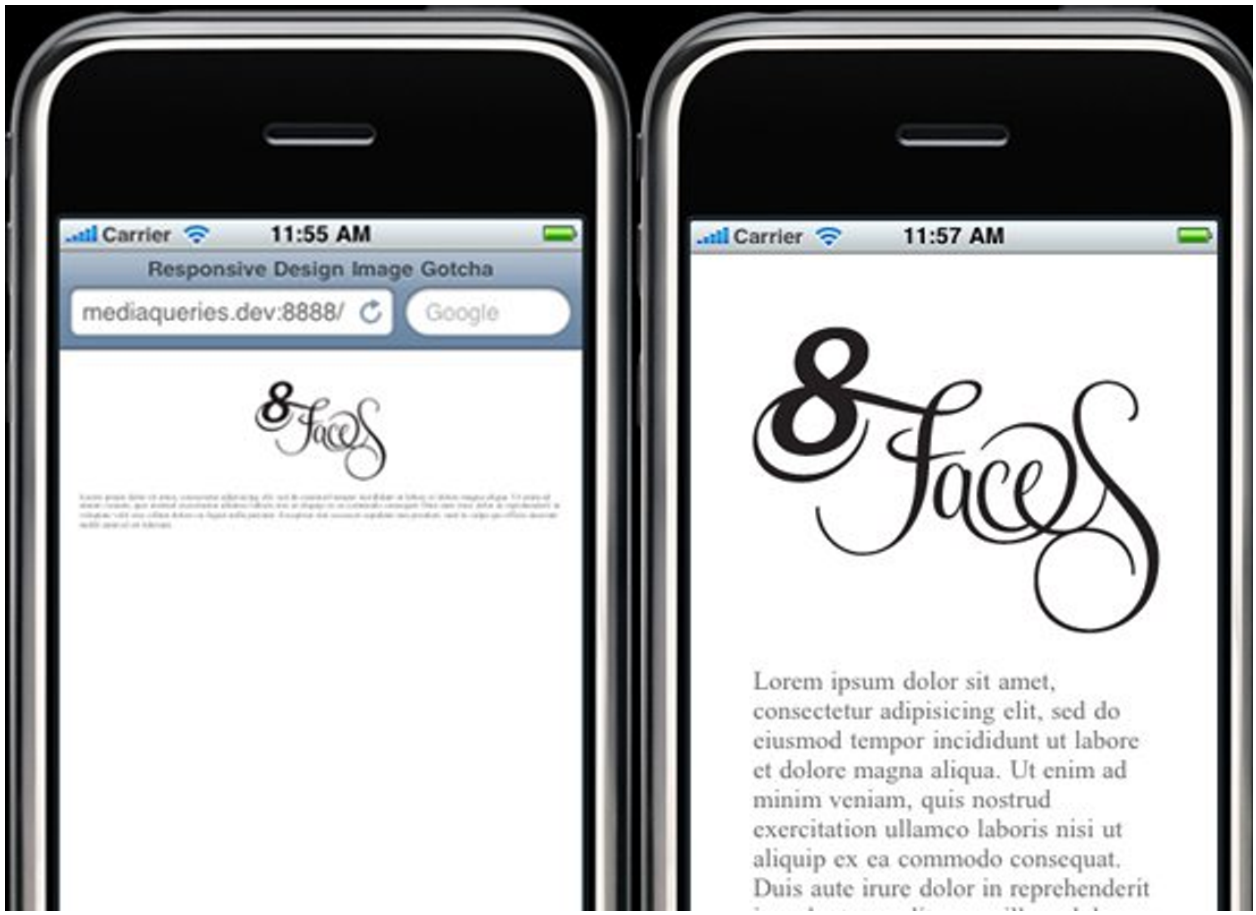
Домашнее задание

💎Вопросы

1. **Как сделать так, чтобы при просмотре на телефоне текст стал читаемым, а картинка - большой?**



Медиа-запросы: В CSS вы можете использовать медиа-запросы для настройки стилей в зависимости от размера экрана.
Например:

```
@media (max-width: 768px) {

  p {
    font-size: 16px;
  }


  img {
```

```
   width: 100%;
   height: auto;
  }
}

@media (min-width: 769px) {

}
```

В приведенном примере, при ширине экрана до 768px текст становится большим, а картинка растягивается на всю ширину экрана.

В приведенном примере, при ширине экрана до 768px текст становится большим, а картинка растягивается на всю ширину экрана.

Viewport Units: Вы можете использовать единицы измерения, зависящие от ширины и высоты видимой области экрана, такие как vw (viewport width) для текста и vh (viewport height) для картинок. Например:

```
p {
  font-size: 5vw;
}


img {
  width: 80vw;
  height: auto;
}
```

Здесь текст будет автоматически масштабироваться в зависимости от ширины экрана, и картинка будет занимать 80% ширины видимой области экрана.

Flexbox и Grid: Используйте CSS Flexbox и Grid для управления расположением элементов на разных экранах.

Оптимизированные изображения: Чтобы картинки загружались быстро на мобильных устройствах, используйте оптимизированные изображения с низким весом.

2. **В чём разница между отзывчивым и адаптивным веб-дизайном?**

Отзывчивый и адаптивный веб-дизайн - это два термина, которые часто используются в веб-разработке для создания сайтов, которые хорошо отображаются на разных устройствах и экранах. Однако они имеют некоторые различия в своем подходе:

Отзывчивый (Responsive) веб-дизайн:

Гибкий макет: Отзывчивый дизайн использует гибкие единицы измерения, такие как проценты и относительные единицы, чтобы элементы макета могли масштабироваться в зависимости от размера экрана.

Медиа-запросы: Основой отзывчивого дизайна являются медиа-запросы в CSS, которые позволяют настраивать стили и макет в зависимости от ширины экрана.

Плавные переходы: Отзывчивый дизайн обычно включает плавные переходы и анимации, чтобы обеспечить более гладкую адаптацию к изменению размеров экрана.

Основной фокус: Основной акцент в отзывчивом дизайне - это создание приятного пользовательского опыта на разных устройствах, поддерживая относительно одинаковый контент и структуру страницы.

Адаптивный (Adaptive) веб-дизайн:

Различные версии: Адаптивный дизайн создает разные версии сайта для разных устройств. Это означает, что для мобильных устройств может быть создан отдельный макет, отличный от десктопного.

Серверное управление: В адаптивном дизайне серверный компонент может определить тип устройства, на котором происходит просмотр, и отправить соответствующую версию сайта на устройство.

Больше контроля: Адаптивный дизайн предоставляет более высокий уровень контроля над тем, какой контент и структура будет представлена на разных устройствах.

Более ресурсоемкий: Поскольку адаптивный дизайн создает разные версии сайта, это может потребовать больше ресурсов и времени на разработку.

Оба подхода имеют свои преимущества и недостатки, и выбор между ними зависит от конкретных потребностей проекта. Отзывчивый дизайн обычно более гибок и экономичен, но адаптивный дизайн может предоставить более точный контроль над отображением на разных устройствах.

3. **Какие величины лучше использовать для шрифтов в гибком дизайне?**

Для шрифтов в гибком (отзывчивом) дизайне лучше использовать относительные величины, такие как проценты (%) или относительные единицы измерения (em, rem), вместо абсолютных единиц, таких как пиксели (px). Вот почему:

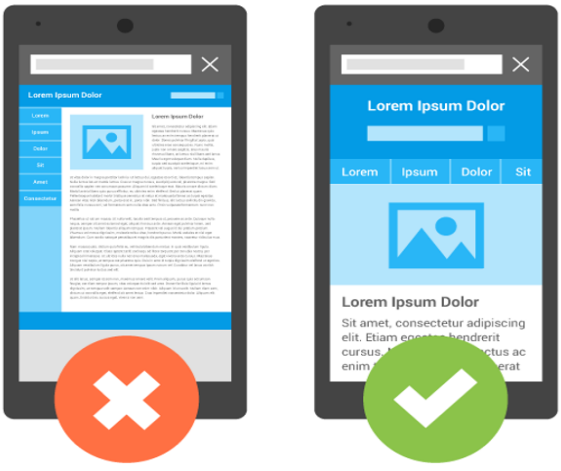
Проценты (%): Проценты позволяют создавать шрифты, которые масштабируются в зависимости от размера родительского элемента или экрана. Например, если вы установите размер шрифта в 100%, это будет 100% от размера шрифта родительского элемента. Это особенно полезно для создания гибкого макета.

em: em - это единица измерения, которая также зависит от размера шрифта родительского элемента. Например, если родительский элемент имеет размер шрифта 16 пикселей, то 1 em будет равен 16 пикселям. Это удобно, если вы хотите устанавливать отступы, ширину и высоту элементов в отношении к размеру шрифта.

rem: rem похож на em, но он всегда относится к размеру шрифта корневого элемента (обычно <html>), что делает его более прогнозируемым и управляемым в контексте всей страницы. Это особенно полезно в сложных макетах.

Использование относительных величин позволяет легко настраивать размеры шрифтов в зависимости от размера экрана или контекста без необходимости жестко задавать конкретные значения для каждого устройства. Это также делает ваш дизайн более гибким и улучшает его способность адаптироваться к различным устройствам и разрешениям экрана.

4. **Какой вид верстки использован на этой картинке? К какой категории шаблонов он относится?**



адаптивный дизайн

5. **Как задать стили для экранов шириной от 800 до 1200 пикселей?**

Вот пример медиа-запроса для стилей, применяемых для экранов шириной от 800 до 1200 пикселей:

```
@media screen and (min-width: 800px) and (max-width: 1200px) {
   body {
   background-color: lightgray;
   font-size: 16px;
  }
}
```

В этом примере, если ширина экрана находится в диапазоне от 800px до 1200px, то стили внутри медиа-запроса будут применяться.

6. **Приведите минимум 2 примера как подключать медиазапросы?**

Подключение медиа-запроса внутри файла CSS с помощью @media:

```
body {
  font-size: 16px;
}
@media screen and (min-width: 800px) and (max-width: 1200px) {
  body {
    font-size: 18px;
    background-color: lightgray;
  }
}
```

```
@media screen and (min-width: 1201px) {
  body {
    font-size: 20px;
    background-color: white;
  }
}
```

Подключение медиа-запроса внутри HTML-документа с использованием элемента <style>:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Пример медиа-запросов</title>
  <style>

    body {
      font-size: 16px;
    }

    @media screen and (min-width: 800px) and (max-width: 1200px) {
      body {
        font-size: 18px;
```

```
      background-color: lightgray;
    }
  }
  @media screen and (min-width: 1201px) {
    body {
      font-size: 20px;
      background-color: white;
    }
  }
 </style>
</head>
<body>


</body>
</html>
```

7. Как можно задавать гибкие изображения?

Использование относительных размеров: Вместо указания фиксированных размеров для изображений, вы можете задавать относительные размеры с помощью процентов или относительных единиц измерения (например, width: 100% или max-width: 100%). Это позволяет изображению автоматически масштабироваться в зависимости от размеров родительского контейнера.

```
img {
 width: 100%;
 max-width: 100%;
 height: auto;
```

```
}
```

Использование max-width для CSS: Установка max-width для изображений гарантирует, что они не будут шире, чем определенная ширина, но могут масштабироваться в меньшую сторону, если контейнер становится уже.

```
img {

    max-width: 100%; /* Максимальная ширина изображения будет равна ширине родительского контейнера */

  height: auto;

}
```

Использование srcset для HTML: Атрибут srcset позволяет браузеру выбирать разные версии изображения в зависимости от разрешения экрана и размера вьюпорта. Это особенно полезно для мобильных устройств с разными разрешениями экранов.

```
<img src="small.jpg" srcset="small.jpg 480w, medium.jpg 800w, large.jpg 1200w" alt="Гибкое изображение">
```

Использование форматов изображений: Используйте современные форматы изображений, такие как WebP, которые обеспечивают более высокое качество при более низком размере файла, что особенно важно для мобильных устройств.

```
<picture>

  <source srcset="image.webp" type="image/webp">

  <img src="image.jpg" alt="Гибкое изображение">

</picture>
```

8. **Как задать стили только для landscape поворота экрана? И что вообще такое landscape и чем он отличается от portrait?**

Для задания стилей только для landscape (горизонтального) поворота экрана можно использовать медиазапросы с ориентацией экрана. Landscape означает горизонтальную ориентацию экрана, когда ширина больше высоты, в то время как portrait означает вертикальную ориентацию экрана, когда высота больше ширины.

Вот пример медиазапроса для стилей, применяемых только в landscape ориентации:

```
@media screen and (orientation: landscape) {
  body {
    background-color: lightblue;
  }
}
```

В этом примере, стили внутри медиазапроса будут применяться только в случае, если экран находится в горизонтальной ориентации.

Различие между landscape и portrait заключается в ориентации экрана:

Landscape (горизонтальная ориентация): Экран шире, чем высок. Это часто соответствует горизонтальному повороту мобильного устройства или использованию устройства в горизонтальной ориентации.

Portrait (вертикальная ориентация): Экран выше, чем широк. Это обычная вертикальная ориентация, когда мобильное устройство или экран компьютера используется в вертикальном положении.

Медиазапросы с ориентацией экрана позволяют создавать адаптивные стили, которые изменяются в зависимости от ориентации устройства и предоставляют лучшее пользовательское взаимодействие на разных устройствах и ориентациях.

9. **Назовите минимум 3 способа как можно тестировать, как выглядит сайт при разных размерах экранов?**

Браузерные инструменты разработчика:

Многие современные браузеры, такие как Google Chrome, Mozilla Firefox, и Safari, предоставляют встроенные инструменты разработчика, которые включают режим адаптивного дизайна. Вы можете открыть эти инструменты, нажав клавишу F12 или правой кнопкой мыши на странице и выбрав "Исследовать элемент" или аналогичный пункт в меню. Затем вы можете выбрать разные размеры экрана для предварительного просмотра.

Веб-сервисы для тестирования адаптивности:

Существуют онлайн-сервисы и инструменты, такие как "BrowserStack", "CrossBrowserTesting" и "Responsinator", которые позволяют загрузить ваш сайт и просматривать его на разных устройствах и разрешениях экрана. Эти сервисы обычно предоставляют широкий спектр разных устройств и браузеров для тестирования.

Локальные устройства и эмуляторы:

Вы также можете тестировать ваш сайт на реальных устройствах или использовать программные эмуляторы. Например, вы можете использовать Android Studio для эмуляции мобильных устройств Android или Xcode для устройств iOS. Это предоставляет более точное тестирование на реальных устройствах.

Ручное изменение размера окна браузера:

Этот метод прост и доступен в любом браузере. Просто откройте ваш сайт в браузере и изменяйте размер окна браузера, чтобы проверить, как ваш сайт реагирует на разные размеры экрана. Это может быть не так точным, как другие методы, но это быстрый способ проверить базовую адаптивность.

10. **Самостоятельно изучите, как можно подключить несколько картинок разных размеров через один тег <img>?**

использовать атрибут srcset для подключения нескольких картинок разных размеров через один тег <img>. Этот атрибут позволяет браузеру выбирать наиболее подходящее изображение на основе характеристик устройства пользователя, таких как размер экрана и разрешение.

Пример использования атрибута srcset:

<img src="small.jpg" srcset="medium.jpg 1000w, large.jpg 2000w" alt="Изображение">

В этом примере:

src - это изображение, которое будет использоваться по умолчанию, если браузер не поддерживает атрибут srcset или не может выбрать наиболее подходящее изображение.
srcset - это список изображений с указанием их ширины (в пикселях или как "w"), разделенных запятыми. Браузер будет выбирать изображение, наиболее подходящее для текущего устройства пользователя.
В данном примере, если ширина окна браузера более 1000 пикселей, браузер загрузит medium.jpg, а если более 2000 пикселей, то large.jpg. Если устройство не поддерживает srcset или не соответствует ни одному из указанных размеров, будет загружено изображение, указанное в атрибуте src.

Этот метод позволяет оптимизировать загрузку изображений и улучшить производительность вашего сайта для разных устройств и разрешений экранов.