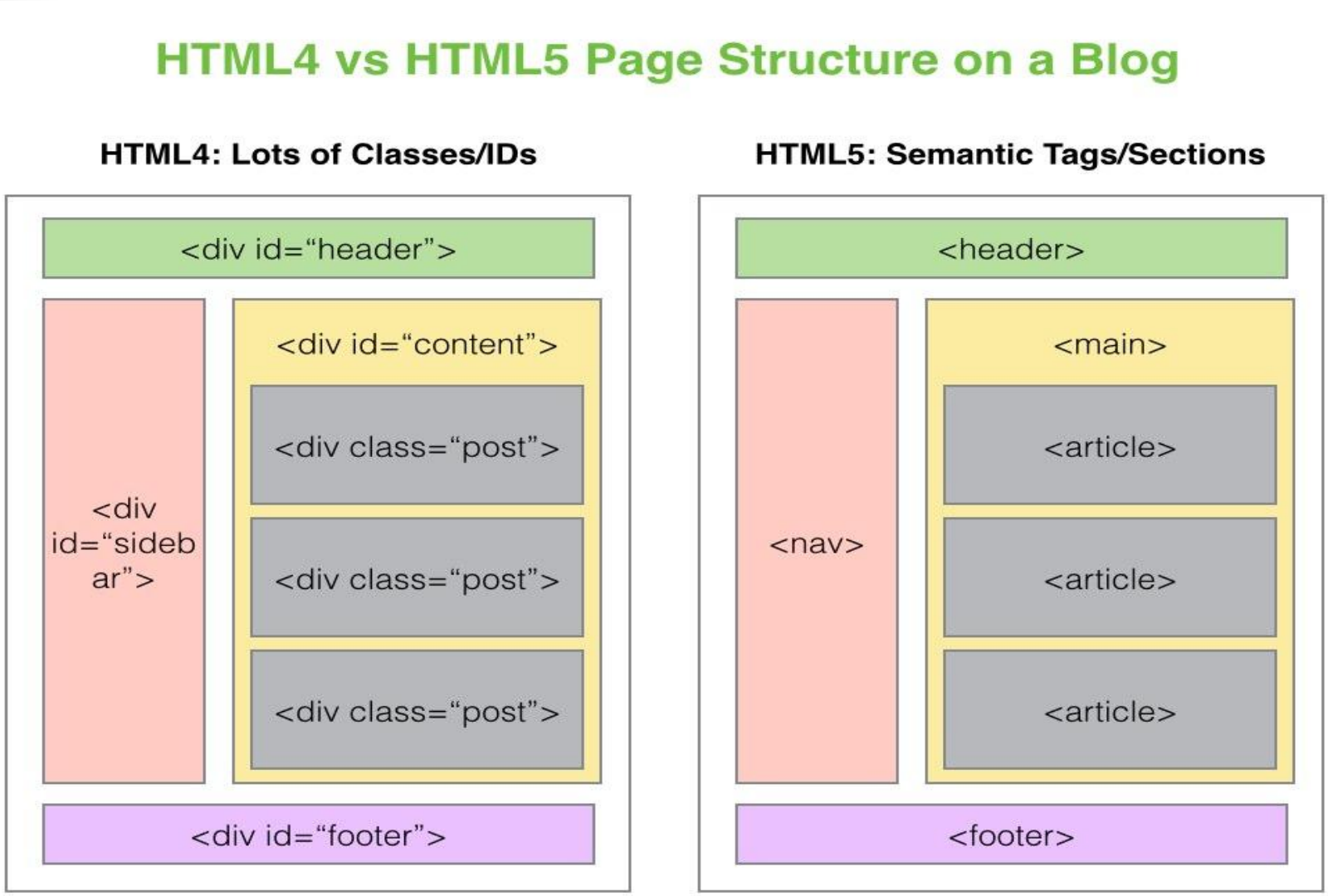
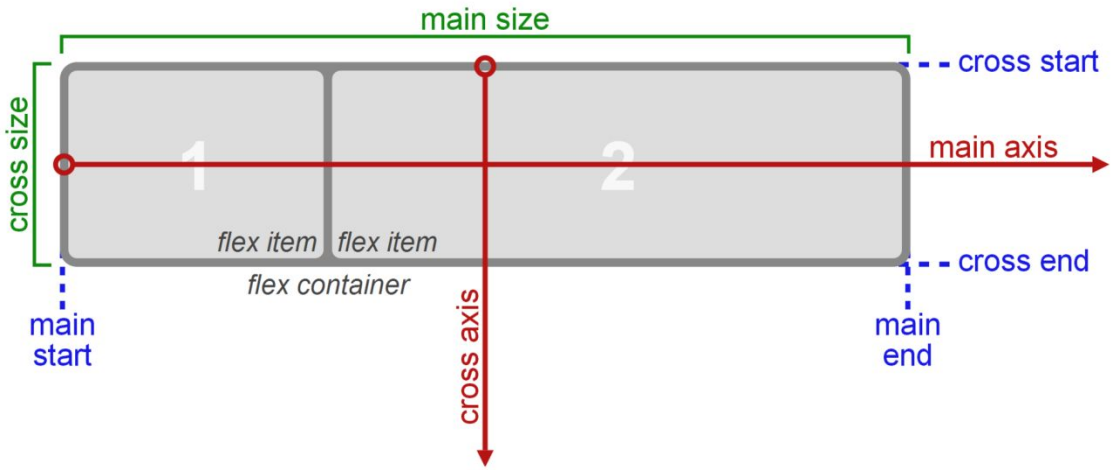
Семантика HTML5



Flexible Box Layout Module

CSS flexbox представляет собой способ компоновки элементов, в основе лежит идея оси.

Flexbox состоит из **flex-контейнера** и **flex-элементов**. Flex-элементы могут выстраиваться в строку или столбик, а оставшееся свободное пространство распределяется между ними различными способами.



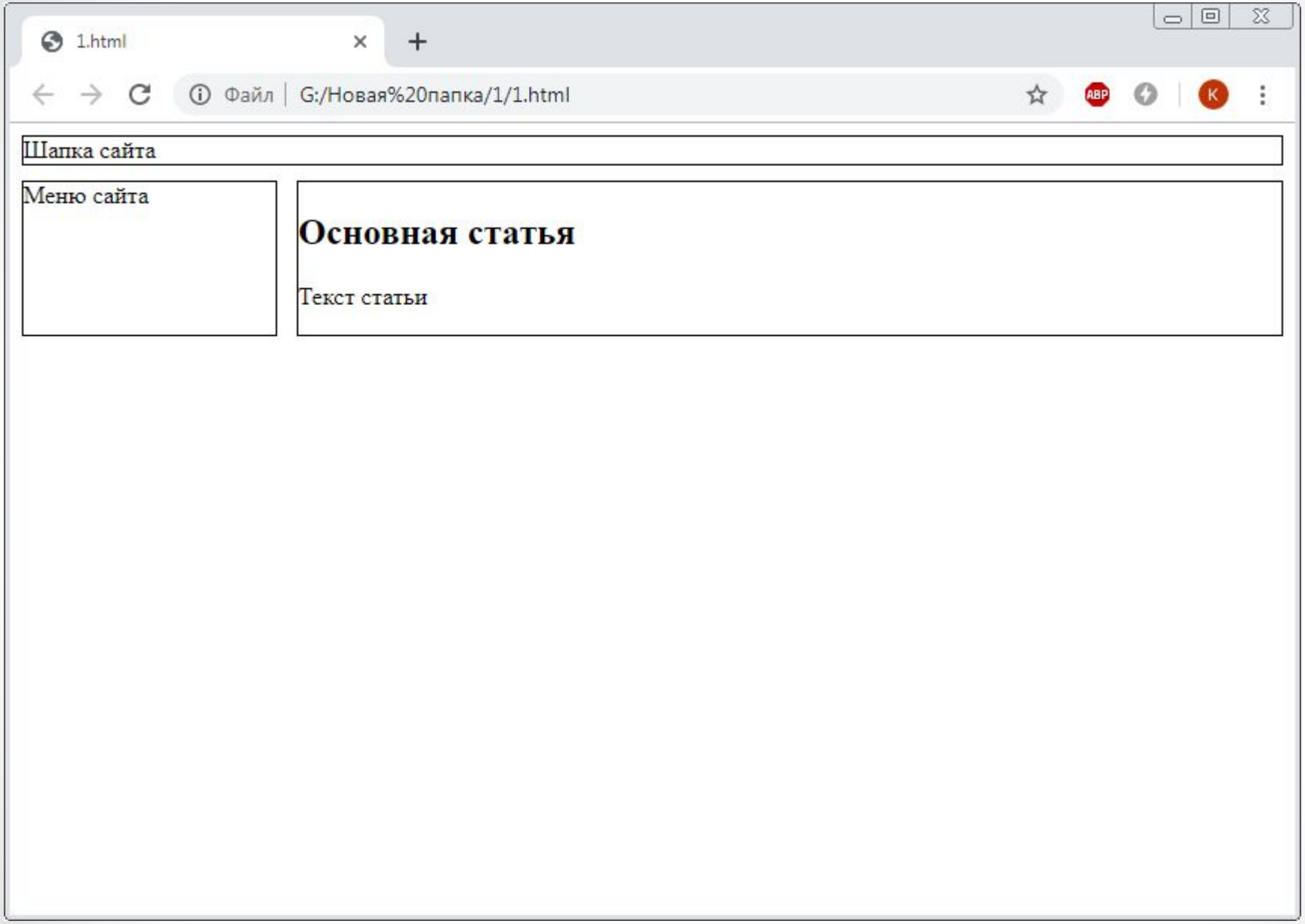
Flexible Box Layout Module

```
<!doctype html>
<head>
    <link rel="stylesheet" href="a.css">
</head>
<body>
    <header>Шапка сайта</header>
    <main>
     <nav>Меню сайта</nav>
     <article>
          <h1>Основная статья</h1>
          <p>Текст статьи</p>
     </article>
    </main>
</body>
</html>
```

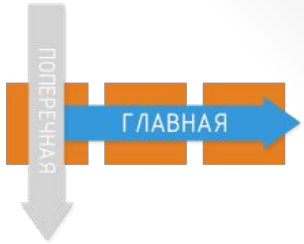
Flexible Box Layout Module

a.css:

```
header { border:1px solid black; margin-bottom:10px; }

main    { display:flex; width:100%; justify-content:space-between; }

nav      { flex-basis:20%; flex-shrink:0; flex-grow:0;
    border:1px solid black;
    }

article  { flex-basis:78%; flex-shrink:0; flex-grow:0;
    border:1px solid black;
    }
```


1.html
Файл | G:/Новая%20папка/1/1.html
Шапка сайта
Меню сайта
Основная статья
Текст статьи

Flexbox контейнер



Направление главной оси

flex-direction: row | row-reverse | column | column-reverse

Управление переносом

flex-wrap: nowrap | wrap | wrap-reverse

Выравнивание по главной оси

justify-content: flex-start | flex-end | center | space-between
space-around

Выравнивание по поперечной оси

align-items: flex-start | flex-end | center | baseline | stretch

Flexbox элемент

Базовый размер:
flex-basis: величина в любых единицах измерения

Коэффициент роста:
flex-grow: величина в любых единицах измерения

Коэффициент сжатия:
flex-shrink: величина в любых единицах измерения

Порядок элементов
flex-order: число

Выравнивание по поперечной оси:
align-self: flex-start | flex-end | center | baseline | stretch

Преимущества flexbox

Все блоки очень легко делаются “резиновым”, что уже следует из названия “flex”. Элементы могут сжиматься и растягиваться по заданным правилам, занимая нужное пространство.

Расположение элементов в html не имеет решающего значения. Его можно поменять в CSS. Это особенно важно для некоторых аспектов адаптивной верстки.

Элементы могут автоматически выстраиваться в несколько строк/столбцов, занимая все предоставленное место.

Есть понятие начала и конца, а не права и лева.

Синтаксис CSS правил очень прост и осваивается довольно быстро.

Введение в медиа-запросы

<link rel="stylesheet" href="a.css" media="all">

В CSS3 доступны следующие параметры media:

all	**Все типы. используется по умолчанию**
braille	Устройства, основанные на системе Брайля
embossed	Принтеры, использующие систему Брайля
handheld	Смартфоны и аналогичные им аппараты
print	**Принтеры и другие печатающие устройства**
projection	Проекторы
screen	**Дисплей, экран монитора**
speech	Речевые синтезаторы
tty	Устройства с фиксированным размером символов
tv	Телевизоры

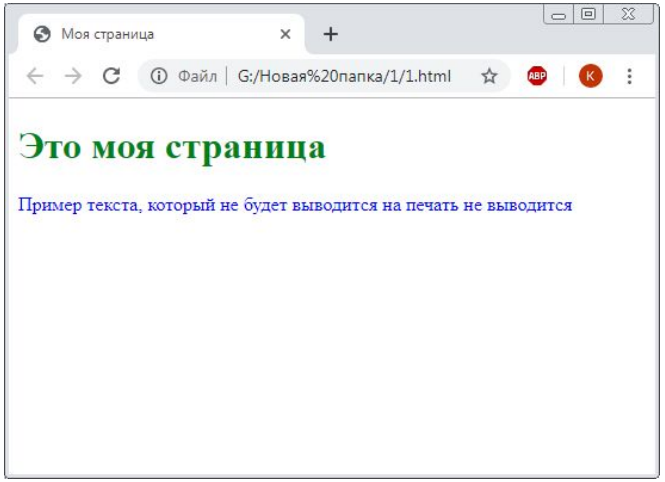
Пример

```
<!doctype html>
<html>
    <head>
     <title>Моя страница</title>
     <link rel="stylesheet" href="comp.css" media="screen">
     <link rel="stylesheet" href="print.css" media="print">
    </head>
    <body>
     <h1>Это моя страница</h1>
     <p>
          Пример текста, который не будет выводится на
          печать не выводится
     </p>
    </body>
</html>
```

Пример

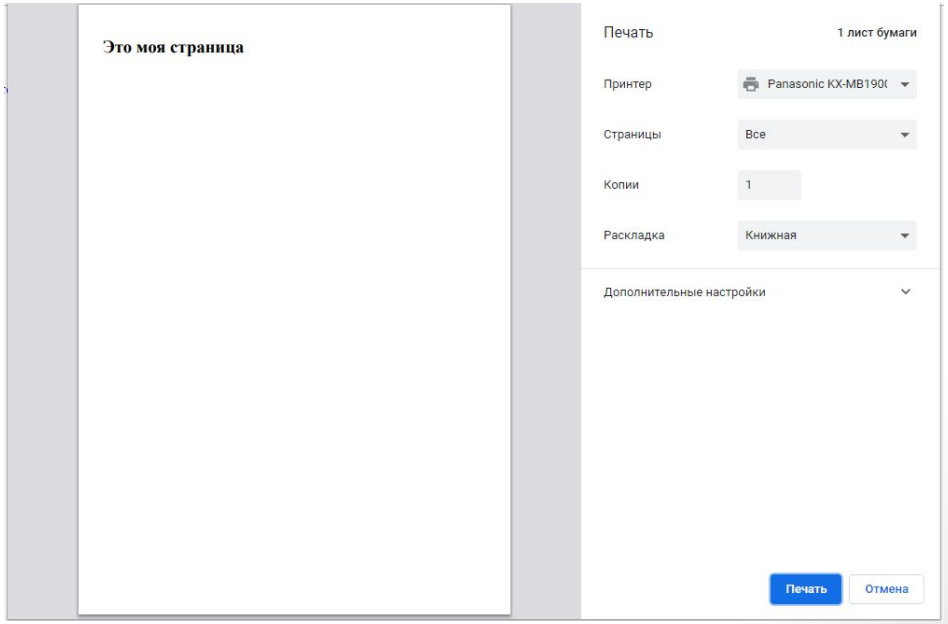
comp.css (для экранов):

```
h1 {color:green;}
p {color:blue;}
```



print.css (для принтера):

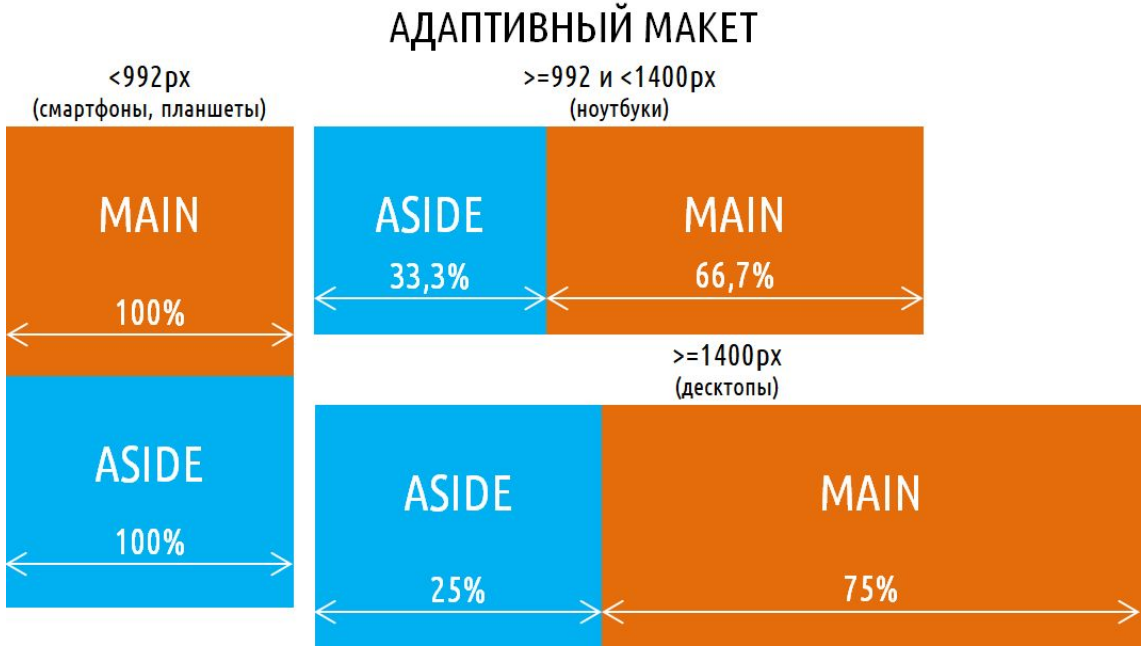
```
h1 {color:black;}
p {display:none;}
```




My Photos
Notes
Calculator
Camera

Адаптивный сайт

Адаптивный сайт - это сайт, который может «приспосабливаться» под различные устройства (ширину рабочей области окна браузера). Т.е. на одних устройствах он может иметь одну структуру, а на других - другую.



Медиа-функции

```
<link rel="stylesheet" href="a.css"
      media="screen and (min-width:1440px)">
```

Логические операторы:

and для объединения

not для отрицания

Медиа-функции:

width

device-width

color

orientation

Resolution

Префиксы:

min-

max-

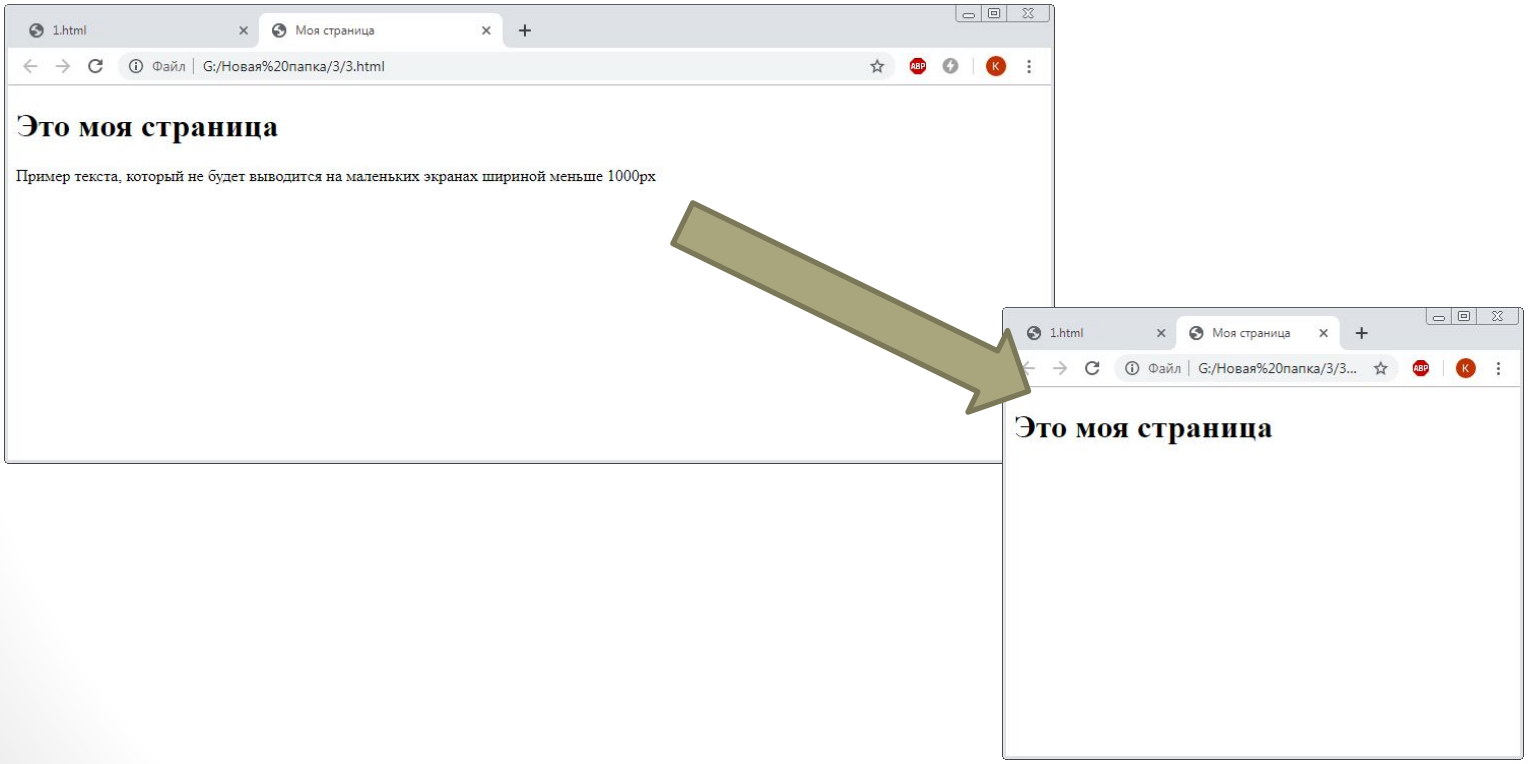
Пример

```
<!doctype html>
<html>
<head>
    <title>Моя страница</title>
    <link rel="stylesheet" href="comp.css"
        media="screen and (max-width:1000px)">
</head>
<body>
    <h1>Это моя страница</h1>
    <p>Пример текста, который не будет выводится
    на маленьких экранах шириной меньше 1000px</p>
</body>
</html>
```

Пример

comp.css:

```
p {display:none;}
```



Альтернативное подключение

Вместо

```
<link rel="stylesheet" href="a.css"
media="screen and (min-width:1440px)">
```

можно использовать медиа-запросы и медиа-функции в самом css-файле:

```
@media screen and (min-width:1440px) {
…css-правила…
}
```

Viewport

- Мобильные браузеры отображают страницы в виртуальном «окне» (viewport, вьюпорт), которое, как правило, шире экрана устройства. Поэтому им не нужно сжимать макет каждой страницы в крошечное окно (что может сломать многие сайты, не оптимизированные под мобильные устройства). Пользователи могут изменять и масштабировать видимую область, чтобы видеть разные части страницы.

- В мобильной версии Safari появился "мета тег viewport" (viewport meta tag), который позволяет веб-разработчикам контролировать размер окна просмотра и масштаб страницы. Многие другие мобильные браузеры также поддерживают этот тег, хотя он не является частью какого-либо веб-стандарта.

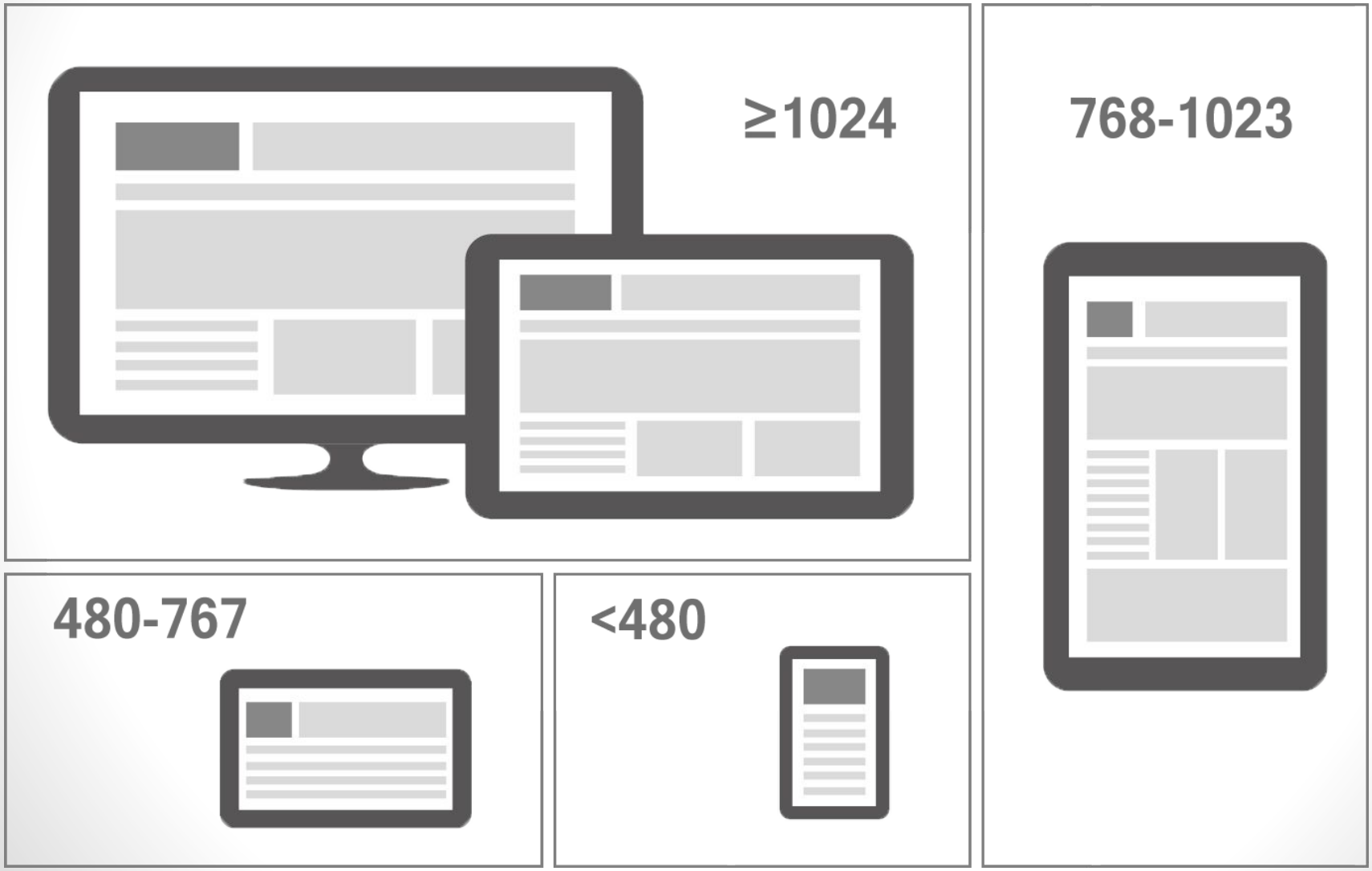
Viewport

<meta name="viewport" content="width=device-width, initial-scale=1">

Width - определяет размер окна просмотра. Он может быть установлен на определенное количество пикселей, скажем,width=600 или на специальное значение device-width, которое означает ширину экрана в пикселях CSS в масштабе 100%.

Свойство initial-scale контролирует уровень масштабирования при первой загрузке страницы.

Реперные точки



Итоговая структура

```
<head>
    <meta name="viewport" content="width=device-width">
    <link rel="stylesheet" href="desk.css"
     media="screen and (min-width:1024px)">
    <link rel="stylesheet" href="tab.css"
     media="screen and (min-width:768px)
     and (max-width:1023px)">
    <link rel="stylesheet" href="bigmob.css"
     media="screen and (min-width:480px)
     and (max-width:767px)">
    <link rel="stylesheet" href="smallmob.css"
     media="screen and (max-width:479px)">
    …
</head>
```

ВЕРСТКА ТИПОВЫХ ЭЛЕМЕНТОВ

«Сложный» логотип

Под сложным логотипом будем понимать элемент, содержащий в себе кроме текста еще и изображение, либо текст, не воспроизводимый стандартным набором шрифтов.

Логотип, как правило, содержит название компании (или товара, которому посвящена страница) и располагается в шапке. Используется в SEO для продвижения компании или продукта путем использования тега h1:

<h1>Сибирская корова</h1>

Однако, в браузере это будет выглядеть просто текстом без изображения. Замена на

<h1><img src="logo.png" alt="Сибирская корова"></h1>

приведет к потере содержательной части тега h1, доступной для обработки поисковыми системами, и, как следствие, ухудшению положения сайта в поисковой выдаче по этой ключевой фразе.

«Сложный» логотип

Для того, чтобы удовлетворить и потребности SEO, и вывести на экран изображение, можно использовать следующее css-правило:

```
h1 {
    display:block;     /* определяем контейнер */
    width: 200px;     /* с шириной и высотой      */
    height: 150px;    /* по размеру логотипа      */
    background: url(logo.png);
    /* далее используем одно из определений */
    font-size:0;
    text-indent: -10000px;
}
```

Горизонтальное меню

Меню сайта является аналогом содержания книги, которое, в свою очередь, представляет из себя список глав. С точки зрения верстки для списков есть соответствующие пары тегов ul-li, либо ol-li:

```
<nav>
    <ul>
        <li><a href="/x"> Меню 1</a></li>
        <li><a href="/y"> Меню 2</a></li>
        <li><a href="/z"> Меню 3</a></li>
    </ul>
</nav>
```

Однако, визуально это выглядит следующим образом:

- Меню 1
- Меню 2
- Меню 3

Горизонтальное меню

Задать горизонтальный вид можно одним из следующих css-правил:

1. Минималистичный вариант:

```
ul {list-style-type:none;}
ul li {display:inline}
```

2. Кнопки равной ширины:

```
ul {list-style-type:none;}
ul li {display:inline-block;width:50%;}
```

3. Обратный порядок:

```
ul {list-style-type:none;}
ul li {float:right;}
```

У каждого из этих способов есть свои достоинства и недостатки.