

A decorative graphic on the left side of the slide. It consists of a blue parallelogram and a light green parallelogram, both tilted at an angle. The blue shape is in the foreground, and the green shape is partially behind it. They are set against a dark blue background with faint, larger-scale geometric patterns.

Responsive



Intro

Отзывчивый дизайн - корректное отображение сайтов вне зависимости от устройства.

Есть три подхода (вида дизайнов): static, responsive и adaptive.

static: все размеры задаются в абсолютных величинах, на всех экранах сайт выглядит одинаково.

Responsive: Дизайн сайта реагирует на любые изменения размера браузера

Adaptive: Дизайн сайта реагирует на изменения браузера только в определенных точках

Часто применяется смешанный подход, когда дизайн реагирует на любые изменения браузера, а в определенных точках дополнительно переопределяется)



Core principles

Главные составляющие отзывчивого дизайна:

- гибкие изображения
- приоритет векторных изображений
- минимум абсолютных величин, гибкие контейнеры
- nesting
- media-queries



Images

Гибкие изображения:

```
img {  
  
width: yourRelativeMeasurement;  
  
height: auto;  
  
}
```

Хороший подход использовать свойства `min-width`, `max-width`

Так, как существует множество устройств с высоким разрешением - хороший подход - использовать векторную графику, т.к. при масштабировании изображения не теряют качество



Containers and behavior

Для большинства контейнеров также предпочтительно указывать относительные величины в комбинации со свойствами min-width/max-width. Обратите внимание, что на экранах с высоким разрешением экрана сайт может растягиваться довольно много, поэтому чаще всего следует указывать какой-то лимит для главного контейнера страницы, больше которого сайт не будет растягиваться.

Типичное поведение внутри контейнера: контент, который не помещается на одной строке - переносится ниже. Поэтому крайне не рекомендуется указывать фиксированную высоту контейнера, так как она может меняться в зависимости от контента.

Если контент организован колонками\карточками и т.п., на мобильных экранах эти блоки, чаще всего занимают по одному на строку, с увеличением ширины - столько, сколько помещается на странице.

На мобильных устройствах шрифты, кнопки и т.п., как правило, больше (для удобства)



Nesting

Часто, бывает, что есть группы элементов, расположение которых зависит друг от друга. По отдельности контролировать их очень тяжело. В таком случае стоит обернуть такие элементы в общий контейнер с позиционированием. Это тот редкий случай где оправданы `position absolute` + статические величины (внутри контейнера), однако сам контейнер часто получает относительные размеры. Такая техника подходит для небольших контейнеров, контент которых не нужно особо адаптировать под размеры экрана.



Media-queries

Медиазапросы - инструмент для применения разных стилей в зависимости от типа / размера устройства и других факторов (особенности пользовательских настроек и .т.п)

Синтаксис:

```
@media condition {
```

```
  selector {
```

```
    property: value;
```

```
  }
```

```
}
```



Media-queries. Conditions

В зависимости от типа устройства: `media-types` - (`all`, `screen`, `print`, `speech`) - можно перечислять через запятую

В зависимости от разных условий: `media-features` -

`width` (`width`, `min-width`, `max-width`), `height`, `orientation`, `resolution`, etc.



Media-queries. Logic

Логические операторы позволяют совмещать простые условия в более сложные.

Какие бывают:

- and (совмещение двух условий в одно): синтаксис - and
- or (применить стили в зависимости от одного из условий): синтаксис - перечисление через запятую
- not(исключение условия): синтаксис - not



Media-queries. Examples

```
@media screen and (min-width: 320px) {
```

```
  .item {
```

```
    width: 100%;
```

```
  }
```

```
}
```

```
@media screen and (min-width: 625px) and (max-width: 900px){
```

```
  .item {
```

```
    width: 50%;
```

```
    font-size: 14px;
```

```
  }
```

```
}
```



Crazy media-queries

```
@media all and (min-width: 901px) and (max-width: 3000px), not print and (min-height: 300px){
```

```
  .item {
```

```
    background: red;
```

```
  }
```

```
}
```