Engineer

Viewport

den del av en sida som ett fönster faktiskt visar

Vad det är

En viewport, på svenska ibland kallad visningsområde, är den rektangel som en webbläsare lägger ut ett dokument i och målar upp det i: på en skärm ytan innanför fönstrets egna verktygsfält, på papper sidytan på varje ark. CSS mäter mot den hela tiden. Det initiala innehållande blocket, den ruta som rotelementet läggs ut i, får sin storlek från den, en media query på `width` eller `height` frågar efter den, och enheterna `vw` och `vh` är hundradelar av den. På en telefon behöver ordet en bestämning, och specifikationerna har flera att välja på: layout, visuell, liten, stor och dynamisk.

På en dator är viewporten och fönstret nästan samma sak. På en telefon går de isär. Eftersom så många dokument formgavs för fönster på datorer lägger mobila webbläsare ut en sida som inte ger några instruktioner i en fast bredd som efterliknar ett datorfönster, enligt specifikationen CSS Viewport vanligen 980–1 024 CSS-pixlar, och skalar sedan ned resultatet så att det får plats på glaset. En sida väljer bort det med ett enda element i sitt head, som tar emot `width`, `height`, tre gränser för skalning, `user-scalable` och `interactive-widget`:

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

Zoom delar viewporten i två. Layoutviewporten är den som CSS lägger ut sidan mot, och nypzoom lämnar den i fred, så radbrytningarna ligger kvar. Den visuella viewporten är den del av sidan som syns på glaset i stunden: den krymper när du zoomar in och flyttas när du panorerar. Som standard krymper ett skärmtangentbord bara den visuella viewporten. Med `interactive-widget=resizes-content` ber sidan webbläsaren att ändra storlek på layouten också, och med `overlays-content` att inte ändra någon av dem.

Telefoner har dessutom webbläsarfält som glider undan. Adressfältet tar plats när en sida laddas och dras in när du scrollar, så den synliga höjden ändras under en sida som inte har rört sig. CSS Values 4 ger tre storlekar för det här: den stora viewporten utgår från att varje infällbart verktygsfält är indraget, den lilla från att alla visas, och den dynamiska följer det som faktiskt gäller. Enheterna bär prefixen, `lvh`, `svh` och `dvh`, med motsvarande bredder, och vanliga `vh` och `vw` är definierade mot den stora viewporten. Ett gränssnitt som lägger sig över innehållet utan att flytta det, vanligen skärmtangentbordet, ändrar ingen av de här längderna.

Använd begreppet i ett beslut

Använd ”Vad är en viewport?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Gränssnitt som laddar snabbt, fungerar i mobilen och beter sig likadant i alla webbläsare dina kunder råkar använda.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Styrde full stack-utvecklingen av GIS-kartor — kartutveckling i full stack, från databas till webbläsare

Styrde full stack-utvecklingen av GIS-kartor Hör av er →

Varför det spelar roll

Responsiv design vilar på viewporten. En brytpunkt för bredd i en stilmall är en fråga om dess bredd, så en sida som saknar `meta`-elementet läggs ut i datorbredd på en telefon och krymps tills texten inte går att läsa, hur omsorgsfullt dess media queries än är skrivna.

Höjden är den svåra axeln. Bredden kan ändras när telefonen vrids eller fönstrets storlek ändras. Höjden kan också ändras när verktygsfältet glider, när tangentbordet öppnas och när telefonen vänds, så allt som tar skärmens höjd som mått måste säga vilken höjd det menar. Den lilla viewporten ger utrymme för innehållet när de infällbara verktygsfälten visas och lämnar en remsa tom yta när de dras in. Den stora fyller skärmen och lägger innehåll bakom verktygsfälten medan de visas. Den dynamiska tar hänsyn till ändringar i verktygsfältens tillstånd, och specifikationen varnar själv för att den kan ändra storlek på innehåll medan du scrollar, vilket flyttar allt nedanför.

Skärmar har också hörn. En telefon med rundade hörn eller ett urtag för kameran är ingen rektangel, och med standardvärdet `auto` förblir hela sidan synlig på den. Med `viewport-fit=cover` ber sidan att viewporten ska vara rektangeln runt

hela den fysiska skärmen, och tar själv på sig att hålla texten fri från urtaget med variablerna `env(safe-area-inset-*)`. De värdena måste vara noll på en rektangulär skärm, så en och samma regel för utfyllnad fungerar för båda sorterna.

Fast positionering hänger också på viewporten. En fast ruta placeras mot layoutviewporten, som CSS Positioned Layout ger samma storlek som den dynamiska viewporten, så rutor om cookies, bakgrunder till modala dialoger och menyer i helskärm står still medan sidan scrollar. Var och en utgår från att ingen förfader har tagit den rollen, och `transform`, `will-change` och `contain` kan göra just det.

Utskrift är den viewport som en stilmall skriven bara för skärm aldrig möter. På papper mäter viewportenheterna sidytan, en media query på `width` frågar efter sidrutan, och ett fast element upprepas på varje sida.

Så ser det ut i praktiken

Sidorna på den här webbplatsen deklarerar `width=device-width, initial-scale=1, viewport-fit=cover`, så de läggs ut i telefonens bredd och går ända ut till kanten under en notch. Stilmallen betalar för den andra halvan: utfyllnaden på sidorna är `max(1rem, env(safe-area-inset-left))` och dess högra tvilling, så texten håller en rem från kanten och flyttas längre in bara där ett urtag finns. På låga skärmar i liggande läge höjs golvet till `max(2rem, 7vw, env(safe-area-inset-left))`, eftersom värdena är noll på en telefon utan urtag och en rem satt för nära kanten.

Sektioner i helskärm tar sin höjd från en gemensam variabel satt till `100dvh`, med `100vh` som reserv, och kommentaren bredvid förklarar varför: `100vh` kan vara högre än det som syns medan adressfältet är framme, och det hade låtit nästa sektion titta fram under hero-ytan. Två element struntar medvetet i variabeln. Märket i sidhuvudet har statiska `vh` som mått, eftersom ett märke knutet till den dynamiska viewporten skulle ändra storlek när webbläsarens fält gled undan och bryta om hero-ytan mitt i en scroll. Den fasta bakgrunden på pekskärmar är `100lvh` hög, så den ändrar aldrig storlek när adressfältet kommer och går, och den ligger på ett eget kompositlager, eftersom webbläsare på pekskärmar rev sönder den i ljusa och mörka band vid tröghetsscroll.

I en tidigare startsideslayout hade varje tjänsteblock en minimihöjd på `100dvh`, så att det fyllde en skärm så länge innehållet var kortare än skärmen. Kommentarerna i stilmallen, granskade i november 2026, beskriver ett test vid 852 × 393 pixlar: ett blocks innehåll blev 412 pixlar högt, och blocken stötte ihop.

I den layouten kostade utfyllnad innanför `100dvh`, genom `box-sizing: border-box`, ingenting på en hög skärm och öppnade ett mellanrum först när innehållet svämmade över.

Vid det dokumenterade implementeringsstadiet för kvalitetsgrinden i augusti 2026 följde testerna media queries. Webbplatsens webbläsarstyrda kvalitetsgrind breddade sina kontroller till en gemensam tabell med tjugotvå fönsterformer, hämtade ur varje media query i stilmallen, och det tog enbart layoutkontrollen från 95 sekunder till 405.

Utskriften blottade skillnaderna mellan motorerna. Utskriftsstilmallen dokumenterar att viewportenheter räknas mot sidrutan i en motor och mot fönstret i en annan, så att en sida som skrevs ut från ett brett fönster förlorade en tredjedel av varje rad i den andra. Sidhuvudets egen stilmall dokumenterar samma glapp från media queryns sida: utskriften behöll fönstrets orientering, höjd och bredd, så ark som skrevs ut från ett brett och ett smalt fönster fick sidhuvuden som skilde sig 47 pixlar åt, tills `screen and` först i frågan begränsade regeln till fönster.

Fällan med innehållande block finns också dokumenterad. Arbetet med reduced-motion visade att fem olika egenskaper var för sig gör ett element till innehållande block för allt som är positionerat mot viewporten, och att det i tysthet hade krympt ett stängningslager till en bråkdel av skärmen.

Var det går fel

Telefonen visar en pytteliten datorsida. Utan `meta`-elementet för viewporten lägger webbläsaren ut sidan i datorbredd och skalar ned den, och sidans media queries ser den bredden. Lägg till `width=device-width, initial-scale=1`, och låt zoomen vara på: `user-scalable=no` tar ifrån läsaren det egna sättet att förstora liten text.

En sektion i helskärm sträcker sig förbi skärmens nederkant. `100vh` är den stora viewporten, så medan adressfältet är framme hamnar sektionens nederdel, och varje knapp som ligger där, bakom det. Använd `svh` för att hålla ett element fritt från utfällda verktygsfält i webbläsaren, och `dvh` bara där storleksändring mitt i en scroll är godtagbar.

Ett överlägg i helskärm krymper till en ruta inne på sidan. Ett överordnat element med `transform`, med `will-change` som nämner en sådan egenskap eller med vissa inställningar för containment kan bli innehållande block för `position: fixed`, och överlägget tar sin storlek från det elementet. Rendera överlägg utanför alla transformerade element.

En utskriven sida tappar slutet av varje rad. Viewportenheter och media queries som är skrivna för fönstret läcker in i utskriften, och en motor som räknar dem mot fönstret lägger ut arket för en skärm. Håll viewportenheter borta från allt som skrivs ut, och sätt `screen and` först i frågor som är tänkta för fönster.

Läs mer

- CSS Viewport Module Level 1 — `meta`-elementet för viewporten och de tre beteendena för `interactive-widget`.
- CSS Values and Units Level 4: viewport-percentage lengths — de små, stora och dynamiska viewportstorlekarna och varje enhet som bygger på dem.
- CSS Positioned Layout Level 3 — fast positionering, egenskaperna som fångar den och vad den gör på papper.
- CSS Environment Variables Level 1 — safe-area-värdena och regeln som gör dem till noll på en rektangulär skärm.
- Media Queries Level 4 — `width`, `height` och `orientation`, mätta på viewporten på skärm och på sidrutan vid utskrift.

Ordlista

Välj utifrån era faktiska förutsättningar

Beskriv arbetsflödet, befintliga gränssnitt och användarna som behöver förändringen. Vi kan definiera en testbar leverans och en överlämning som gör att teamet kan underhålla lösningen.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är en viewport?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

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

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/viewport/