Engineer

CSS

språket som sier hvordan en side ser ut

Hva det er

CSS, kort for Cascading Style Sheets og fortsatt ofte kalt CSS3, er regelspråket en nettleser bruker på et dokument for å avgjøre hvordan hvert element tegnes: farger, størrelse, avstand, plassering, skrifttype og bevegelse, enten dokumentet havner på en telefon, en bred skjerm eller et ark papir.

Et stilark er en liste med regler. Hver regel kobler en selektor, som plukker ut elementer, med deklarasjoner, der hver deklarasjon er en egenskap og en verdi:

```
.note {
    color: #3a3a3a;
    margin-block: 1.5rem;
}
```

Det første ordet i navnet er den delen som former det daglige arbeidet. Flere regler gjør rutinemessig krav på den samme egenskapen for det samme elementet, og kaskaden (*cascade*) avgjør hvilken som vinner. Hovedtrinnene, i rekkefølge, er opphav og viktighet (nettleserens standardverdier, leserens egne innstillinger, nettstedets stilark og om en deklarasjon er merket `!important`), deretter kaskadelag, deretter spesifisitet og til slutt rekkefølge, der en senere regel slår en tidligere med samme vekt. Spesifisitet telles per selektor som tre tall: ID-er, deretter klasser, attributter og pseudoklasser, deretter elementnavn. Den universelle selektoren `*` legger ikke noe til noen av dem. Arv virker ved siden av kaskaden: noen verdier, som farge og skrift, går videre til barna til et element, mens andre, som marger og rammer, blir der de ble satt.

CSS har ikke ett enkelt versjonsnummer. Etter CSS 2.1 gikk arbeidsgruppen over til moduler: Level 3 bygger på 2.1 én modul om gangen, og derfra får hver modul sine egne nivåer, så Selectors er et utkast på Level 4, mens moduler uten noen

eldre motpart starter på Level 1. «CSS3» er kallenavnet på den modulære epoken, og det årlige øyeblikksbildet (*snapshot*) som definerer dagens CSS, sier rett ut at det ikke finnes noen CSS Level 4.

Regelen med størst praktisk vekt er hvordan språket behandler feil. En nettleser som møter en egenskap den ikke kjenner, eller en verdi som er ugyldig for egenskapen sin, må overse den ene deklarasjonen og bruke resten av regelen. Regelen finnes slik at nye egenskaper og verdier kan legges til uten å knekke eldre nettlesere, og det er den som lar et nettsted bruke en funksjon som bare noen nettlesere støtter. Prisen er taushet. Siden tegnes, litt feil, og det eneste sporet er en gjennomstreket linje i utviklerverktøyene som noen må gå og lete etter.

Det meste som sammenlignes med CSS, viser seg å være en måte å skrive det på. Sass og Less er preprosessorer som kompilerer til CSS. Verktøyrammeverk (*utility frameworks*) som Tailwind genererer det ut fra klassenavn i oppmerkingen. CSS-in-JS-biblioteker lager det fra JavaScript-komponenter, og et `style` -attributt setter deklarasjoner rett på ett element. Hver av dem endrer hvordan CSS skrives, og hver ender som deklarasjoner kaskaden sorterer på nøyaktig samme måte.

Bruk begrepet i en beslutning

Bruk «Hva er CSS?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Grensesnitt som laster raskt, fungerer på mobilen og oppfører seg likt i alle nettlesere kundene dine måtte bruke.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Styrte fullstack-utviklingen av GIS-kart — fullstack-kartutvikling, fra database til nettleser

Styrte fullstack-utviklingen av GIS-kart Ta kontakt →

Hvorfor det er viktig

Beslutningen CSS tvinger et team til å ta, er hvor en regel får lov til å svikte i det stille. Språket dropper det det ikke kan bruke, og lar en sterkere regel vinne uten kommentar, så hvert stilark av en viss alder inneholder deklarasjoner som ikke gjør noe. Det et team rår over, er om noen oppdager det, og når.

Tilgjengelighet avhenger av det mer enn de fleste regner med. Kontrast, en synlig fokusring, å respektere et ønske om redusert bevegelse, en brukbar modus for tvungne farger, tekst som tåler å bli forstørret: hver av dem er en

håndfull deklarasjoner, og hver kan slås av av en regel et annet sted som tilfeldigvis er mer spesifikk. Siden ser fin ut for alle som ikke trengte funksjonen.

Vedlikeholdskostnaden avhenger også av det. Et stilark som ingen kan forutsi oppførselen til, bare vokser, fordi å slette en regel kan ødelegge en side ingen sjekket. Den vanlige sluttilstanden er en fil der mesteparten av vekten er overstyringer av tidligere overstyringer, og hver endring gjøres ved å prøve verdier til skjermen ser riktig ut.

Verifiseringen avhenger av det mest direkte. En nettleser som godtar det den kan, og overser resten, kan ikke fungere som testen. En linter som leser kildekoden, fanger skrivefeil og ugyldige verdier før de sendes ut. Den kan ikke se hva en minifiserer eller et annet byggetrinn gjør med filen etterpå, og den kan ikke se hvilken av to gyldige regler som vinner på en gitt side. Kontroller mot den leverte filen og visuell sammenligning av gjengitte sider dekker det linteren ikke kan.

Å organisere kaskaden tidlig, med flat spesifisitet, navngitte lag og designtokens holdt i egendefinerte egenskaper (*custom properties*), koster litt disiplin per endring. Å organisere om et stilark som vokste uten den disiplinen, koster som regel en omskriving.

Slik ser det ut i praksis

Kolofonen til dette nettstedet beskriver én CSS-pakke, satt sammen og merket med et fingeravtrykk ved bygging, med de håndtegnede pilene og margnotatene tegnet med CSS alene, og med Stylelint som holder kildekoden til reglene sine i kvalitetsporten. Den samme siden lenker til W3C CSS Validator og sier rett ut hva leseren kan vente seg: «Regn med at den rapporterer feil, og les dem før du tror på dem». Per august 2026 stopper profilen dens før `system-ui`, `@property`, `clamp()` i en `font-size` og scrollstyrte animasjoner, så mot /terms/ rapporterte den 176 innvendinger, og 175 av dem gjaldt funksjoner nettleserne leverer og nettstedet bruker. Den siste var reell. En `rotate: 0deg` i hviletilstand mistet enheten sin i minifiseringen, `rotate: 0` er ikke en vinkel, og nettleseren forkastet hele deklarasjonen, så en animasjon ble levert uten rotasjonen den var skrevet med. Verdispesifikasjonen tillater en naken null for en vinkel bare på noen få eldre steder. Stylelint så aldri problemet, fordi den leser kildekoden og minifisereren skriver den om senere, og som kolofonen sier, leser «en egen kontroll» nå «det som faktisk ble levert».

Spesifisitet var hele historien i et stykke tilgjengelighetsarbeid. Nettstedet respekterte innstillingen for redusert bevegelse med én universell regel som slo av alle overganger, og med en emulert innstilling animerte elleve selektorer fortsatt. En universell selektor gir null poeng, så enhver regel med en klasse slo den. Rettelsen var strukturell og ikke en sterkere selektor: alt som beveger seg, ligger nå i én sen del av stilarket, og en linter avviser en transform med overgang alle andre steder. Fargeoverganger består, siden de ikke er bevegelse, og bevegelse må skrives som en overgang og ikke som en keyframe-animasjon, fordi det er overgangen som lar seg avbryte rent. Lærdommen som er skrevet ned sammen med arbeidet, er kort: «en regel som taper på spesifisitet, feiler i det stille».

Farge er der stilark glir ut først, og nettstedets palett ble lukket ved 28 farger, hver dokumentert med kontrastforholdet den måler på flaten den står på, og elleve merkevareverdier ble deklarert én gang som tokens. To farger som ligger nærmere hverandre enn en perseptuell avstand på 0,02, regnes som én farge med to skrivemåter, og linteren fant to nesten like skrivemåter så snart den kjørte. Gjennomsiktighet skrives med relativ fargesyntaks i stedet for blandingsfunksjonen, og grunnen oppsummerer hele siden: det er «nettopp fordi vakten ikke kan se gjennom en blanding».

Hvor det går galt

En deklarasjon forsvinner, og siden tegnes likevel

Ett galt tegn gjør en deklarasjon ugyldig, og nettleseren hopper over den uten å stoppe:

```
.card {
    padding: 16; /* a length needs a unit: ignored */
    colour: #222; /* not a property name: ignored */
    margin-block: 1rem; /* applied */
}
```

Kortet tegnes med en marg, men uten padding og farge, og nettleseren sier ikke fra. Kjør en linter over kildekoden i byggingen, og la byggingen stoppe på feilene den finner. Der en minifiserer, en preprosessor eller en bundler skriver om CSS-en etter at linteren har kjørt, sjekker du også resultatet, fordi det er den filen nettleseren leser.

En regel taper mot en sterkere

Regelen til en komponent møter en mer spesifikk selektor fra et annet sted, eller en like sterk som lastes senere, og taper. Den vanlige refleksen er en lengre selektor eller en `!important`, som vinner i dag og gjør neste endring dyrere. Hold spesifisiteten flat, med klasser i stedet for ID-er og lite nesting, og bruk kaskadelag for å sette hele grupper, som reset, base, komponenter og verktøyklasser, i en uttrykkelig rekkefølge som spesifisitet ikke kan velte. For `!important`-deklarasjoner går lagrekkefølgen motsatt vei, slik at det tidligste laget vinner, og det er verdt å vite før du griper til den.

Stilarket bare vokser

Nytt arbeid legger til regler, fordi det tar lengre tid å bevise at en gammel regel er ubrukt, enn å skrive en ny. Verdier skrives inn på nytt som litteraler ved siden av tokenene som definerer dem, og nesten like farger formerer seg til ingen kan si hvilken som er ment. Hold felles verdier i egendefinerte egenskaper, mål hvilke regler et sett med sider faktisk bruker, og slett som en del av arbeidet, med et størrelsesbudsjett i byggingen som gjør veksten synlig.

Oppsettet holder bare i én størrelse

Et design som er sjekket i én bredde med kort engelsk tekst, brekker under lengre oversettelser, forstørret tekst og smale telefoner. Faste høyder klipper innholdet sitt, og faste bredder tvinger fram rulling sidelengs. Retningslinjene for tilgjengelighet ber om tekst som kan forstørres til 200 %, og om innhold som brytes om i en bredde på 320 CSS-piksler. Gi boksene størrelse ut fra innholdet, bruk `min()`, `max()` og `clamp()` for flytende verdier, og test med lange strenger og zoom før du kaller et oppsett ferdig.

Les mer

- CSS Snapshot 2026 — hvilke moduler som utgjør CSS i dag, og hvorfor språket har nivåer i stedet for versjoner.
- CSS Cascading and Inheritance Level 5 — hele sorteringsrekkefølgen, kaskadelag og arv, trinn for trinn.
- Selectors Level 4 — hver selektor, og hvordan spesifisiteten dens regnes ut.
- Syntax and basic data types — reglene i CSS 2.1 for å overse ukjente egenskaper og ulovlige verdier, med eksempler.
- CSS Values and Units Module Level 4 — lengder, vinkler, `clamp()` og hvor en naken null fortsatt er tillatt.

Ordliste

Velg ut fra deres faktiske rammer

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er CSS?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/css/