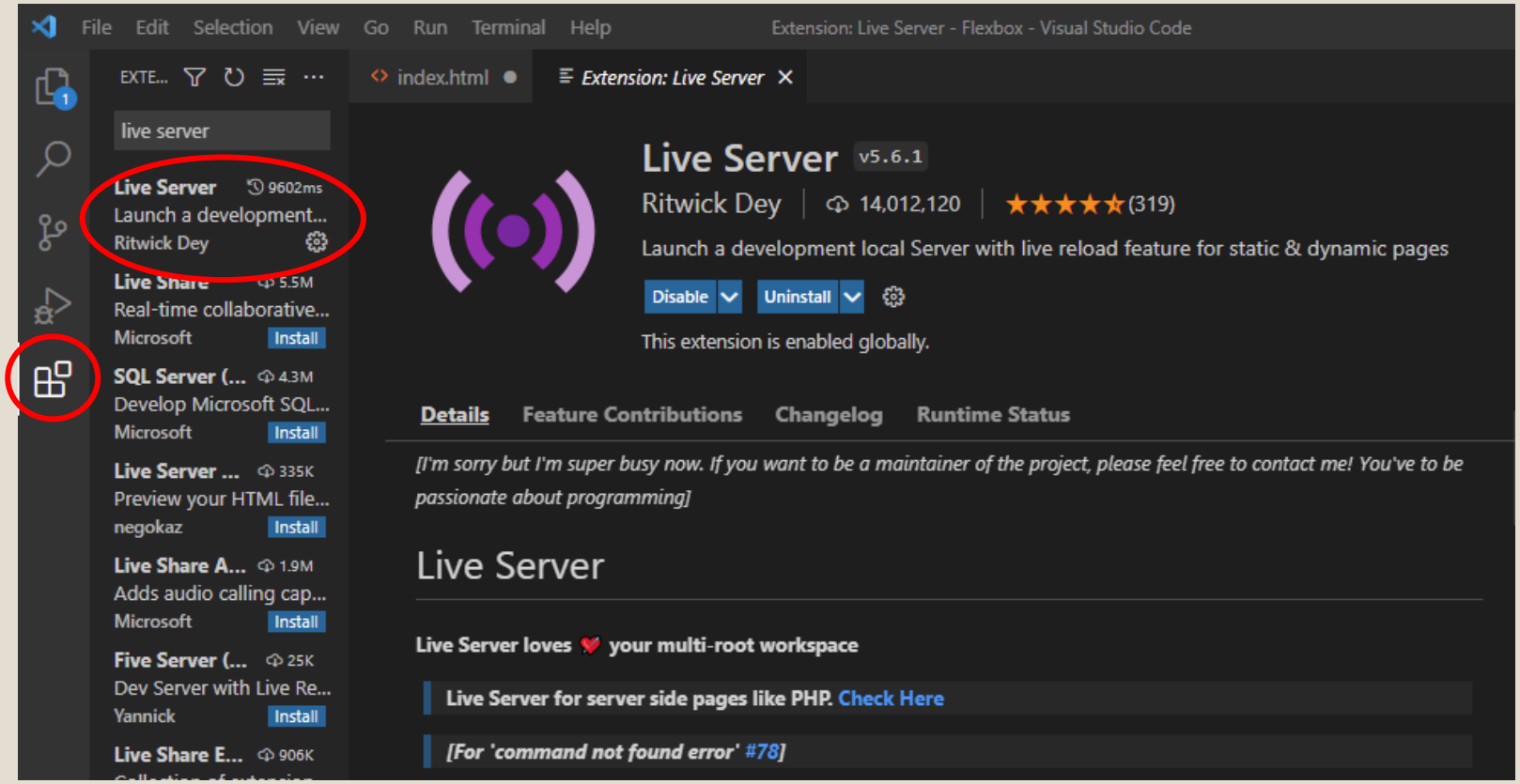
PROGRAMMERING FÖR UX-PRODUKTION 04

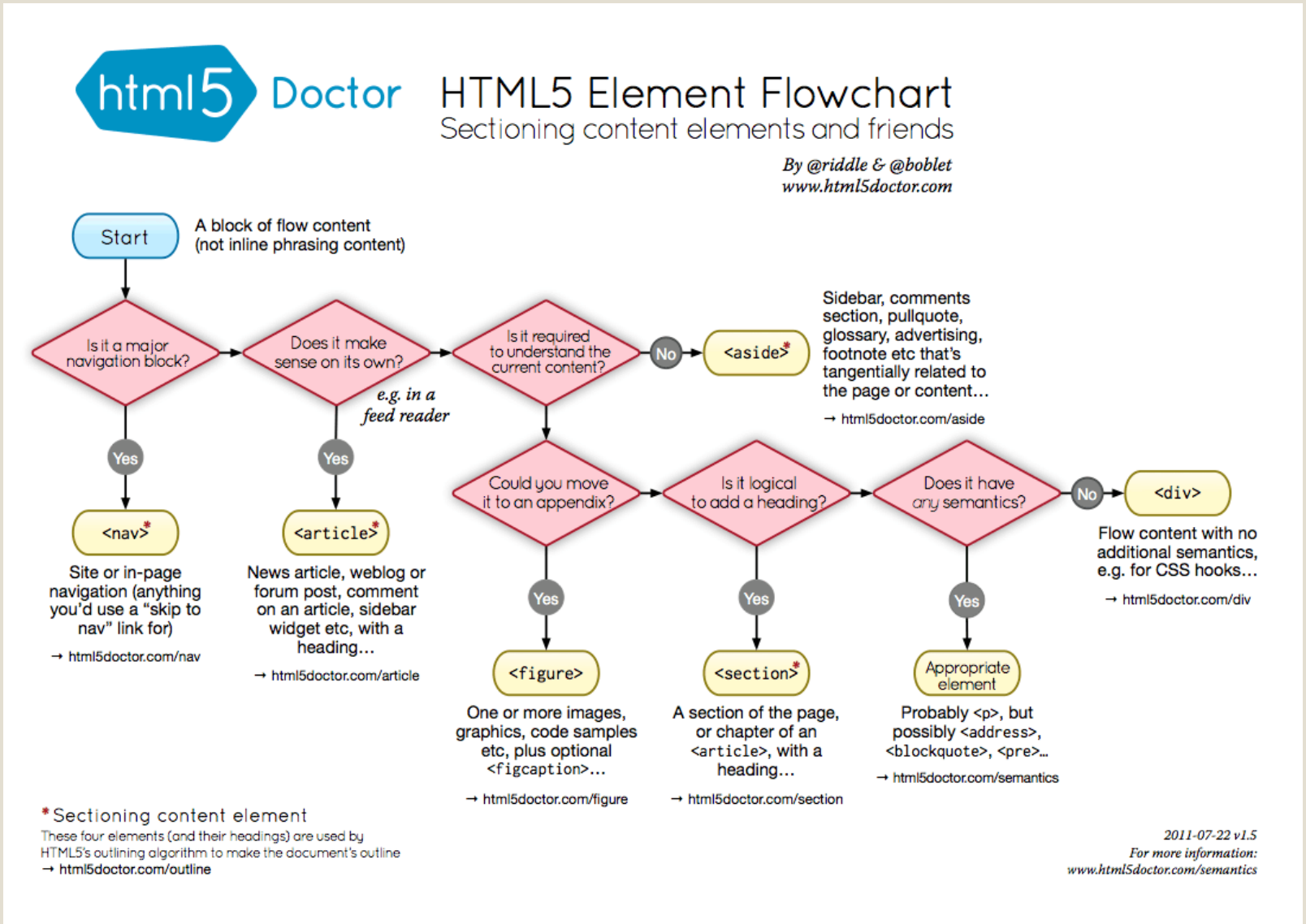
Mer HTML och CSS

Tips VS Code – Live server



Semantiska taggar

- <article>
- <aside>
- <details>
- <figcaption>
- <figure>
- <footer>
- <header>
- <main>
- <mark>
- <nav>
- <section>
- <summary>
- <time>
  - M fl


html5 Doctor
HTML5 Element Flowchart
Sectioning content elements and friends
By @riddle & @boblet
www.html5doctor.com
Start
A block of flow content
(not inline phrasing content)
Is it a major navigation block?
Yes
<nav>*
Site or in-page navigation (anything you'd use a "skip to nav" link for)
→ html5doctor.com/nav
Does it make sense on its own?
e.g. in a feed reader
Yes
<article>*
News article, weblog or forum post, comment on an article, sidebar widget etc, with a heading…
→ html5doctor.com/article
Is it required to understand the current content?
No
<aside>*
Sidebar, comments section, pullquote, glossary, advertising, footnote etc that's tangentially related to the page or content…
→ html5doctor.com/aside
Could you move it to an appendix?
Yes
<figure>
One or more images, graphics, code samples etc, plus optional <figcaption>…
→ html5doctor.com/figure
Is it logical to add a heading?
Yes
<section>*
A section of the page, or chapter of an <article>, with a heading…
→ html5doctor.com/section
Does it have any semantics?
Yes
Appropriate element
Probably <p>, but possibly <address>, <blockquote>, <pre>…
→ html5doctor.com/semantics
No

Flow content with no additional semantics, e.g. for CSS hooks…
→ html5doctor.com/div
*Sectioning content element
These four elements (and their headings) are used by HTML5's outlining algorithm to make the document's outline
→ html5doctor.com/outline
2011-07-22 v1.5
For more information:
www.html5doctor.com/semantics

Debugga CSS

- Se alla elements kantlinjer

```
*{
        border: 1px dashed red;
 }
```

- Ta reda på om en selektor matchar något

```
selektor {
        background-color: hotpink;
 }
```

- Gör alla elementet halvt genomskinligt, se det som är under

```
*{
        opacity: 0.8;
 }
```

Positionering

När webbläsaren *renderar* en sida så får alla element en given position. *Positionering* handlar om att flytta elementet till en annan plats, än där de ursprungligen ska hamna.

Standardpositionen har värdet *static*. Det är där webbläsaren placerar elementet om man inte anger något annat.

Använd left, top, right, bottom för att ange den nya positionen.

```
.no-positioning {
        position: static;
}
```

Postionering - relativ

Relative flyttar elementet i förhållande till sin standardposition.

```
.stay-still {
	position: relative;
}

.move-right {
	position: relative;
	left: 2em;
}
```

em = Relative to the font-size of the element (2em means 2 times the size of the current font)

CSS Units (w3schools.com)

Positionering - absolute

Absolute plockar ut elementet från sidan och sätter in det igen, direkt i sin *närmast positionerade föregångare*. Element som kommer efter, fyller igen hålet.

```
.container { position: relative; }
.moved { position: absolute; }
```

```
<div class="container">
       <div> 1 </div>
       <div class="moved"> on top of 2 </div>
       <div> 2 </div>
</div>
```

Positionering - Fixed

Ett fixed element plockas ut från sidan precis som absolute. Eterföljande element fyller igen hålet. Elementet sätts tillbaka direkt på *viewport*. När man scrollar kommer ett fixed element att vara kvar på samma plats.

```
.never-moves {
	position: fixed;
}
```

Demo - Positionering

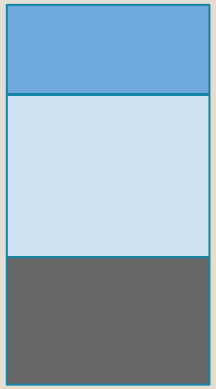
- Kod: positionering 1 och positionering2

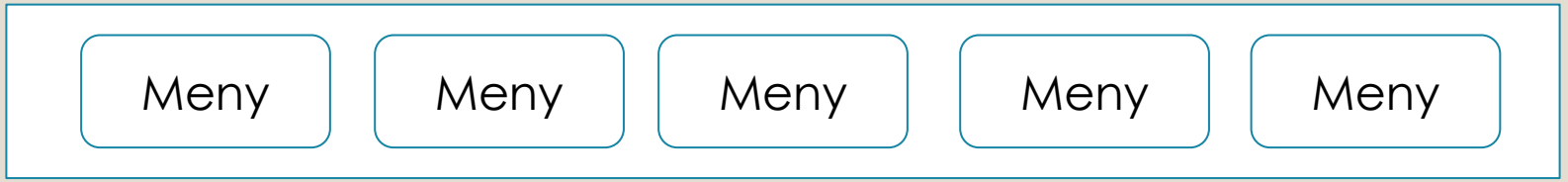
Flexbox

CSS flexible box layout lägger ut element i en riktning: antingen horisontellt eller vertikalt.

Används till:

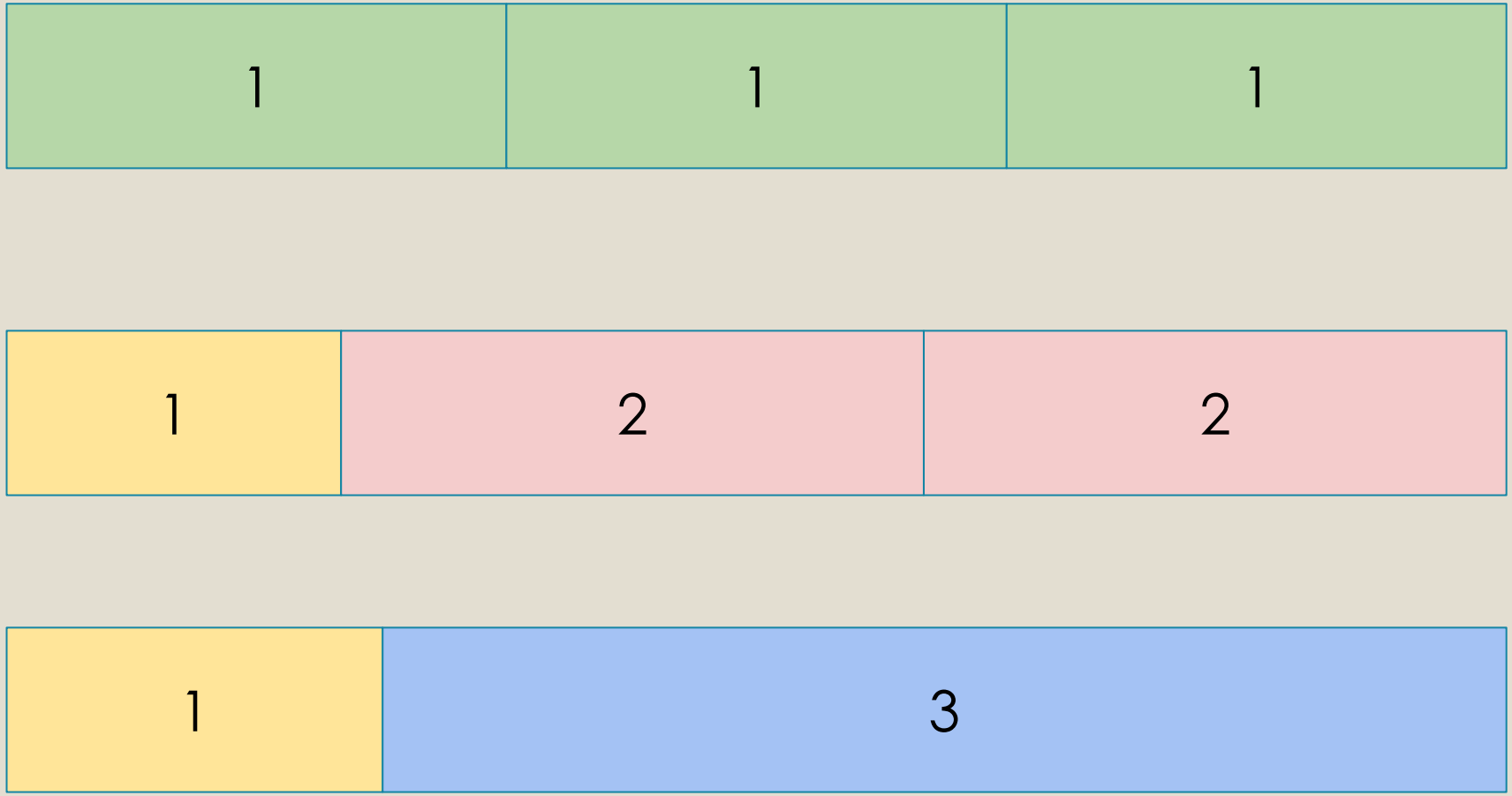
- vertikal layout med header, main och footer
- horisontell menyrad





Flexbox - dela på utrymmet, flex-basis

Flexbox - dela på utrymmet, flex-grow



Flexbox - dela på utrymmet

```
<div class="flex-horiz-container">
       <div class="left"> left </div>
       <div class="rest"> the rest </div>
</div>

.flex-horiz-container {
       display: flex;
       flex-direction: row;
}
.left { flex-basis: 25vw; }
.rest { flex-grow: 1; }
```

Demo - Flexbox

- Kod: flex1 och flex2

Grid

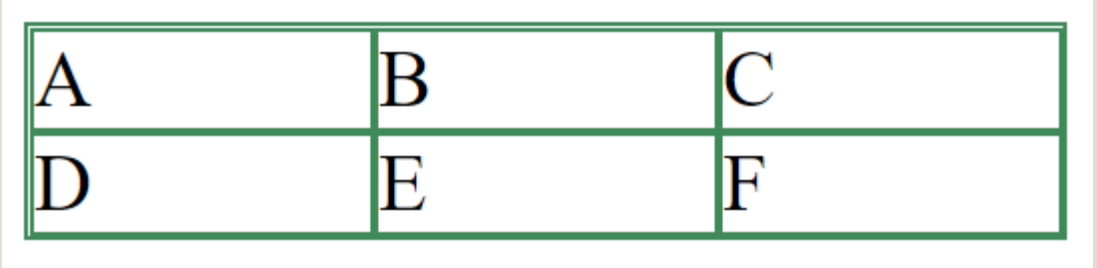
- *CSS grid layout* lägger ut element i två dimensioner.
- grid = "rutnät"
- Exempel på använding:
  - flera kolumner på en webbsida (=de flesta webbsidor)
  - visa produkter i en webshop

- En grid-layout måste ha ett överordnat element med visningsegenskapen inställd på grid eller inline-grid.
- Direkta underordnade element i rutnätsbehållaren blir automatiskt rutnätsobjekt.
- Väldigt...väldigt många olika sätt att göra samma sak!

Exempel

HTML

```
<div class="grid-container">
        <div> A </div>
        <div> B </div>
        <div> C </div>
        <div> D </div>
        <div> E </div>
        <div> F </div>
</div>
```



CSS

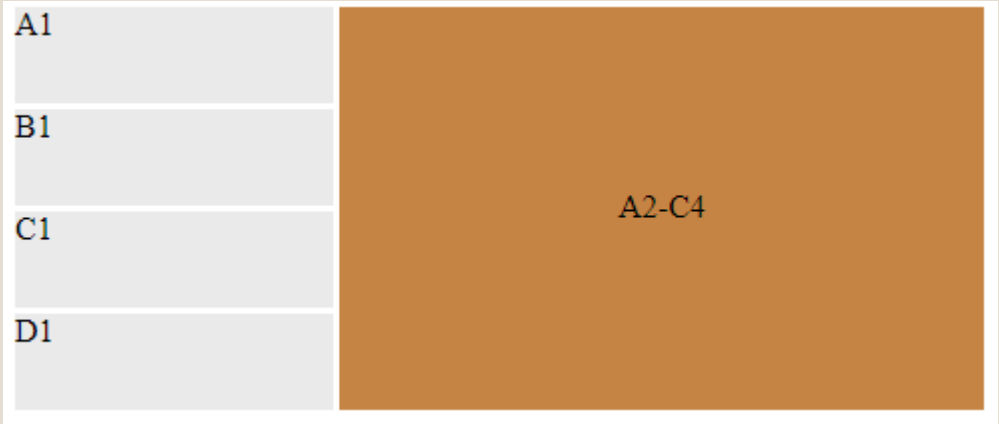
```
.grid-container {
        display: grid;
        grid-template-columns: 1fr 1fr 1fr;  Ger tre kolumner
}
```

Grid med olika storlekar

```
<div class="grid-container">
        <div> A1 </div>
        <div class="large"> A2-C4 </div>
        <div> B1 </div>
        <div> C1 </div>
        <div> D1 </div>
</div>
```

```
.grid-container {
        display: grid;
        grid-template-columns: 1fr 1fr 1fr;
}
```

```
.large {
        grid-column: 2 / span 2
        grid-row: 1 / span 4
}
```



Media Queries

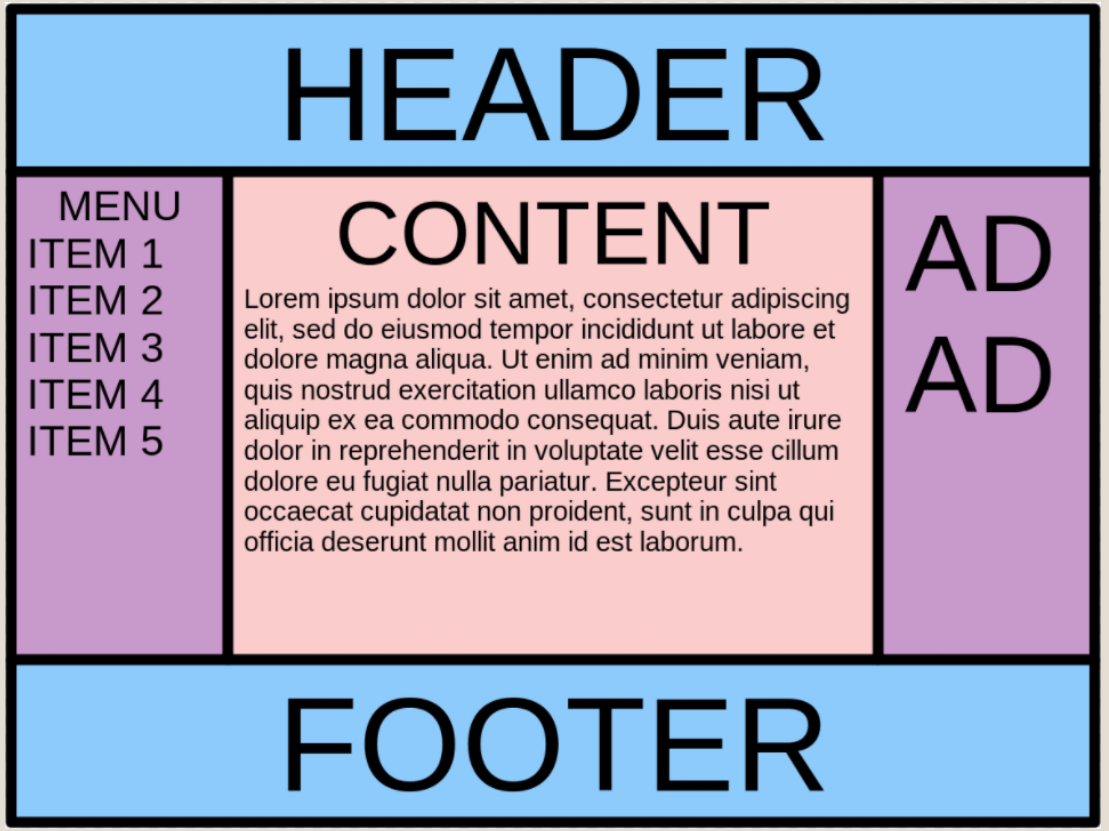
Används för att låta olika CSS-regler gälla för olika skärmbreddar. Ändra storleken på webbläsarens fönster för att testa.

```
@media (max-width: 600px) {
    main {
                background-color: hotpink;
    }
}

@media (max-width: 300px) {
    main {
                background-color: lime;
    }
}
```

Media queries

- Det går också att bestämma olika typer av devices:

The holy grail



The holy grail - HTML

```
<div class="grid">
        <header>Header</header>

        <aside class="sidebar-left">Left Sidebar</aside>

        <article>Article</article>

        <aside class="sidebar-right">Right Sidebar</aside>

        <footer>Footer</footer>
 </div>
```

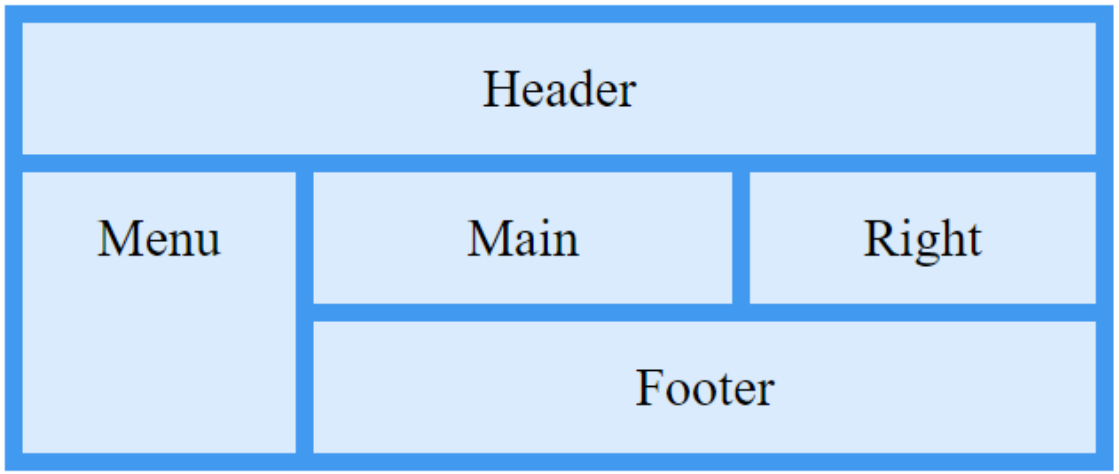
The holy grail - CSS

```
.grid {
        display: grid;
        grid-template-columns: 10vw auto 10vw;
        grid-gap: 0.1em;
    }

    header, footer {
        grid-column-start: 1;
        grid-column-end: 4;
    }

    @media all and (max-width: 450px) {
    aside, article {
            grid-column-start: 1;
            grid-column-end: 4;
        }
    }

    body {
        margin: 0;
        padding: 0;
}

header, aside, article, footer {
    background: #eaeaea;
    display: flex;
    align-items: center;
    justify-content: center;
    height: 25vh;
}

aside, article{
    padding: 1vw;
}


.sidebar-left{
    background-color: steelblue;
}
.sidebar-right{
    background-color: papayawhip;
}
```

Grid-Area

- Ett mycket flexibelt sätt att definiera hur layouten på sidan ska se ut.

- Består av tre delar:
  - Ett antal DIV-ar med innehåll och ett class-namn (`<div class="item1">Header</div>`)
  - En referens till ett area-namn, via property grid-area (`.item1 { grid-area: header; }`)
- GRID-css där ordning och placering bestäms av area-namnet, via grid-template-areas.
  - `grid-template-areas:`
    - `'header header header header header header'`
    - `'menu main main main right right'`
    - `'menu footer footer footer footer footer';`
- Regler: Understanding CSS Grid: Grid Template Areas



Demo - Grid

- Kod: enkel1, enkel2, olika, holygrail, grid-i-grid och grid-area

Demo – Intro Formulär

- Förmulär används för att låta användaren mata in data/uppgifter.
- Kod: formexempel.html

Uppgifter

- Sedan förra gången:
  - Profil-sida, som lämnas in som Github Pages-sida under torsdagen.
- Idag:
  - Gruppuppgift: **Projekt 2:Meny**
    - I Projekt2: Meny, så ska ni välja en befintlig restaurang, som ni ska skapa en liten hemsida till.
    - Hemsidan ni skapar ska "passa" restaurang stil/tema så gott ni kan.
    - Kraven för att klara uppgiften:
      - -Minst 2 html sidor, en startsida med lite "flash" och en med någon meny för maten
      - -Ska skala efter webbläsarens storlek, med hjälp av flexbox och/eller grid.
      - -Ska utnyttja Html5 taggarna Bonusuppgift 1 är att ni ska anpassa och ändra menyn(både matmenyn och navigeringsmenyn) för att vara optimal för mindre skärmar/mobil.
      - Dessutom ska det finnas ett formulär, med möjlighet att boka bord. Kalenderdatum, Namn och antal personer ska kunna anges
      - Bonusuppgift: Lägg till en "Parallax Scrolling" effekt (https://jolly-kalam-23776e.netlify.app/parallaxsite/)
      - Ni ska lämna in uppgiften som en Github pages länk, och ange också vilken restaurang ni valde, med länk till deras hemsida.
  - Börja med att välja resturang, kolla befintlig hemsida, och gör sedan er egen webbsida, helt utan förutfattade meningar. Inspireras enbart av innehållet på restaurangens egna sida.

Länkar

- CSS Grid Layout (w3schools.com)
- A Complete Guide to Grid | CSS-Tricks
- Grid template areas - CSS: Cascading Style Sheets | MDN (mozilla.org)
- CSS Media Queries (w3schools.com)
  - CSS3 Media Queries - Examples (w3schools.com)
- CSS Flexbox (Flexible Box) (w3schools.com)
- Understanding flex-grow, flex-shrink, and flex-basis | CSS-Tricks