften
- Tillgänglighetsanalysen/valideringen innehåller inga fel eller varningar (möjligtvis vissa undantag)
- Du har använt conventional commits i stora delar av din versionshistorik
- Du har lagt till en del sökmotoroptimering

För VG

- Utöver kraven för G...
- Din versionshistorik innehåller små commits och en konsekvent logg
- Du följer "best practice" i hur HTML-dokumentet ska struktureras
- Formatering och indentering av dokumentet håller hög nivå
- Bilder innehåller alt-text och storlek
- Du har ansträngt dig lite extra för att fixa till sökmotoroptimeringen och reflekterat kring användning av taggar, struktur och extra metadata

Vanliga misstag

Du har lagt filerna under mappen `.github`

"Flytta ut" dina filer så de inte ligger under mappen `.github`, utan på "rot-nivå".

Alt-texter på bilder

Alt-texter på bilder: beskriv bilden för någon som inte *ser* den. T.ex. säger "Bild på mig" ingenting, jämfört med "Leende person med glasögon och guldfärgat hår som står på en klippa framför ett hav.". Läs mer här

Vidare läser en "skärmläsare" upp "Picture of" (eller motsvarande), så om din alt-text är "Profile picture of me" så blir uppläsningen "Picture of profile picture of me".

Om bilden är dekorativ och ingår i en länk som innehåller text så behöver bilden ingen text, exempelvis såhär:

```html
<<a href="https://link.com" target="_blank" rel="noopener noreferrer">Läs
```

ID-attribut

Du ska *aldrig* sätta attributet `id` på ett HTML-element om du inte explicit kan motivera varför det ska finnas där.

```html
<div id="section"></div> <!— ❌ NEJ! —>
```

▶ **Mer information**

Extra radbrytningar

Radbrytningar med `<br>` ska inte finnas med. Avstånd fixas sedan med CSS senare i kursen. Endast t.ex. om du behöver radbryta brödtext eller rubrik av någon specifik

anledning. Men absolut aldrig två br: `<br><br>` ! Överväg också att låta webbläsaren sköta radbrytningarna, så att texten flödar korrekt. Om du behöver begränsa textens bredd för läsbarhet, så görs detta också med CSS senare.

Om inputs och labels

Tips: om du lägger din input inuti en label, så behöver du inte ha med något id:

```html
<label>
  <span>Ditt namn</span>
  <input type="text" placeholder="För- och efternamn">
</label>
```

Jämfört med `for` och `id` :

```html
<label for="name">Namn</label>
<input id="name" type="text" placeholder="För- och efternamn">
```

Vilket sätt som är bäst beror på layouten/designen, men oftast funkar det bra med att ha `<input>` nästlad inuti `<label>` .

Tänk på att du har gjort din koppling rätt först när `<label>` -texten går att klicka på, och input-rutan fokuseras automatiskt!

<b> vs.

Fundera på när `b` -taggen ska ersättas med `strong` -taggen, liksom `em` med `i` : https://developer.mozilla.org/en-US/docs/Web/HTML/Element/b#usage_notes

Telefonnummer

Skriv alltid telefonnummer med landskod, så underlättar du uppringning via mobil: `<a href="tel:+46701234567">070 123 45 67</a>`

Gemener

Alla HTML-taggar ska skrivas med gemener. Jämför `<Section>` med `<section>` .

Öppna externa länkar i ny flik

Lägg gärna till attributet `target="_blank"` på länkar som leder till en extern sida (t.ex. LinkedIn), så behåller du besökaren på din sida och sidan öppnas i en ny flik, enligt konvention.

För ökar säkerhet och "minimerad tracking" kan du lägga till följande attribut:

```html
<a href="https://link.com" target="_blank" rel="noopener noreferrer">Titl
```

Mer, mycket svårläst info här

Bildnamn för ökad sökmotoroptimering

Tips! Döp filnamnet till hela ditt namn, t.ex. `fornamn_efternamn.jpg` så ökar sökmotorsynligheten på såväl sidan som bilden. Om det är önskvärt. Obs - ej krav.

Förväxla inte `<dl>` med `<li>`

`<dl>` -taggen är mer lämpad att använda för t.ex. ett "index" (tänk dig ett receptregister i en kokbok) snarare än för att lista utbildningar/arbetslivserfarenhet. Läs mer här: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/dl

Du saknar alla nödvändiga element på en bild

En bildfil ska alltid ha följande attribut:

```html
<img
    src="bild.jpg"
    width="200"
    height="200"
    loading="lazy"
    alt="Beskrivande text på bildens innehåll för någon som inte ser">
```

- Bredd och höjd ska alltid sättas för att undvika att "sidan hoppar" vid inladdning. Exakt storlek kan styras med hjälp av CSS senare, så sätt bildens verkliga storlek.

T.ex. om din bild är 640x480 px, så skriver du `width="650" height="480"`. Observera att vi *inte* skriver med måttenheten `px` i attributets värde.

- Du ska *alltid* ha med en alt-text! ALLTID.
- Vill du optimera laddningen på sidan kan du lägga till `loading="lazy"` som gör att bilden laddas in först när den blir synlig i webbläsarens "viewport", dvs. när användaren har scrollat till bilden.
- Bilder ska aldrig innehålla specialtecekn (å, ä, ö, !, osv.) och aldrig blanksteg/space. Ersätt blanksteg med t.ex. `_`.

Läs mer om s.k. "Cumulative Layout Shift" (CLS) och dess inverkan på "performance" på [web.dev]

Om du har en länk med text, som också innehåller en bild, då behöver bilden inte en alt-text för då är bilden endast "dekorativ". Exempel:

```html
<a href="somepage.html">Läs mer <img src="arrow.png" width="20" height="1
```

Namngivning av CSS-klasser (out of scope för uppgiften)

Sätt CSS-klasser också med gemener, och helst engelska: `Bild -> image`. Det underlättar t.ex. i internationella teams för att skapa konsekvens i namngivningen i koden.

Submit-knapp

I formulär, överväg att byta ut `<button>` mot en `<input type="submit" value="Send">` för att "skicka" formulär, och använd på motsvarande sätt `<input type="reset">` för att nollställa formulär.

Gemener i texten

Skriv rubriker och annat med gemener (bortsett från stor begynnelsebokstav). Vi sätter versaler i CSS sedan. Det tenderar att bli "skrikigt" i texten med versaler, även om jag förstår syftet med att ha versaler redan i HTML, men det är inte praxis.

```html
<h1>OM MIG OCH MITT LIV</h1> <!— ❌ NEJ! —>

```

```
<h1>Om mig och mitt liv</h1> <!— Ja! —>
```

Din huvudfil/"landningssida" ska heta `index.html`

Webbservern letar automatiskt efter en fil som heter `index.html` som förstasida att visa, så döp din "huvudfil" till just `index.html` och ingenting annat.

Fel typ på input-fält

Använd rätt typ på input-fält, ska du t.ex. skriva ett formulärfält för e-post:

```html
<input type="text" placeholder="Email" name="email"> <!— ❌ NEJ! —>
<input type="email" placeholder="Email" name="email"> <!— Ja! —>
```

Referens

Hoppa rubriknivåer

Hoppa inte över rubriknivåer, dvs. du ska inte ha såhär exempelvis:

```html
<h1>Mitt namn</h1>
<h3>Min titel</h3>
```

Utan prioriter att ha rubrikerna i nummerordning:

```html
<h1>Mitt namn</h1>
<h2>Min titel</h2>
```

Storleken kan du fixa senare i CSS, om det är anledningen till att du valt en mindre rubriknivå än nästkommande siffra.

Oformaterat dokument

Indentera ditt dokument korrekt. I VSCode kan du t.ex. använda den automatiska inbyggda formateringen för att få till det. I Command Palette, skriv "Format Document" så

får du ordning på tabbningen. Se instruktioner här

Filnamn

Undvik internationella tecken (å, ä, ö) i filnamn liksom specialtecken (exempelvis ! eller #). Ersätt med `a`, `o` och `_` eller `-`.

Senast uppdaterad: 2024-10-27 19:14

Föregående sida
Prepp-uppgift 1

Nästa sida
Inl. 2: Kundprojekt