Занятие 8

Мобильная верстка adaptive и responsive

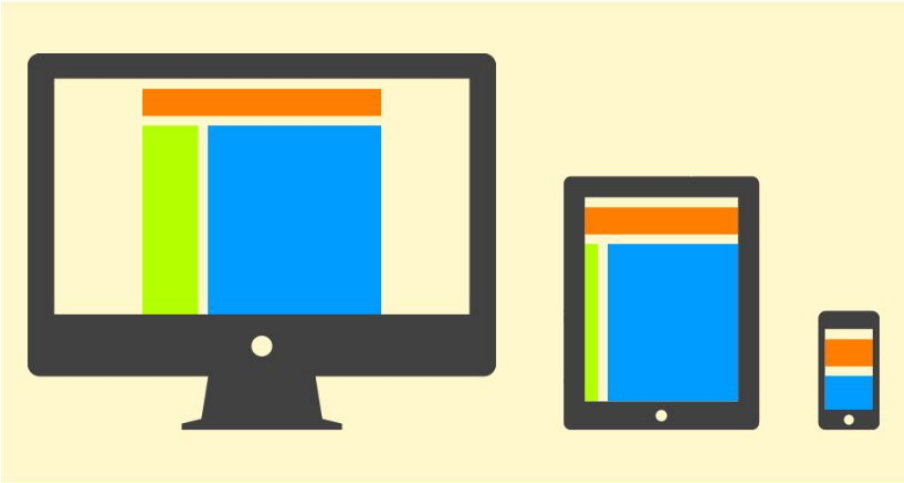
Новые относительные величины em, rem, vw, vh

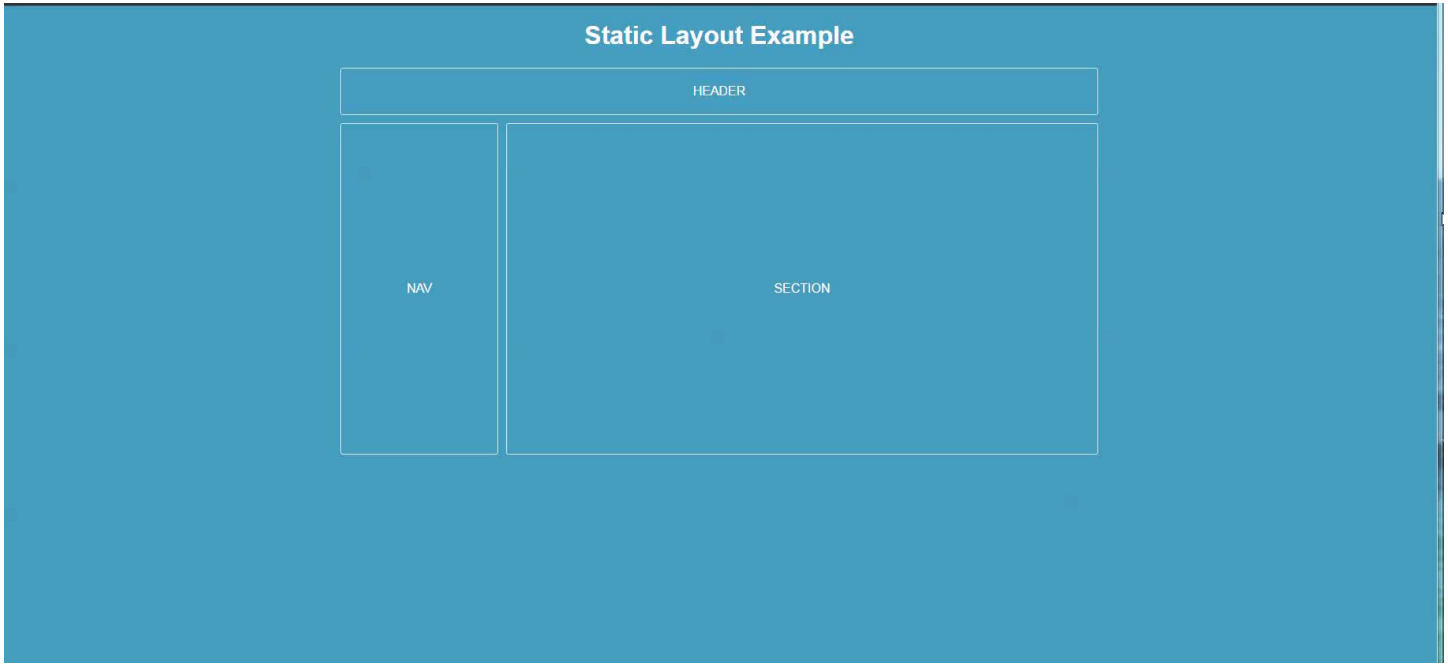
```
// В CSS мы уже знакомились с такими единицами измерения как px и %.
// Однако для мобильной верстки зачастую гораздо проще использовать относительные величины. Такие как em или rem

// // em - относительная величина, единица которой будет равна высоте шрифта блока родителя. Те, если у блока родителя font-size: 16px, то задав дочернему блоку width: 6.25em мы получим 100px, вычисляется по формуле: X(em) = size(px) / parent-font-size. В нашем случае 6.25(em) = 100 / 16(em). Но надо быть осторожным, потому, что размер шрифта также может изменяться в относительных величинах.
<div class="box" style="font-size: 16px;">
   <div class="box-container" style="font-size: 0.75em;"> // Тут высота шрифта изменилась на 0.75em (12px)
       <div class="box-item" style="width: 6.25em"></div> // Ширина будет высчитываться не относительно 16px, а относительно
   </div>                                                 // 0.75em, что в свою очередь 12px, и в итоге мы получим
</div>                                                    // 6.25em * (12px || 0.7em) = 75px.

// // rem - очень похож на em, за исключением того, что единица будет равна не высоте шрифта родителя, а высоте шрифта элемента <html>, которая по-умолчанию всегда 16px.

// // vw и vh - отношение к ширине и высоте видимой части браузера, используется как %, но без привязки к родительским элементам.
width: 100vw; // ширина будет равна 100% ширины окна браузера;
height: 100vh;// высота будет равна 100% высоты окна браузера;
```

Layout страницы

1. **Фиксированные макеты – Static**

px

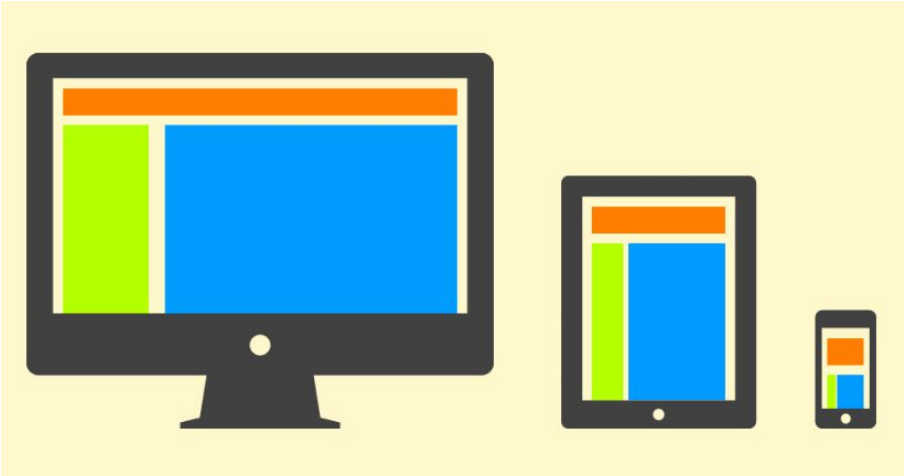


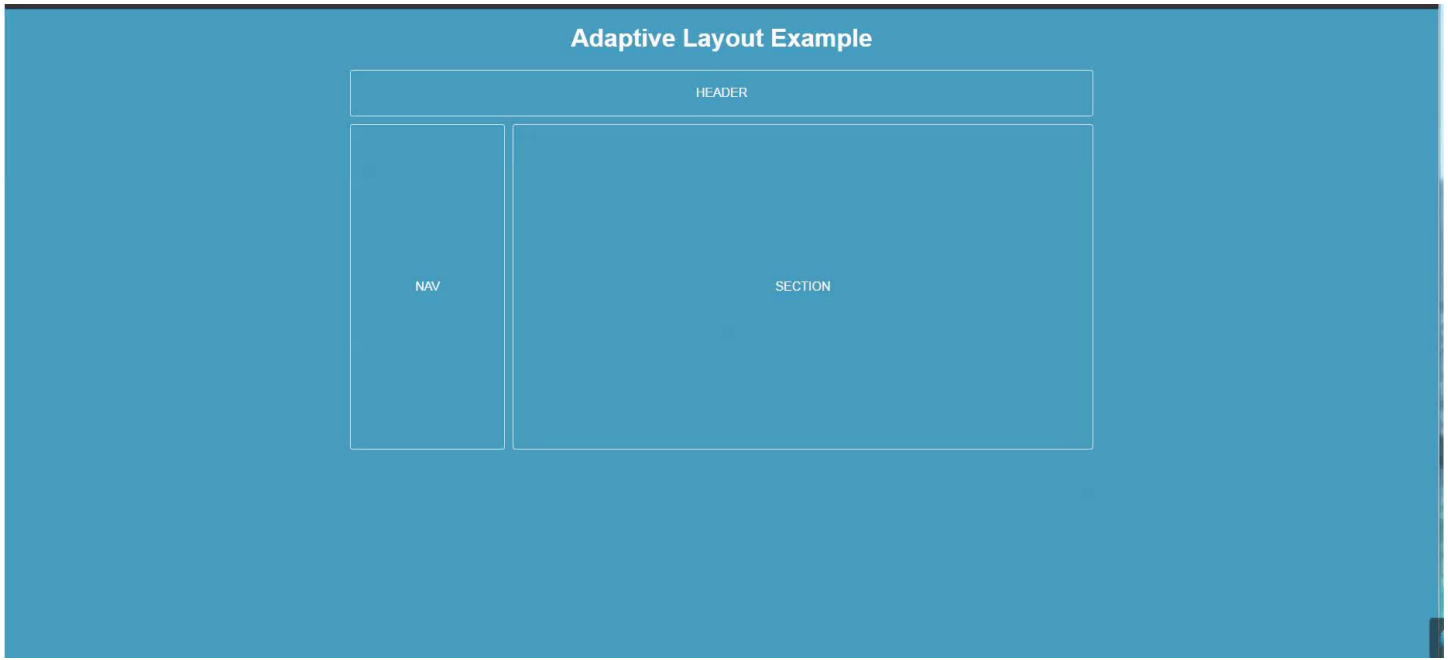

Static Layout Example
HEADER
NAV
SECTION

Layout страницы

2. **Резиновые макеты – Liquid**

%



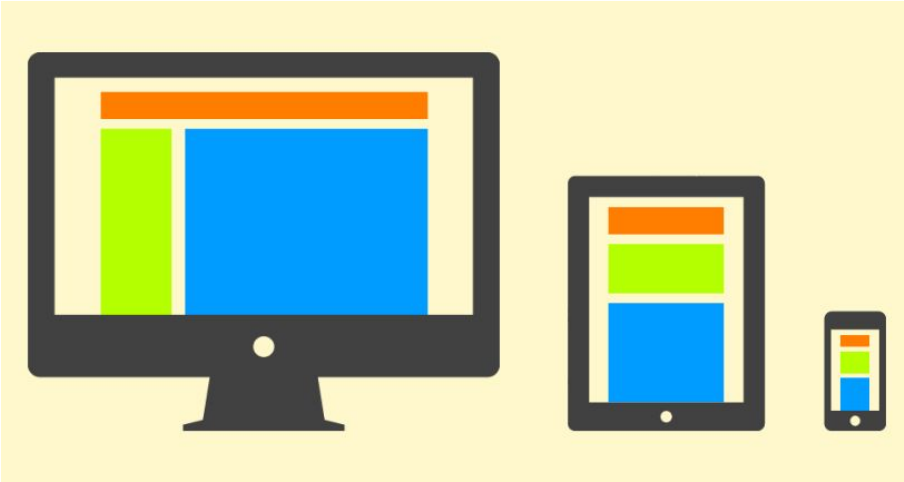

Adaptive Layout Example
HEADER
NAV
SECTION

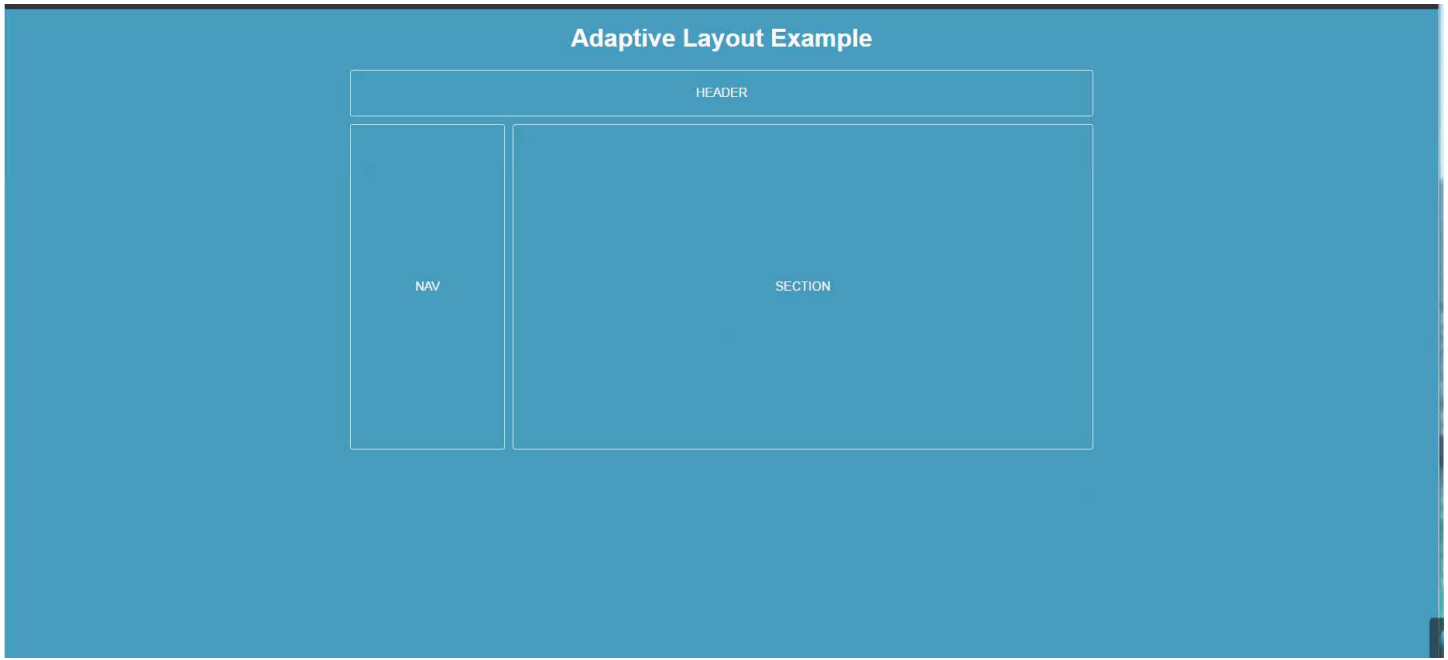
Layout страницы

3. **Адаптивные макеты – Adaptive**

px + media-queries + script(?)

Bootstrap v2



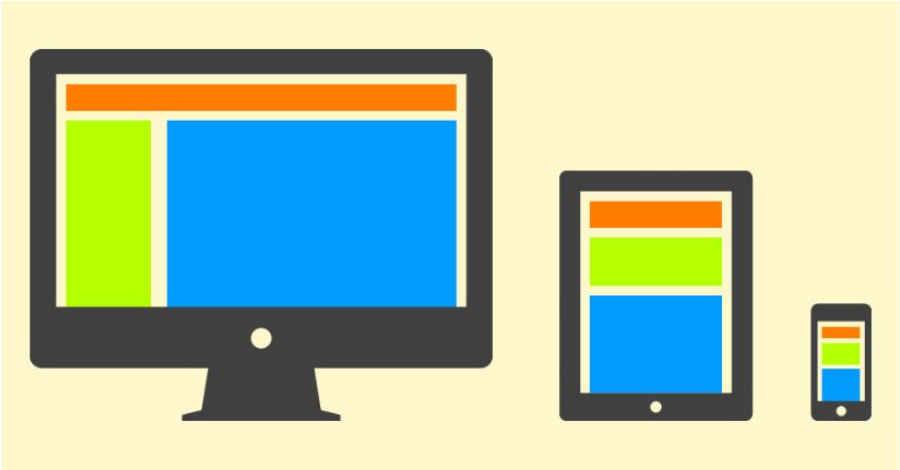

Adaptive Layout Example
HEADER
NAV
SECTION

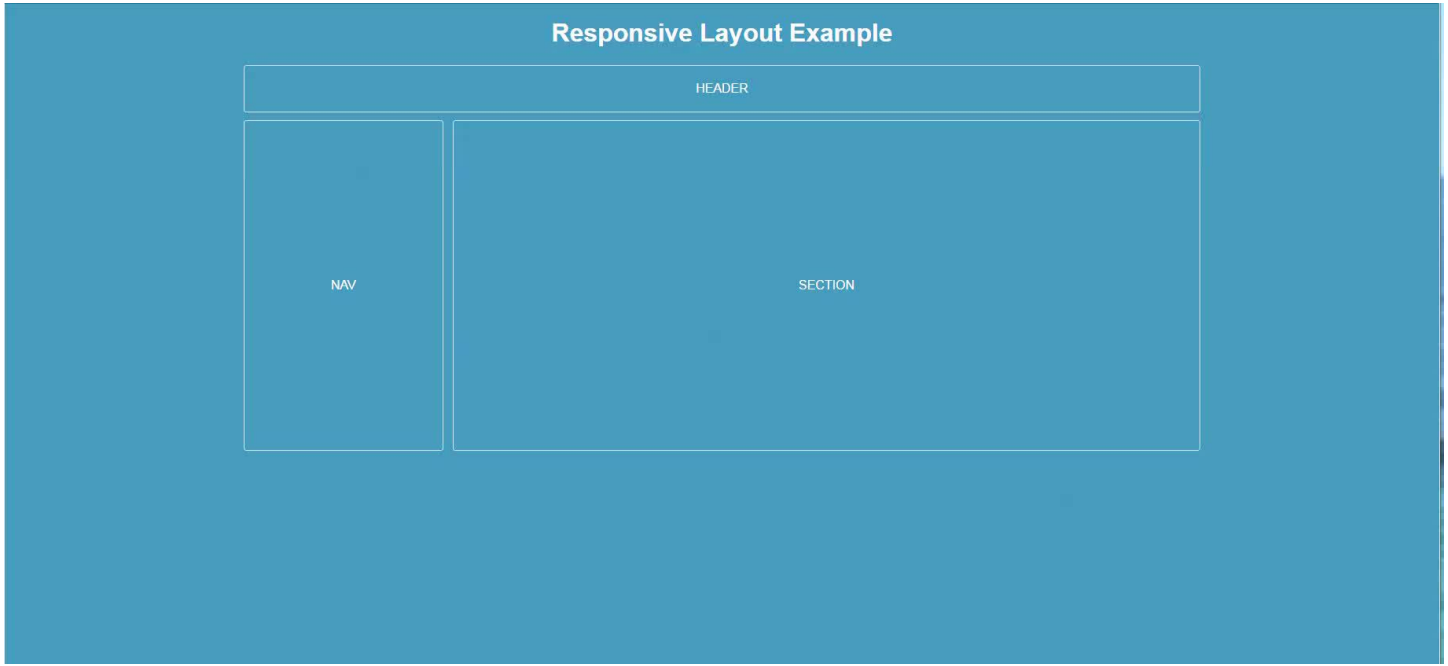
Layout страницы

4. **Отзывчивые макеты –Responsive**

% + media-queries

Bootstrap v3




Responsive Layout Example
HEADER
NAV
SECTION

Медиа-запросы css

```
// Для корректного отображения элементов на всех устройствах необходимо изменять размеры и позиционирование некоторых блоков.

// Для того, чтобы это реализовать необходимо получать тип устройства, и прописывать стили для него.

// В старых версиях CSS2 и HTML4.01 для этих целей можно было подключать отдельные файлы со стилями, указав в атрибуте media тип устройства из: all, handheld, print, screen, projection, tv и тд.
<link rel="stylesheet" type="text/css" media="screen" href="styles.css"> // Стили для экрана монитора;
<link rel="stylesheet" type="text/css" media="print" href="print.css">       // Стили для вывода на печать;

// Или @media (указанное устройство){};
@media screen {...}
@media handheld {...}

// @media - это обычный объект, внутри которого находится перечень элементов и их свойств. Описанные там свойства применяются к странице только тогда, когда наше устройство и его характеристики отвечают условиям этого @media.

// При этом все свойства внутри блока @media не имеют повышенных приоритетов. Поэтому медиа-запросы желательно описывать внизу *.css файла.
```

```
// С появлением CSS3 расширились возможности поддержки мобильных гаджетов. Кроме типа устройства были добавлены некоторые
характеристики устройства, такие как:
// // Минимальная/максимальная ширина экрана (min-width/max-width);
// // Минимальная/максимальная высота экрана (min-height/max-height);
// // Соотношение сторон (aspect-ratio);
// // Разрешение экрана (resolution);
// // Портретная/альбомная ориентация устройства (orientation);
// // И даже плотность пикселей для retina (-webkit-device-pixel-ratio).

.team{margin: 0 -15px;font-size: 0;}
.team-item{display: inline-block;vertical-align: top;width: 33.3%;padding: 0 15px;font-size: 16px;}

@media (max-width: 640px) { // Эти стили применяются только если ширина окна браузера менее 640px.
   .team{margin: 0;}
   .team-item{display: block; width: 100%; margin-bottom: 20px;}
}
// В программировании это было бы эквивалентно записи:
if(window.width <= 640){ ... }

// Кроме того, эти условия можно расширять, более конкретизируя необходимый девайс. Для этого появились логические операторы:
// // and - связывает условия между собой:
@media (min-width: 360px) and (max-width: 640px) { ... }
// // оператор запятая - перечисление, свойства применяются, если хотя бы одно из условий выполняется:
@media (orientation: portrait), (max-width: 640px) { ... }
// // not - отрицание, следует быть внимательным, тк not на все условие, если не используется запятая.
@media all, (not print) { ... }                    // Все устройства, кроме печати;
@media not screen and (max-width: 640px) { ... }  // Все устройства, кроме настольных пк, и с шириной экрана менее 640px
// // only - заблокировать медиа-запрос от старых браузеров, которые не поддерживают CSS3.
@media only screen { ... } // Сработает для всех пк, браузер которых знает CSS3.
```

Подходы в респонсив вёрстке.

```
// У каждого типа девайса есть разбежка разрешений. Так называемые чекпоинты. Я выделяю следующие:
// // Мобильный телефон от 320px до 479px;
// // Маленький планшет - мобильные в landscape view от 480px до 767px
// // Планшет от 768px до 1024px;
// // Ноутбук от 1025px (на данный момент уже давно 1280px, но промежуточное значение относится к пк) до 1440px;
// // Настольный компьютер от 1440px.

// Обычно для респонсив верстки в работе используются ограничения по ширине (min-width) или (max-width), остальные гораздо реже. Рассмотрим три основных подхода на примере стилей для планшета:

// // Подход first-mobile в таком подходе сперва верстается основной макет для мобильного телефона, в медиа-запросы добавляются стили для планшета и пк. На данный момент считается более перспективным за счет экономии трафика для мобильных устройств и других небольших преимуществ (например :hover).
@media (min-width: 480px) { ... }

// // Подход first-desktop полная противоположность first-mobile, при таком подходе сначала верстается макет для пк, а затем  добавляются медиа-запросы для планшета и мобильного.
@media (max-width: 1024px) { ... }

// // Точечный подход, стили описываются на каждый девайс отдельно. Хорошая практика при написании модулей и виджетов.
@media (min-width: 640px) and (max-width: 1024px) { ... }
```

Метатег viewport

```
// Вспоминаем другие мета-теги, которые нам известны.

// meta viewport необходим исключительно для мобильных устройств.

// В мобильном браузере неадаптированные сайты не вписываются в ширину экрана (тк скорее всего верстка бы сломалась), а открывается в виртуальном окне (viewport) которое отображает видимую область веб страницы (то, что можно увидеть без прокрутки). А пользователь увеличивая и уменьшая масштаб страницы может увидеть сайт полностью.

// Чтобы решить эту проблему в компании apple разработали meta-тег, который позвол отображение сайта на мобильном устройстве.

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">

// Давайте разберем параметры:
// // width/height - устанавливает ширину отображения. Можно задавать от 200px до 10000px или device-width, в таком случае ширина отображаемого сайта будет равна ширине видимой части окна мобильного телефона (тк определяющий показатель - ширина, высоту можно игнорировать);
// // initial-scale - устанавливает масштабирование, принимает значения от 0.1 до 10, по-умолчанию всегда 1;
// // user-scalable - принимает значения yes/no, указывает может ли пользователь масштабировать страницу вручную;
// // maximum-scale/minimum-scale - устанавливает максимальный предел масштабирования принимает значения от 0.1 до 10.

// Используется meta viewport в заголовке страницы, внутри тега <head>
```

Сетка сайта

```
// Сетка сайта - это важный и мощный инструмент дизайнера, представляет собой набор невидимых вертикальных направляющих, позволяющих располагать элементы, управлять вниманием пользователя, и создать математически точную верстку за короткий промежуток времени.
// Основной принцип создания сетки заключается в делении макета на равные части (колонки).
// Все колонки должны быть одинаковой ширины, и иметь одинаковые пробелы между собой.
// Самым распространенным и удобным на данный момент является деление на 12 колонок.

// Рассмотрим возможность разметить страницу для вэба на примере сайта, построенного на сетке в 12 колонок.
https://drive.google.com/open?id=1e3HOhfAGUxogaezrvCTPccq2iZrU34Kt.
// Зная количество колонок блока мы можем вычислить его ширину в % по формуле
width(%) = (100% / 12) * cols ~ 8.3 * cols.
// Теперь можно начинать записывать стили для нашей сетки.
```

```
.row{
    font-size: 0;
    margin: 0 -10px;
}
.col{
    display: inline-block;
    vertical-align: top;
    font-size: 16px;
    box-sizing: border-box;
    padding: 0 10px;
}
```

```
.col_1{width: 8.33333333%;}
.col_2{width: 16.66666667%;}
.col_3{width: 25%;}
.col_4{width: 33.33333333%;}
.col_5{width: 41.66666667%;}
.col_6{width: 50%;}
.col_7{width: 58.33333333%;}
.col_8{width: 66.66666667%;}
.col_9{width: 75%;}
.col_10{width: 83.33333333%;}
.col_11{width: 91.66666667%;}
.col_12{width: 100%;}
```

Мобильная сетка

```
// Для мобильных девайсов обычно изменяется размер шрифта, размер блоков и их обтекание. Сетка может сократиться до 6 колонок.

// Пример мобильного дизайна с сеткой. https://drive.google.com/open?id=1ou4ongI_Z1A-dDs3wtokYD3IGTq_skyh.

// Теперь можно поработать с классами для мобильных колонок и медиа-запросами.

@media (max-width: 1024px) {
    .col-td_1{width: 8.33333333%;}
    .col-td_2{width: 16.66666667%;}
    .col-td_3{width: 25%;}
    .col-td_4{width: 33.33333333%;}
    .col-td_5{width: 41.66666667%;}
    .col-td_6{width: 50%;}
    .col-td_7{width: 58.33333333%;}
    .col-td_8{width: 66.66666667%;}
    .col-td_9{width: 75%;}
    .col-td_10{width: 83.33333333%;}
    .col-td_11{width: 91.66666667%;}
    .col-td_12{width: 100%;}
}

// Увидят только с планшета.

@media (max-width: 640px) {
    .col-md_1{width: 16.66666667%;}
    .col-md_2{width: 33.33333333%;}
    .col-md_3{width: 50%;}
    .col-md_4{width: 66.66666667%;}
    .col-md_5{width: 83.33333333%;}
    .col-md_6{width: 100%;}
}

// Увидят только с мобильного телефона.

// В какой последовательности должны быть записаны эти три блока css свойств?
```