Engineer

visningsområde

den delen av en side et vindu faktisk viser

Hva det er

Et visningsområde, i bestemt form visningsområdet og på engelsk *viewport*, er rektangelet en nettleser legger ut et dokument i og maler det inn i: på en skjerm flaten innenfor vinduets egne verktøylinjer, på papir sideområdet på hvert ark. CSS måler mot det hele tiden. Den innledende inneholdende blokken, boksen rotelementet legges ut i, får størrelsen sin fra det, en media query på `width` eller `height` spør om det, og enhetene `vw` og `vh` er hundredeler av det. På en telefon trenger ordet en presisering, også i flertall, og spesifikasjonene har flere å by på: layout, visuelt, lite, stort og dynamisk.

På en stasjonær maskin er visningsområdet og vinduet nesten det samme. På en telefon skilles de. Fordi så mange dokumenter ble laget for vinduer på stasjonære maskiner, legger mobilnettlesere ut en side som ikke gir noen instruksjoner, i en fast bredde som etterligner et slikt vindu, ifølge CSS Viewport-spesifikasjonen typisk 980–1 024 CSS-piksler, og skalerer resultatet ned til det får plass på glasset. En side melder seg ut av dette med ett element i `head`, som godtar `width`, `height`, tre grenser for skalering, `user-scalable` og `interactive-widget`:

```
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

Zoom deler visningsområdet i to. Layout-visningsområdet er det CSS legger ut siden mot, og knipezoom lar det være i fred, så linjeskiftene blir stående. Det visuelle visningsområdet er den delen av siden som er på glasset akkurat da: det krymper når leseren zoomer inn, og glir når vedkommende panorerer. Som standard krymper et skjermtastatur bare det visuelle visningsområdet; `interactive-widget=resizes-content` ber nettleseren endre størrelsen på layouten også, og `overlays-content` ber den endre ingen av dem.

Telefoner legger dessuten til verktøylinjer som glir bort. Adressefeltet tar plass når en side lastes, og trekkes inn når leseren ruller, så den synlige høyden endrer seg under en side som ikke har flyttet seg. CSS Values 4 gir tre størrelser navn for dette: det store visningsområdet antar at hver inntrekkbar verktøylinje er trukket inn, det lille antar at alle vises, og det dynamiske følger det som faktisk er tilfellet. Enhetene bærer forleddene, `lvh`, `svh` og `dvh`, med tilsvarende bredder, og vanlige `vh` og `vw` er definert mot det store visningsområdet. Et grensesnitt som legger seg over innholdet uten å flytte det, typisk skjermtastaturet, endrer ingen av disse lengdene.

Bruk begrepet i en beslutning

Bruk «Hva er et visningsområde?» 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

Responsiv design bygger på visningsområdet. Et breddebrytpunkt i et stilark er et spørsmål om bredden, så en side som utelater `meta`-elementet, legges ut i stasjonær bredde på en telefon og krympes til teksten ikke lar seg lese, uansett hvor nøye media queries er skrevet.

Høyden er den vanskelige aksen. Bredden kan endres ved rotasjon eller endring av vindusstørrelsen; høyden kan også endres når verktøylinjen glir, når tastaturet åpnes og når telefonen snus, så alt som tar mål av skjermens høyde, må si hvilken høyde det mener. Det lille visningsområdet gir rom for innhold mens de inntrekkbare verktøylinjene vises, og etterlater en stripe tom plass når de trekkes inn. Det store fyller skjermen og legger innhold bak verktøylinjene mens de vises. Det dynamiske tar hensyn til endringer i verktøylinjenes tilstand, og spesifikasjonen advarer selv om at det kan endre størrelsen på innhold mens leseren ruller, noe som flytter alt som ligger under.

Skjermer har også hjørner. En telefon med avrundede hjørner eller en utsparing for kameraet er ikke et rektangel, og med standardverdien, `auto`, forblir hele siden synlig på den. Å legge til `viewport-fit=cover` ber om at visningsområdet skal være rektangelet som tegnes rundt hele den fysiske skjermen, og gir siden

jobben med å holde teksten unna utsparingen ved hjelp av variablene `env(safe-area-inset-*)`. Disse innfelte avstandene skal være null på en rektangulær skjerm, så én regel for indre marg dekker begge typer.

Fast posisjonering henger også på visningsområdet. En boks med fast posisjon plasseres mot layout-visningsområdet, som CSS Positioned Layout gir samme størrelse som det dynamiske visningsområdet, så bannere om informasjonskapsler, bakteppet bak modale vinduer og menyer i full skjerm står stille mens siden ruller. Hver av dem forutsetter at ingen av de overordnede elementene har tatt over den rollen, og `transform`, `will-change` og `contain` kan gjøre nettopp det.

Utskrift er det visningsområdet et stilark bare for skjerm aldri møter. På papir måler visningsområdeenhetene sideområdet, en media query på `width` spør om sideboksen, og et element med fast posisjon gjentas på hvert ark.

Slik ser det ut i praksis

Sidene på dette nettstedet deklarerer `width=device-width, initial-scale=1, viewport-fit=cover`, så de legges ut i telefonens bredde og går fra kant til kant under et kamerahakk. Stilarket betaler for den andre halvdelen: den indre margen på sidene er `max(1rem, env(safe-area-inset-left))` og tvillingen for høyre side, så teksten holder en rem fra kanten og flyttes lenger inn bare der det finnes en utsparing. På lave skjermer i liggende format hever gulvet seg til `max(2rem, 7vw, env(safe-area-inset-left))`, siden de innfelte avstandene er null på en telefon uten utsparing, og én rem lå for nær kanten.

Seksjoner i full skjerm henter høyden fra en felles variabel satt til `100dvh`, med `100vh` som reserve, og kommentaren ved siden av forklarer hvorfor: `100vh` kan være høyere enn det som vises mens adressefeltet er ute, og hadde latt neste seksjon titte fram under toppseksjonen. To elementer overser variabelen med vilje. Merket i toppfeltet er dimensjonert i statisk `vh`, fordi et merke knyttet til det dynamiske visningsområdet ville endret størrelse når verktøylinjene gled bort, og endret oppsettet i toppseksjonen midt i en rulling. Den faste bakgrunnen på berøringsskjermer er `100lvh` høy, så den aldri endrer størrelse når adressefeltet kommer og går, og den ligger på sitt eget komposisjonslag, fordi nettlesere på berøringsskjermer rev den opp i lyse og mørke bånd når siden rullet videre av egen fart.

I et tidligere forsideoppsett hadde hver tjenesteblokk en minimumshøyde på `100dvh`, slik at den fylte én skjerm så lenge innholdet var kortere enn skjermen. Kommentarene i stilarket, kontrollert i november 2026, beskriver en test ved 852 ganger 393 piksler: innholdet i en blokk ble 412 piksler høyt, og blokkene støtte rett mot hverandre. I det oppsettet kostet indre marg innenfor de `100dvh`, gjennom `box-sizing: border-box`, ingenting på en høy skjerm og åpnet et mellomrom først når innholdet rant over.

På det dokumenterte implementeringsstadiet for kvalitetsporten i august 2026 fulgte testingen media queries. Nettstedets nettleserdrevne kvalitetsport utvidet sjekkene sine til en felles viewport-tabell med tjueto vindusformer, utledet fra hver media query i stilarket, og det alene tok layoutsjekken fra 95 sekunder til 405.

Utskrift avslørte motorene. Utskriftsstilarket dokumenterer at visningsområdeenheter løses opp mot sideboksen i én motor og mot vinduet i en annen, så en side skrevet ut fra et bredt vindu fikk en tredjedel av hver linje kuttet bort i den andre. Stilarket for selve toppfeltet dokumenterer den andre siden av det samme gapet, den som gjelder media queries: utskriften beholdt vinduets retning, høyde og bredde, så ark skrevet ut fra et bredt og et smalt vindu fikk toppfelt med 47 piksler forskjell, inntil `screen and` foran uttrykket begrenset regelen til vinduer.

Fellen med den inneholdende blokken er også dokumentert. Arbeidet med redusert bevegelse fant at fem ulike egenskaper hver for seg gjør et element til inneholdende blokk for alt som er posisjonert relativt til visningsområdet, og at dette i det stille hadde krympet et lukkelag til en brøkdel av skjermen.

Hvor det går galt

Telefonen viser en bitteliten stasjonær side. Uten `meta`-elementet for visningsområdet legger nettleseren ut siden i en stasjonær bredde og skalerer den ned, og media queries ser den bredden. Legg til `width=device-width, initial-scale=1`, og la zoom være på: `user-scalable=no` tar fra leseren den egne måten å forstørre liten tekst på.

En seksjon i full skjerm går forbi bunnen av skjermen. `100vh` er det store visningsområdet, så mens adressefeltet er ute, ligger bunnen av seksjonen bak det, sammen med enhver skjermknapp som er plassert der. Bruk `svh` til å holde

et element unna utvidede verktøylinjer i nettleseren, og `dvh` bare der det er akseptabelt at størrelsen endres midt i en rulling.

Et overlegg i full skjerm krymper til en boks inne i siden. Et overordnet element med `transform`, med `will-change` som nevner en slik egenskap, eller med bestemte innstillinger for inneslutning (*containment*) kan bli inneholdende blokk for `position: fixed`, og overlegget tar størrelsen fra det elementet. Gjengi overlegg utenfor ethvert transformert element.

En utskrevet side mister slutten av hver linje. Visningsområdeenheter og media queries som er tilpasset vinduet, lekker inn i utskriften, og en motor som løser dem opp mot vinduet, legger arket ut for en skjerm. Hold visningsområdeenheter unna alt som skal skrives ut, og sett `screen and` foran media queries som er ment for vinduer.

Les mer

- CSS Viewport Module Level 1 — `meta`-elementet for visningsområdet og de tre oppførslene til `interactive-widget`.
- CSS Values and Units Level 4: viewport-percentage lengths — det lille, store og dynamiske visningsområdet og hver enhet som bygger på dem.
- CSS Positioned Layout Level 3 — fast posisjonering, egenskapene som fanger den, og hva den gjør på papir.
- CSS Environment Variables Level 1 — de innfelte avstandene for trygge områder og regelen som gjør dem null på en rektangulær skjerm.
- Media Queries Level 4 — `width`, `height` og `orientation`, målt på visningsområdet på skjerm og på sideboksen ved utskrift.

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 et visningsområde?» 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/viewport/