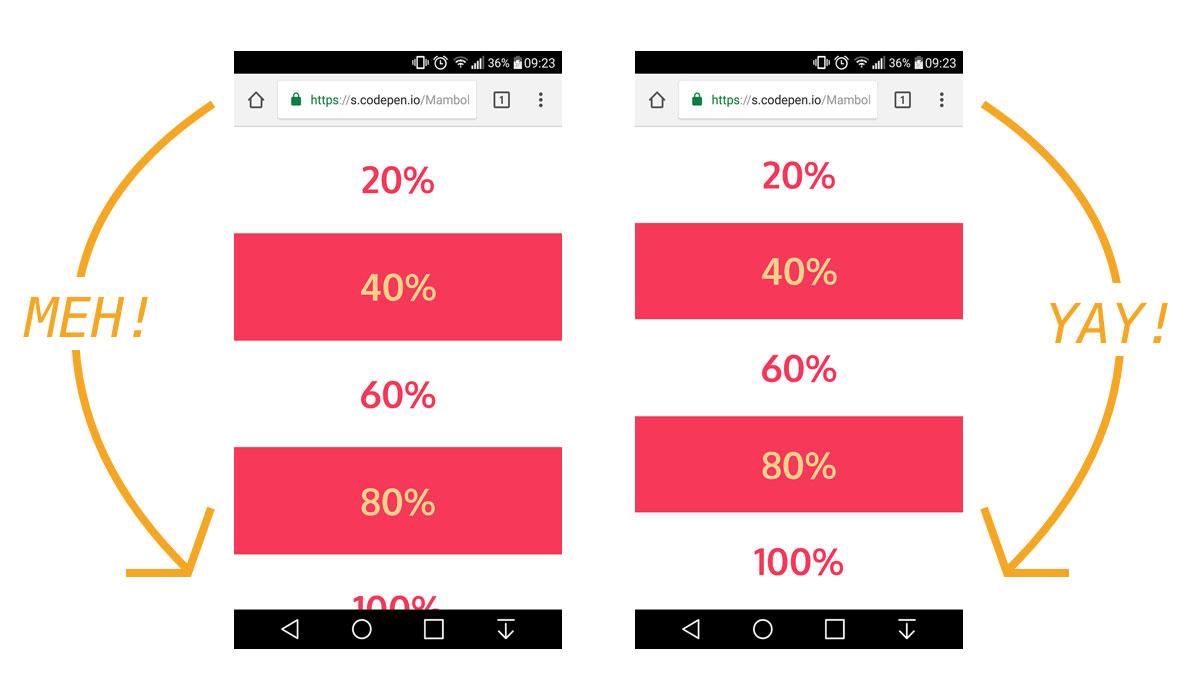
**[ 뷰포트 추가단위 : dvh, svh, lvh ]**

뷰포트 단위로 높이 값을 주고 싶을 때 vh 단위를 사용한다.

[2020.04.22 - [Frontend/CSS] - [css 단위] vw, vh, vmin, vmax (Viewport 단위)](https://abcdqbbq.tistory.com/51)

그치만 모바일 주소 표시줄 영역에 의해서 100vh로 높이 값을 설정했다고 하더라도

화면이 짤리는 경우가 발생할 수 있다.

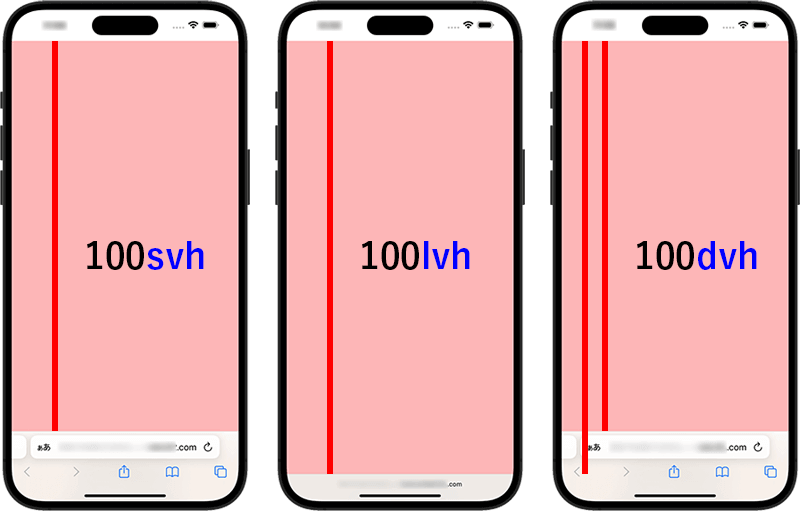
이미지 출처 : https://css-tricks.com/the-trick-to-viewport-units-on-mobile/

그럴 때 해결할 수 있는 방법 중 하나는

div {height:100vh; height:-webkit-fill-available;}

위와 같이 -webkit-fill-available 값을 height값에 적용하는 방법이 있는데

이 방법은 ios에서만 적용이 되고 안드로이드에서는 적용이 안되는 문제가 있기 때문에 권장하진 않는다.

이미지 출처 : https://blanche-toile.com/web/large-small-and-dynamic-viewport-units

### 1. dvh (Dynamic Viewport Height)

dvh는 주소 표시줄이 스크롤을 통해 축소가 되건 노출이 되고 있건 상관 없이

현재 보여지는 뷰포트 높이를 **동적**으로 가져온다.

위 이미지의 예시 처럼 100dvh는 주소표시줄의 **유무에 따라 값이 달라진다.**

div {height:100dvh;}

### 2. svh (Short Viewport Height)

svh는 사용자 화면 기준으로 가장 짧은 뷰포트 값을 가져온다.

주소 표시줄이 없어져도 기존 주소표시줄의 높이 값을 뺀 나머지 값을 가져온다.

div {height:100svh;}

### 3. lvh (Large Viewport Height)

lvh는 svh와는 반대로 사용자 화면 기준으로 가장 긴 뷰표트 값을 가져온다.

주소 표시줄이 있더라도 주소 표시줄이 없을 때의 총 화면의 길이를 가져온다.

div {height:100lvh;}

위 3가지의 단위를 볼 때

주소 표시줄의 유무에 따라 뷰포트 높이가 가변적으로 변할 때를 고려해서

dvh를 사용하는 것이 가장 무난하고 우리가 원하는 화면을 보여주는데에 좋은 선택이 될 것이다.

\*dvh, svh, lvh는 브라우저(pc, mobile) 하위 버전에서는 제공이 되지 않기 때문에 작업 환경을 고려한 적용이 필요하다.

**[ 뷰포트 단위의 문제점과 해결책 ]**

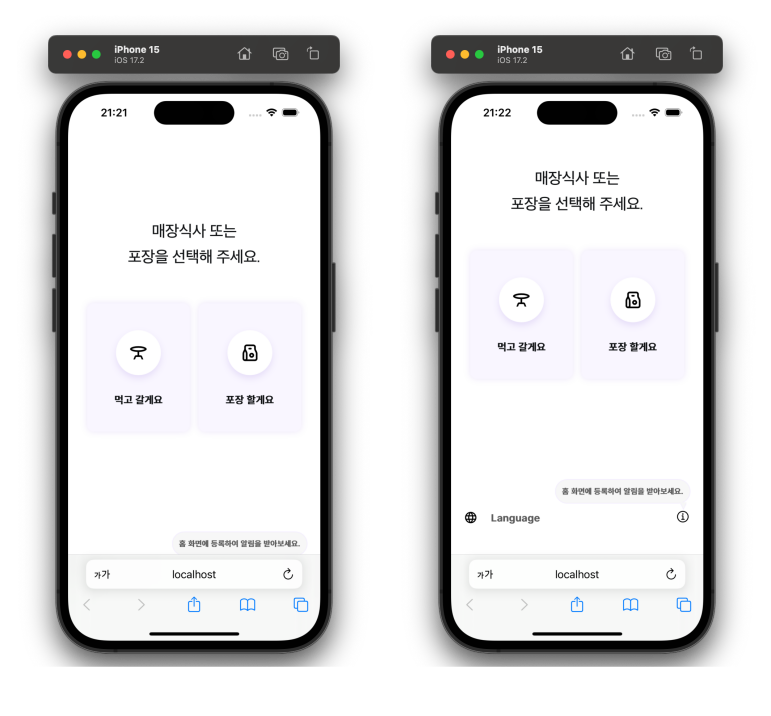
vh (viewport height)와 vw(viewport width)는 기존부터 존재하던 단위다. 둘 다 [뷰포트](https://developer.mozilla.org/ko/docs/Glossary/Viewport)를 기준으로 측정되는 단위이다.

이는 데스크탑 브라우저에서는 무리 없이 잘 동작한다. 그러나 문제는 모바일에서 발생했다.

예를 들어, 레이아웃의 높이 값을 100vh로 지정하는 경우를 살펴보자.

## **문제점**

높이를 100vh로 지정을 했기에 나는 화면이 브라우저에 꽉차게 나오기를 기대했다. 그러나 ios, safari 기준으로 봤을 때 다음과 같이 화면이 나왔다.



의도치 않은 스크롤이 생긴것을 확인할 수 있다.

그 이유는 일부 모바일 브라우저에서는 사용자와의 인터랙션에 따라 브라우저의 UI가 변할 수 있기 때문이다.

예를 들어, safari에서의 100vh는 주소창이 없을 때를 기준으로 삼기 때문에 다음과 같은 스크롤이 생긴것이다.

## **새로운 단위**

위와 같은 문제를 해결하기 위해 새로운 단위들이 도입되었다.

***도입되기 전까지 해결책과 문제점들..***<https://dev.to/frehner/css-vh-dvh-lvh-svh-and-vw-units-27k4#the-new-css-units>

### **svh, svw (small viewport)**

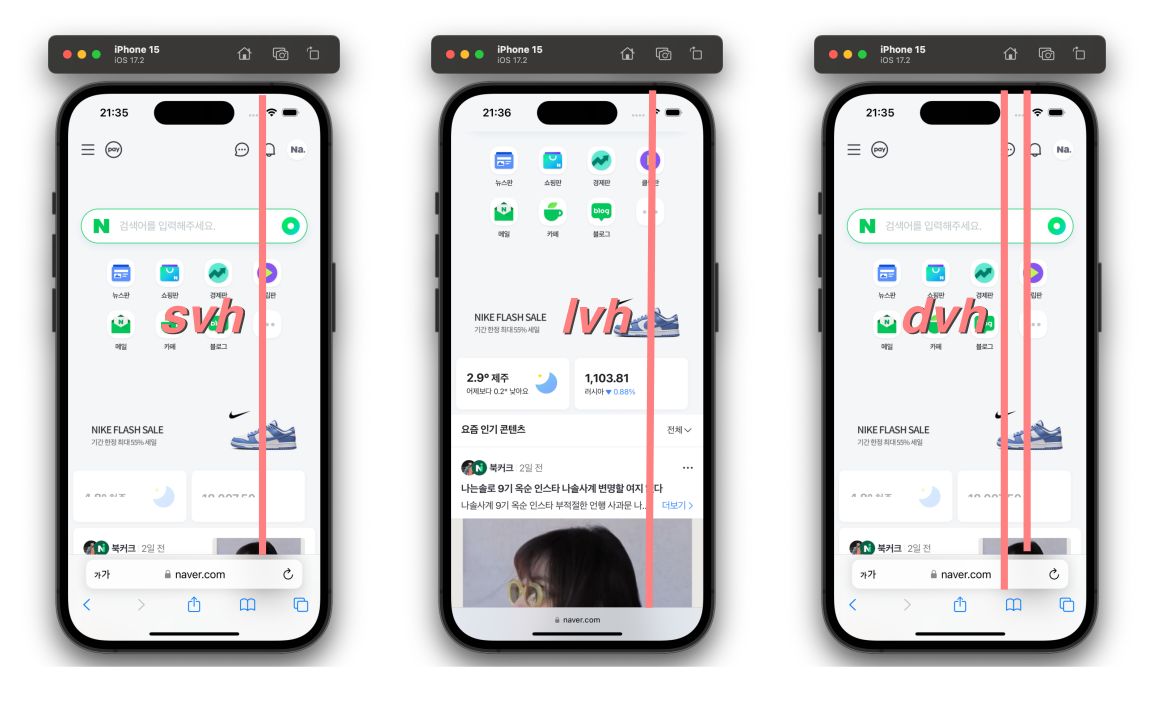
* 브라우저의 UI가 가장 크고, 컨텐츠가 가장 작을때를 가정한다.
* 예를 들어, 모바일 브라우저에서 주소창이 보이는 경우를 기준으로 한다.
* 정적인 값으로 브라우저의 UI가 변한다고 값이 늘어나거나 줄지 않는다.

### **lvh, lvw (large viewport)**

* 브라우저의 UI가 가장 작고, 컨텐츠가 가장 클때를 가정한다.
* 예를 들어, 모바일 브라우저에서 주소창이 없는 경우를 기준으로 한다.
* 정적인 값으로 브라우저의 UI가 변한다고 값이 늘어나거나 줄지 않는다.

### **dvh, dvw (dynamic viewport)**

* 브라우저 UI 크기에 따라 동적으로 변한다.
* 그렇기에 사용자가 스크롤등을 할 경우 페이지의 크기가 갑자기 변해 안좋은 사용자 경험(layout shifting)을 주거나, 성능에 영향을 미칠 수 있으므로 주의해서 사용해야 한다.



## **그럼 기존 vh, vw는..??**

동작하는 것을 보면 lvh와 vh가 비슷하게 동작하는 것처럼 보인다.  
하지만 정의 자체는 다르므로 아래를 링크를 한번 읽어보면 좋을 것 같다.

CSS Values and Units Module Level 4  
<https://www.w3.org/TR/css-values-4/#viewport-relative-lengths>

vh는 뷰포트와 관련이 있고, lvh는 뷰포트 + 브라우저 UI와 관련이 있다. 따라서 lvh를 사용하면 같은 기기에서도 브라우저마다 값이 다르게 적용될 수 있지만, vh를 사용하면 일괄되게 적용될 수 있다.

이러한 차이점들을 알고 잘 활용하면 좋을 듯 하다.