**Що таке адаптивна верстка?**

Адаптивний веб-дизайн (RWD - Responsive web design) створює систему, що дозволяє одному сайту (з одним URL-адресою та одним джерелом контенту) реагувати і адаптуватися до розмірів пристрою користувача. Адаптивний веб-сайт створений з використанням верстки з гнучким макетом, який підлаштовується під розміри екрану пристрою.

По-суті, завдяки адаптивної верстці, ваш веб-сайт буде відмінно виглядати і добре працювати як на настільному комп'ютері (або ноутбуці), так і на планшеті, і в браузері мобільного телефону.

У минулому розробники створювали більше одного сайту, для відповідності сторінок екранів різних розмірів. З урахуванням того, що на ринку сьогодні представлено багато типів пристроїв, це здається абсолютно неефективним ... вірно?

Сам термін «адаптивна верстка» був фактично придуманий в 2010 році веб-дизайнером Ітаном Маркотт. Сьогодні адаптивна верстка в веб-дизайні вже не нова тенденція, а швидше за перевірений часом спосіб мислення, що стоїть за створенням сайтів.

На сьогодні наявність адаптивного веб-сайту більше не є ще однією можливістю для розвитку вашої інфраструктури - це вже необхідність!

**Переваги мобільного адаптивного дизайну**

* Перевага адаптивного макета номер один - це отримання гарантії того, що будь-який користувач на будь-якому пристрої матиме найкращі можливості взаємодії з контентом на вашому сайті.
* Адаптивна верстка веб-сайту також є відмінним способом поліпшити контент на вашому сайті. Ви зможете переконатися, що ті, хто використовує мобільний пристрій, бачать всю важливу інформацію.
* Завдяки особливостям алгоритму Google, адаптивний веб-дизайн підвищує видимість сайту в пошукових системах, оскільки він зручний для перегляду на мобільних пристроях.
* Сайт з якісним поданням контенту на мобільному пристрої буде перебувати в результатах пошуку вище, ніж сайт, добре відображає контент тільки на десктопах.

**Чому адаптивний дизайн важливий для бізнесу**

* Розширюється охоплення клієнтів завдяки захопленню користувачів невеликих пристроїв (планшетів і смартфонів);
* Постійний досвід роботи з широкою аудиторією, який може збільшити кількість потенційних клієнтів, продажу та конверсії;
* Аналітика, відстеження та звітність за версіями сайтів для десктопів і мобільних пристроїв можуть бути в одному місці;
* Витрати часу і вартість управління контентом знижується;
* Більше 60% запитів в Google на кінець першого кварталу 2019 робляться з мобільних пристроїв.

Є два способи, за допомогою яких можна забезпечити взаємодію користувача з сайтом через мобільні додатки.

Перший називається динамічним показом (**Dynamic Serving**), в якому використовується один і той же URL-адресу, але різні коди HTML і CSS. Сторінки розпізнають пристрій, на якому вони проглядаються, і надають відповідний код.

Другий спосіб - це взагалі окремий **мобільний сайт**. Коли користувачі відвідують сайт з мобільного пристрою, вони відправляються на інший - мобільний URL-адресу.

Врахуйте, що на Google щодня доводиться більше 5,6 мільярда пошукових запитів.

Рекомендована Google конфігурація для сайтів, оптимізованих для смартфонів, - це сайти з адаптивним веб-дизайном.

Існує безліч різних пристроїв з екранами різних розмірів: телефони, фаблет, планшетні і домашні ПК, ігрові консолі, телевізори і навіть електронні аксесуари, які можна носити прямо на собі. Розміри екранів постійно змінюються, тому важливо, щоб сайт міг адаптуватися до будь-якого з них - не тільки зараз, але і в майбутньому.

Про адаптивний веб-дизайні вперше написав **Ітан Маркотт** в статті журналу **A List Apart**. Це саме те, чого так не вистачало пристроїв і їх користувачам. Макет підлаштовується під пристрій, виходячи з його можливостей і розміру. Наприклад, певний контент на телефоні розташовується в одній колонці, а на планшеті - в двох.

**Налаштування області перегляду**

Сторінки, адаптовані для перегляду на різних пристроях, повинні містити в контейнері head-тег viewport. Він повідомляє браузеру, яким чином потрібно контролювати розміри і масштаб сторінки.

* Щоб контролювати масштабування вікна перегляду в браузері, використовуйте метатег **viewport**.
* Додайте **width = device-width**, щоб адаптувати ширину вікна перегляду до екрану пристрою.
* Вставте **initial-scale = 1**, щоб забезпечити співвідношення **1:1** між пікселями CSS і незалежними пікселями пристрою.
* Щоб сторінка була доступна, перевірте, чи не відключено чи призначене для користувача масштабування.

**Синтаксис**

<meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

Намагаючись поліпшити роботу з сайтами, мобільні браузери відображають сторінку тієї ж ширини, що і на екрані ПК (зазвичай вона складає близько 980 пікселів, але на різних пристроях може відрізнятися). Після цього браузер збільшує розмір шрифтів і змінює розмір контенту, щоб вмістити сторінку на екрані. В результаті користувачі бачать різні шрифти, і для роботи з сайтом їм доводиться збільшувати його масштаб.

Щоб сторінка підлаштувалася під ширину потрібного екрану (в незалежних пікселях), використовуйте метатег viewport **width = device-width**. З його допомогою розмір і положення контенту зміняться, і сайт буде добре виглядати на будь-якому пристрої.

Деякі браузери не редагують ширину сторінки і розмір контенту, а змінюють орієнтацію сайту на альбомну або збільшують масштаб. За допомогою атрибута **initial-scale = 1** можна вказати браузеру співвідношення пікселів CSS і пристрої, що дорівнює 1:1 незалежно від орієнтації дисплея. Завдяки цьому сторінка буде краще виглядати в альбомної орієнтації.

**Спеціальні можливості області перегляду**

Налаштувавши initial-scale, ви також можете встановити наступні атрибути для області перегляду:

* minimum-scale
* maximum-scale
* user-scalable

Після їх налаштування встановлюється обмеження на збільшення і зменшення області перегляду, якщо через це можуть стати недоступними більш доступного режиму.

**Масштабування контенту для області перегляду**

Для перегляду сайтів на звичайних комп'ютерах і мобільних пристроях використовується вертикальна прокрутка. Якщо користувачеві доводиться прокручувати сайт по горизонталі або зменшувати масштаб, щоб побачити сторінку повністю, він відчуває великі незручності.

* Не використовуйте великі елементи з фіксованою шириною.
* Для коректного відображення контенту не обмежуйте його певної шириною області перегляду.
* Використовуйте медіазапити CSS, щоб вказати різні стилі для великих і маленьких екранів.

Використовуючи метатег **viewport** при розробці сайту для мобільних пристроїв, можна помилково створити контент сторінки, які не повністю відповідає певній галузі перегляду. Наприклад, якщо ширина показуваного зображення більше області перегляду, доведеться використовувати горизонтальну прокрутку. Щоб позбавити користувачів від цієї необхідності, буде потрібно підігнати контент під шир ну області перегляду.

Діапазон розмірів екрану і ширини в пікселях CSS дуже великий (наприклад, телефони можуть значно відрізнятися від планшетів або навіть від інших телефонів), тому відображення контенту не повинно залежати від конкретної ширини області перегляду.

Установка великих абсолютних значень ширини CSS для компонентів сторінки зробить div занадто великим для більш вузької області перегляду (наприклад, на пристроях шириною 320 пікселів CSS, таких як iPhone). Замість цього можна використовувати значення в відносних одиницях, наприклад width: 100%. Також намагайтеся уникати великих абсолютних значень позиціонування. Через них на маленькому екрані елемент може виявитися за межами області перегляду.

**Поліпшення чуйності за допомогою медіазапитів CSS**

Медіазапити - це прості фільтри, які можна застосовувати до стилів CSS. Вони дозволяють змінювати стилі на підставі характеристик пристрою, пов'язаних з відображенням контенту, включаючи тип, ширину, висоту, орієнтацію і навіть дозвіл екрана.

* Медіазапити також дозволяють вибрати стиль на основі характеристик пристрою.
* Додайте min-width замість min-device-width для коректного відображення сайту на більшості пристроїв.
* Щоб не порушувати структуру макета, використовуйте елементи відносних розмірів.
* Наприклад, ви можете помістити в медіазапит print всі стилі, необхідні для друку:

**Синтаксис**

<link *rel*="stylesheet" *href*="print.css" *media*="print">

Крім використання атрибут media в посиланні на таблицю стилів існує ще два способи застосувати медіазапроси **@media** та **@import**, які можна вбудувати в файл CSS: Пріоритет віддається першим двом методам, більш ефективним, ніж синтаксис **@import.**

*@media* print {  
 */\* print style sheets go here \*/*}  
  
*@import* url(print.css) print;

Логіка медіазапросов не є взаємовиключної, тому до блоку CSS застосовуються всі фільтри, що відповідають його критеріям. Порядок застосування фільтрів обумовлений стандартними правилами CSS.

**Застосування медіазапитів на основі розміру області перегляду**

За допомогою медіазапитів можна створити чуйну середу, в якій до кожного розміром екрану будуть застосовуватися відповідні стилі. Синтаксис медіазапити допускає створення правил, які застосовуються на підставі характеристик пристрою.

## Розглянемо синтаксис написання медіа запитів

Типи пристроїв, для яких зазначено це правило:

* screen - тип пристрою, у якому застосовуватиметься це правило. У разі це дисплей (экран);
* print – принтер або режим попереднього перегляду сторінки перед друком;
* all - всі типи пристроїв;
* speech – гаджети, які читають контент з екрану (скринрідери).

Умови медіа запитів поділяються основними логічними операторами:

* and - потрібно виконання всіх зазначених умов;
* кома, - потрібно виконання однієї із зазначених умов;
* not – заперечення зазначеної умови. Тобто. якщо вказана умова не виконується.
* У круглих дужках зазначаються медіа функції. Це найпростіші функції, які сприймаються як правило

*@media* (query) {  
 */\* CSS Rules used when query matches \*/*}

В адаптивному веб-дизайні найбільш часто використовуються функції **min-width**, **max-width**, **min-height** і **max-height** (хоча можливі й інші запити).

|  |  |
| --- | --- |
| Атрибут | Результат |
| min-width | Правило застосовується до браузеру, ширина якого більше значення, зазначеного в запиті. |
| max-width | Правило застосовується до браузеру, ширина якого менше значення, зазначеного в запиті. |
| min-height | Правило застосовується до браузеру, висота якого більше значення, зазначеного в запиті. |
| max-height | Правило застосовується до браузеру, висота якого менше значення, зазначеного в запиті. |
| orientation = portrait | Правило застосовується до браузеру, висота якого не менше його ширини. |
| orientation = landscape | Правило застосовується до браузеру, ширина якого більше висоти. |

**Синтаксис**

<link *rel*="stylesheet" *media*="(max-width: 640px)" *href*="max-640px.css">  
<link *rel*="stylesheet" *media*="(min-width: 640px)" *href*="min-640px.css">  
<link *rel*="stylesheet" *media*="(orientation: portrait)" *href*="portrait.css">  
<link *rel*="stylesheet" *media*="(orientation: landscape)" *href*="landscape.css">  
<style>  
 *@media* (min-width: 500px) and (max-width: 600px) {  
 h1 {  
 color: fuchsia;  
 }  
  
 .*desc*:*after* {  
 content:" In fact, it's between 500px and 600px wide.";  
 }  
 }  
</style>

* При ширині браузера від 0 пікс. до 640 пікс. застосовується max-640px.css.
* При ширині браузера від 500 пікс. до 600 пікс. застосовуються стилі з @media.
* При ширині браузера від 640 пікс. і вище застосовується min-640px.css.
* Якщо в браузері ширина більше висоти, застосовується landscape.css.
* Якщо в браузері висота більше ширини, застосовується portrait.css.

**Примітка до min-device-width**

Також можливе створення запитів на підставі **\*-device-width**, хоча робити це наполеглево не рекомендується.

Різниця невелика, але дуже важлива: **min-width** виходить з розміру вікна браузера, а **min-device-width** - з розміру екрану пристрою. На жаль, деякі браузери (включаючи застарілу версію браузера для Android) не завжди правильно визначають ширину області перегляду і замість неї можуть повідомити розмір екрану в пікселях пристрою.

До того ж, використання **\*-device-width** може перешкодити контенту підлаштуватися під екран звичайного комп'ютера або іншого пристрою, на якому можна змінити розмір вікна. Причина в тому, що цей запит заснований на розмірі конкретного пристрою, а не вікна браузера.

**Використання відносних одиниць**

Основний принцип адаптивного веб-дизайну (і головна відмінність від макетів з фіксованою шириною) - рухливість і пропорційність. Використання відносних розмірів допомагає спростити макет і запобігти випадковому створення компонентів, які не вміщаються в область перегляду.

Наприклад, установка параметра width рівним 100% для блоку div верхнього рівня призведе до того, що він буде заповнювати всю ширину області перегляду і ніколи не буде занадто малий або великий для неї. Блок div в будь-якому випадку буде відповідати екрану, будь то iPhone (320 пікс.), Blackberry Z10 (342 пікс.) Або Nexus 5 (360 пікс.).

Крім того, використання відносних одиниць дозволяє браузерам відображати контент, виходячи з призначеного для користувача масштабу. Це означає, що горизонтальна панель прокрутки на сторінці не знадобиться.

**Not recommended — fixed width**

div.*fullWidth* {  
 width: 320px;  
 margin-left: auto;  
 margin-right: auto;  
}

**Recommended — responsive width**

div.*fullWidth* {  
 width: 100%;  
}

**Вибір контрольних точок**

Будьте обережні при визначенні контрольних точок на основі класів пристроїв. Якщо при установці цих точок ви будете орієнтуватися на всі доступні пристрої, бренди або операційні системи, технічне обслуговування сайту стане практично неможливим. Краще, якщо контент буде сам визначати, як макет повинен підлаштовуватися під контейнер.

* Створюйте контрольні точки на основі контенту, а не для окремих пристроїв, продуктів або брендів.
* Спочатку розробіть дизайн для самого маленького мобільного пристрою, а потім переходите до версій для великих екранів.
* Обмежте довжину рядків 70-80 символами.

Прийоми адаптивному веб-дизайну швидко еволюціонують, але є багато перевірених варіантів, які добре працюють при використанні як настільних комп'ютерів, так і мобільних пристроїв

Більшість макетів, які використовуються для створення адаптивних веб-сторінок, можна віднести до однієї з п'яти категорій шаблонів:

* Mostly Fluid (Найбільш гумовий)
* Column Drop (Стовпці один під одним)
* Layout Shifter (Рухається макет)
* Tiny Tweaks (Крихітні зміни)
* Off Canvas (Поза екрану)

У деяких випадках на сторінці може використовуватися поєднання шаблонів, наприклад Column Drop і Off Canvas. Ці шаблони, які спочатку були визначені Люком Вроблевського (Luke Wroblewski), є надійною відправною точкою для створення будь-якої адаптивної сторінки.

Додаткові джерела:

<https://www.w3schools.com/css/css_rwd_mediaqueries.asp>

<https://internetingishard.com/html-and-css/responsive-design/>

[https://freehost.com.ua/](https://freehost.com.ua/ukr/faq/articles/chto-takoe-media-zaprosi-css-i-dlja-chego-oni-nuzhni/?gclid=CjwKCAjwm8WZBhBUEiwA178UnPQBsoA2iwYhG_D7cd3yrxopndg19OeliQwkMV1db_H62G6-3EMHtBoCEvIQAvD_BwE)