JAVARUSH

Путь к карьере Frontend Fullstack разработчика

Модуль 1. WEB CORE

Уровень 13. Отзывчивая верстка.
Отзывчивая графика. Часть 1.

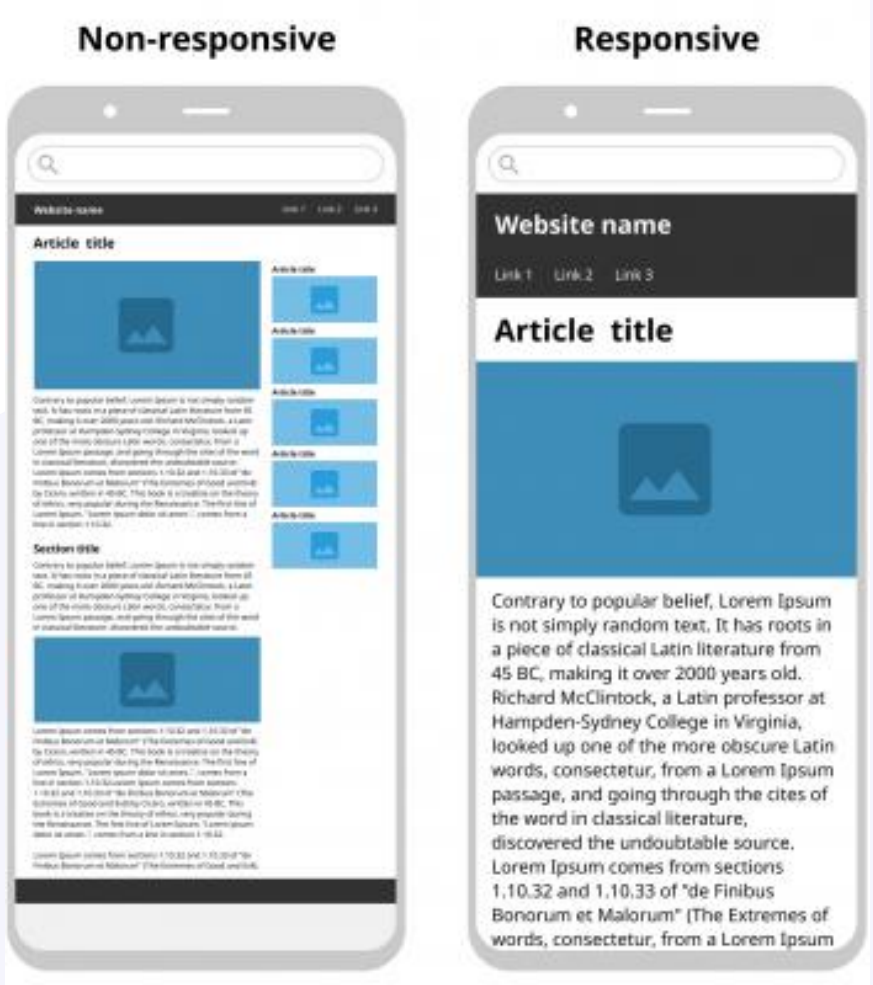
JAVARUSH

Responsive design

Это подход к созданию сайтов, которые автоматически подстраиваются под разные размеры экрана, будь то огромный монитор или маленький смартфон. Это важно, потому что сегодня люди используют самые разные устройства для просмотра веб-страниц.

Как это было раньше?

В начале 2000-х сайты создавались в основном для больших экранов компьютеров. Если открыть такой сайт на телефоне, всё будет выглядеть ужасно: мелкий текст, огромные картинки, неудобная навигация.



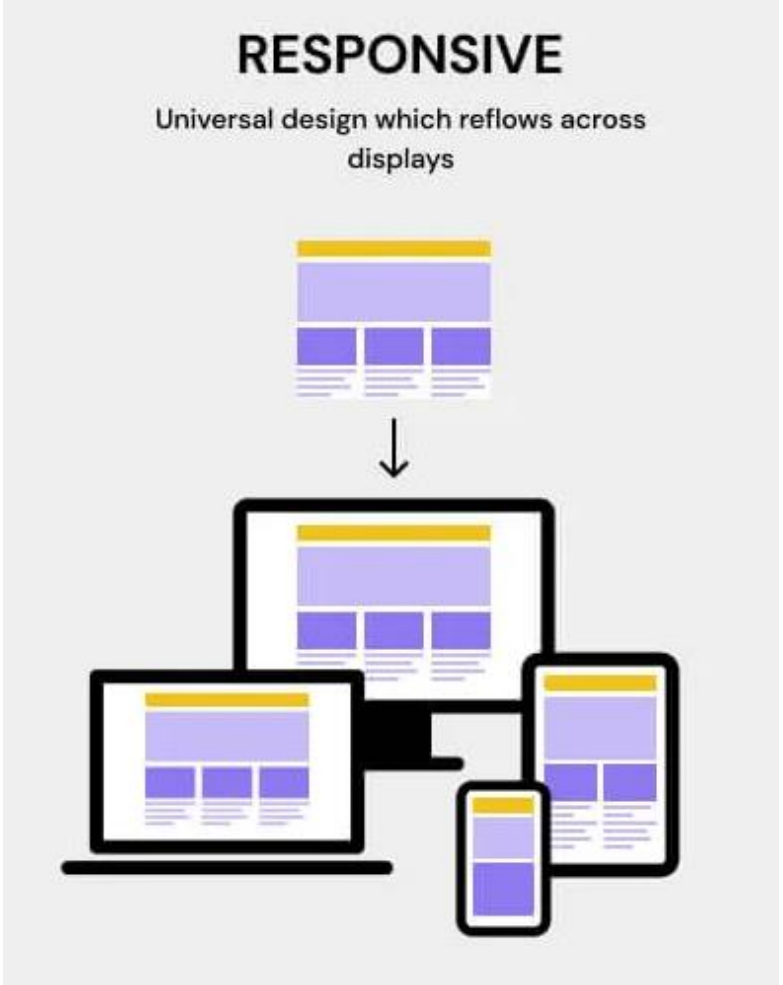
Что изменилось?

С появлением смартфонов возникла необходимость делать сайты удобными для маленьких экранов. Текст должен быть читаемым, картинки - подходящего размера, а интерфейс - интуитивно понятным без необходимости увеличивать масштаб.

Как это реализуется?

Магия адаптивного дизайна происходит благодаря CSS. С помощью CSS (media queries) мы можем задавать разные стили для разных размеров экрана. Например, на большом экране мы можем показывать три колонки контента, а на маленьком - перестроить их в одну колонку для удобства чтения.

В принципе, можно разбить устройства на разные категории и верстать для каждой из них отдельно, но это займет слишком много времени, да и кто знает, какие стандарты будут через пару лет?



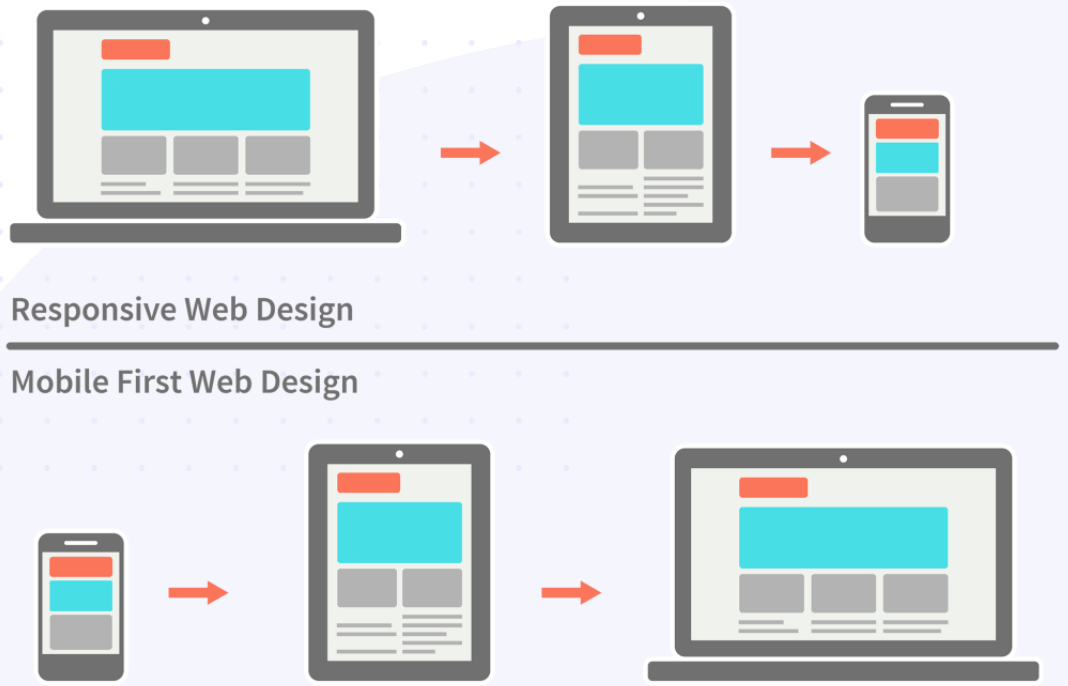
Mobile-First

Это когда сначала всё делают для смартфона, а потом для всего остального.

Мобильный трафик превышает десктопный: Сегодня большинство пользователей заходят на сайты со смартфонов. Игнорировать это - значит терять большую часть аудитории. Google и другие поисковые системы отдают предпочтение мобильно-дружественным сайтам в результатах поиска.

Как это работает?

1. Рисуется дизайн, чтобы на смартфоне всё выглядело отлично. На основе этого дизайна пишется код. Или код и дизайн делаются параллельно на основе общего прототипа.
2. Если нужно — делают отдельную версию для компьютера, если нет — доделывают мобильный дизайн, чтобы он «расширялся» до большого экрана.
3. Если нужно что-то улучшить в программе или сервисе, то сначала делают это для смартфонов, а потом для компьютеров. Иногда версия для смартфона работает лучше, чем настольная или в вебе.



JAVARUSH

Нюансы проектирования под мобильные устройства

1. **Удобство взаимодействия:**
    - **Учитывайте размер пальцев:** Элементы управления (кнопки, ссылки) должны быть достаточно большими для комфортного нажатия пальцем.
    - **Минимизируйте ввод с клавиатуры:** Используйте автозаполнение, выпадающие списки и другие элементы, чтобы уменьшить необходимость ввода текста.
2. **Макет и отображение:**
    - **Вертикальная ориентация:** Контент должен быть организован в одну колонку с вертикальной прокруткой.
    - **Ограниченное пространство:** Используйте эффективно каждый пиксель экрана, избегайте лишних элементов и информации.
    - **Горизонтальный контент:** Будьте осторожны с горизонтальной прокруткой, она может быть неудобной на мобильных устройствах.
3. **Функциональность:**
    - **Мобильные возможности:** Используйте возможности мобильных устройств, такие как геолокация, камера, сканер отпечатков пальцев, для улучшения пользовательского опыта.
    - **Уведомления:** Будьте осторожны с уведомлениями, чтобы не раздражать пользователей.
4. **Другие особенности:**
    - **Скорость загрузки:** Мобильные пользователи часто имеют медленное интернет-соединение, поэтому оптимизируйте ваш сайт для быстрой загрузки.
    - **Жесты:** Учитывайте распространенные жесты на мобильных устройствах (тапы, свайпы, щипки) при проектировании интерфейса.

Desktop-first

Большие экраны предоставляют больше пространства для размещения элементов интерфейса. Это позволяет создавать сложные и функциональные дизайны, которые не всегда возможно реализовать на мобильных устройствах. Например, на десктопе можно использовать таблицы с множеством столбцов, подробные формы ввода, расширенные панели инструментов и другие элементы, требующие много места.

Десктопные сценарии использования

При разработке для десктопа важно учитывать особенности взаимодействия пользователя с клавиатурой и мышью. Например:

- **Точность курсора:** Кнопки и ссылки могут быть меньше, так как курсор мыши позволяет более точно наводить на элементы.
- **Постоянная видимость курсора:** Учитывайте, что при движении мыши контент не должен неожиданно появляться или исчезать, чтобы не сбивать пользователя с толку.
- **Поддержка клавиатуры:** Обеспечьте возможность управления интерфейсом с помощью клавиатуры, включая горячие клавиши и навигацию с помощью Tab.
- **Копирование и вставка:** Учитывайте, что пользователи могут копировать и вставлять большие объемы текста, поэтому предусмотрите соответствующие возможности в интерфейсе.

Fluid Layouts

Это подход, при котором элементы на странице масштабируются пропорционально размеру экрана. Это позволяет сайту выглядеть гармонично на любых устройствах, от больших мониторов до маленьких смартфонов.

Вместо фиксированных значений в пикселях, fluid design использует относительные единицы измерения, такие как проценты. Например, если элементу задана ширина 50%, он всегда будет занимать половину ширины своего родительского контейнера, независимо от размера экрана.

Преимущества гибкости макета:

- Улучшение отображения контента на различных устройствах
- Лучшая адаптация к изменениям размеров экрана
- Повышение удобства использования и доступности контента

Пример - https://lh7-us.googleusercontent.com/RCILGWNSRZwqI2S8eVDiAmWq4e70GLaFFa0wtrajN237myRxF5KtOR80BxKRRgsTbbyhu84HT0LgB8BODzCc0Bv0kvB0o98-h6BQkUIFreZ_aq3zLmbWFb4rg8HjMuapNX72_C0dW31XJQxPeMI0FQI

JAVARUSH

Пример Fluid Layouts

Fluid design не является полноценной заменой адаптивного дизайна, так как он не позволяет кардинально менять макет страницы для разных устройств. Используйте fluid design в сочетании с медиа-запросами, чтобы создавать полностью адаптивные сайты, которые отлично выглядят и работают на любых экранах.

```
<div class="container">
   <div class="wide">Широкий элемент (50%)</div>
   <div class="narrow">Узкий элемент (20%)</div>
   <div class="narrow">Узкий элемент (20%)</div>
</div>
```

```
.container {
   width: 100%;
}

.wide {
   width: 50%;
}

.narrow {
   width: 20%;
}
```

JAVARUSH

Основы медиа-запросов

При вёрстке адаптивных сайтов часто нужно, чтобы какой-то набор стилей применился только при соблюдении определённых условий.

@media позволяет применять определенные стили только при выполнении заданных условий. Условия могут включать размеры экрана, разрешение, ориентацию и другие параметры.

```
@media media-type and (media-feature) {
   /* CSS-правила, применяемые при выполнении условий */
}
```

- media-type: тип медиа, для которого будут применяться стили. Чаще всего используется screen
- media-feature: особенности медиа, такие как ширина, высота, ориентация и т. д.

JAVARUSH

Типы устройств

Есть три типа устройств, которые мы можем указать:

- all — медиавыражение применится ко всем устройствам. Если не задать никакой тип, по умолчанию применится этот;
- print — стили внутри такого медиавыражения применятся при печати на принтерах или экспорте в PDF, в том числе в режиме предпросмотра документа;
- screen — для устройств с экранами.

В большинстве случаев, когда вы пишете стили только для экрана, указывать типы screen или all не нужно. Реальное практическое использование есть только у типа print.

JAVARUSH

Характеристики устройства

Сегменты экрана:

- horizontal-viewport-segments и vertical-viewport-segments позволяют определить, на сколько частей разделён экран устройства. Это полезно для адаптации стилей под складные устройства или устройства с несколькими экранами.

Переполнение контента:

- overflow-block и overflow-inline проверяют, как устройство обрабатывает контент, который не помещается на экране по блочной или строчной оси соответственно. Это позволяет настроить стили для случаев, когда контент обрезается, прокручивается или разбивается на страницы.

Тип экрана:

- grid определяет, является ли экран растровым (современные экраны) или сеточным (старые устройства). Это может быть полезно для настройки стилей под специфические устройства.

Разрешение экрана:

- resolution, min-resolution и max-resolution проверяют разрешение устройства в точках на дюйм (dpi) или точках на сантиметр (dpcm). Это позволяет адаптировать стили под экраны с разной плотностью пикселей.

```
/* Стили для устройств с одним сегментом
экрана */
@media (horizontal-viewport-segments: 1) {
    /* ... */
}

/* Стили для устройств, где контент по
блочной оси можно прокручивать */
@media (overflow-block: scroll) {
    /* ... */
}

/* Стили для устройств с сеточным экраном */
@media (grid: 1) {
    /* ... */
}

/* Стили для устройств с разрешением не
более 300 dpi */
@media (max-resolution: 300dpi) {
    /* ... */
}
```

JAVARUSH

Характеристики страницы и окна браузера

Медиа-запросы позволяют адаптировать стили не только под характеристики устройства, но и под размеры самого окна браузера.

Проверка ширины и высоты:

- width, min-width, max-width: Проверяют ширину окна браузера, минимальную или максимальную ширину.
- height, min-height, max-height: Проверяют высоту окна браузера, минимальную или максимальную высоту.
- Значения: положительное число с любыми единицами измерения (px, em, rem, %).

```
/* Стили для экранов с шириной не более 1280 пикселей */
@media (max-width: 1280px) {
    /* ... */
}

/* Стили для экранов с минимальной высотой 30rem */
@media (min-height: 30rem) {
    /* ... */
}
```

JAVARUSH

Характеристики страницы и окна браузера

Ориентация экрана:

- orientation: Проверяет ориентацию окна браузера (альбомная или портретная).
- Значения: landscape (альбомная), portrait (портретная).

```
/* Стили для альбомной ориентации */
@media (orientation: landscape) {
    /* ... */
}
```

Соотношение сторон:

- aspect-ratio, min-aspect-ratio, max-aspect-ratio: Проверяют соотношение ширины и высоты окна браузера.
- Значения: два целых положительных числа, разделенных слэшем (например, 16/9).

```
/* Стили для экранов с соотношением сторон 16:9 */
@media (aspect-ratio: 16/9) {
    /* ... */
}
```

JAVARUSH

Характеристики качества отображения

Медиа-запросы позволяют адаптировать стили не только под размеры и тип устройства, но и под особенности отображения контента на экране. Это дает вам возможность создавать более гибкие и оптимизированные стили.

Режим отображения (display-mode):

- display-mode проверяет, в каком режиме запущено ваше веб-приложение или сайт: в браузере, в полноэкранном режиме, как отдельное приложение и т.д.
- Значения: browser, fullscreen, minimal-ui, standalone, picture-in-picture.
- **Важно:** Не все режимы поддерживаются во всех браузерах. Проверяйте поддержку на https://caniuse.com/?search=%40media%20display-mode

```
/* Стили для полноэкранного режима */
@media (display-mode: fullscreen) {
    /* ... */
}
```

JAVARUSH

Логические операторы

Если нужно учесть несколько условий одновременно или исключить определенные ситуации? Для этого в CSS есть логические операторы:

- **Запятая** (,): Объединяет несколько условий. Стили применятся, если выполняется хотя бы одно из них.
- and: Объединяет условия, требуя выполнения всех.
- not: Исключает определённое условие. Стили применятся, если условие НЕ выполняется.
- only: Используется перед типом устройства, чтобы старые браузеры игнорировали медиа-запрос.

Важно:

- and используется между типом устройства и характеристиками, а также между несколькими характеристиками.
- not инвертирует только ту часть выражения, к которой он непосредственно применён.
- only полезен для обеспечения совместимости со старыми браузерами.

```
/* Стили для альбомной ориентации ИЛИ экранов
шириной до 960px */
@media (orientation: landscape), (max-width:
960px) {
    /* ... */
}

/* Стили для печати, НО только если ширина не
менее 320px */
@media print and (min-width: 320px) {
    /* ... */
}

/* Стили для экранов с шириной от 320px до 640px
*/
@media (min-width: 320px) and (max-width: 640px)
{
    /* ... */
}

/* Стили для ВСЕХ устройств, КРОМЕ экранов с
шириной от 380px */
@media not (screen and (min-width: 380px)) {
/* ... */
}
```

JAVARUSH

Синтаксис диапазонов

В современных браузерах появился новый, более интуитивный способ задавать диапазоны значений в медиа-запросах. Теперь вы можете использовать знакомые операторы сравнения <, >, <= и >=

```
@media (680px <= width <= 1280px) {
    /* ... */
}
```

Новый синтаксис поддерживается не всеми браузерами, поэтому проверяйте совместимость перед использованием.

JAVARUSH

Процентные значения в CSS

Процентные значения в CSS позволяют задавать размеры элементов относительно их родительских контейнеров. Это делает макет более гибким, так как элементы автоматически подстраиваются под изменения размеров родителя.

```
.container {
   width: 80%; /* Контейнер занимает 80% ширины своего родителя
(обычно это окно браузера) */
}

.box {
   width: 50%; /* Блок занимает 50% ширины своего родителя
(.container) */
   padding-top: 25%; /* Создаем соотношение сторон 1:2 */
}
```

Важно:

- Процентные значения всегда рассчитываются относительно родительского элемента.
- Если у родительского элемента не заданы явные размеры, то процентные значения могут вести себя непредсказуемо.
- Для создания соотношения сторон можно использовать padding-top или padding-bottom с процентным значением.

JAVARUSH

Относительные единицы (em и rem)

Относительные единицы измерения em и rem позволяют создавать гибкие и масштабируемые макеты, где размеры элементов зависят от размера шрифта. Это особенно полезно при создании адаптивных сайтов, которые должны хорошо выглядеть на разных устройствах с разными настройками шрифтов.

Единица em:

- Размер в em рассчитывается относительно размера шрифта родительского элемента.
- Если размер шрифта родителя меняется, то и размеры дочерних элементов, заданные в em, также изменятся пропорционально.
- Это может быть полезно для создания элементов, размер которых должен быть связан с размером окружающего текста.

Единица rem:

- Размер в rem рассчитывается относительно размера шрифта корневого элемента (<html>).
- Это делает размеры элементов более предсказуемыми и независимыми от вложенности элементов.
- rem идеально подходит для задания базовых размеров шрифта, отступов и других элементов, которые должны масштабироваться вместе с основным текстом страницы.

JAVARUSH

Гибкие изображения для разных устройств

Атрибут srcset в HTML позволяет указать несколько вариантов одного и того же изображения с разными размерами или разрешениями.

Браузер сам выберет наиболее подходящий вариант, учитывая ширину экрана пользователя и плотность пикселей его устройства. Это позволяет оптимизировать загрузку изображений и улучшить производительность сайта.

```
<img
    src="images/default.jpg"
    srcset="images/image-small.jpg 480w, images/image-medium.jpg 800w, images/image-large.jpg 1200w"
    alt="Пример адаптивного изображения">
```

В этом примере браузер выберет:

- image-small.jpg, если ширина контейнера меньше 480px.
- image-medium.jpg, если ширина контейнера от 480px до 800px.
- image-large.jpg, если ширина контейнера больше 800px.

JAVARUSH

Уточняем выбор изображения

Атрибут sizes работает в паре с srcset, чтобы дать браузеру ещё больше информации для выбора оптимального изображения. Он позволяет указать, какой ширины должно быть изображение в разных условиях отображения, например, на разных размерах экрана.

```
<img
    src="images/default.jpg"
    srcset="images/image-small.jpg 480w, images/image-medium.jpg 800w, images/image-large.jpg 1200w"
    sizes="(max-width: 600px) 480px, (max-width: 900px) 800px, 1200px"
    alt="Пример адаптивного изображения">
```

В этом примере браузер будет выбирать изображение, учитывая не только его ширину (srcset), но и условия отображения (sizes):

- Если ширина экрана меньше или равна 600px, будет использовано image-small.jpg.
- Если ширина экрана от 601px до 900px, будет использовано image-medium.jpg.
- Если ширина экрана больше 900px, будет использовано image-large.jpg.

JAVARUSH

is very importan

JAVARUSH

Домашнее задание

Уровень 13. Отзывчивая верстка.
Отзывчивая графика.
Лекции 0-4.