

План практической работы для преподавателя

Оглавление

Введение	1
Подключение внешнего файла CSS.....	1
Практическое задание	2
Обзор основных типов селекторов.....	3
Обзор основных атрибутов.....	3
Шрифты	3
Подключение шрифта с помощью Google fonts	4
Цвет.....	6
Фон.....	6
Блочная модель	6
Позиционирование	7
Figma и CSS	10

Введение

CSS (англ. Cascading Style Sheets — каскадные таблицы стилей) — формальный язык описания внешнего вида документа, который написан с использованием языка разметки HTML. Подготовкой и выпуском спецификации занимается консорциум W3C - <http://www.w3.org/Style/CSS/>

CSS используется применительно к языкам разметки HTML и XHTML, но может также применяться к любым XML-документам, например, к SVG или XUL.

Подключение внешнего файла CSS

В предыдущей лабораторной работе мы уже подключали CSS с помощью атрибута style внутри тегов. Такой способ является очень громоздким и устаревшим. Гораздо удобнее выделить все стили в отдельный документ. Данный способ является универсальным и самым распространенным. Он позволяет создать стили CSS в отдельном текстовом документе, который имеет расширение .css (например: style.css). Плюсы данного способа в том, что все стили для всего сайта записываются в одном документе, что существенно облегчает дальнейшую работу с внешним видом сайта. На сайте всегда присутствуют элементы, которые на всех страницах сайта отображаются одинаково (например, это может быть меню, шапка, подвал, какие-либо кнопки и прочие элементы).

Пример подключения файла .css в файле .html

```
<head>
<link rel="stylesheet" href="https://web-legko.ru/mysite.css">
/*Подключение файла стилей с стороннего сайта*/
<link rel="stylesheet" href="css/style.css">
/*Подключение файла стилей из директории сайта*/
</head>
```

Значение атрибута тега `<link>` — `rel` остаётся неизменным независимо от кода, как приведено в данном примере. Значение атрибута `href` задаёт путь к CSS-файлу, он может быть задан как относительно, так и абсолютно. Таким способом есть возможность подключать стили с другого сайта.

Практическое задание

1. Создайте страницу HTML, назовите ее `index.html`.
2. В данном документе напишите «скелет» страницы HTML

```
<!DOCTYPE html>
<html>
<head>
<title>Практическая работа «Подключение CSS к сайту»</title>
</head>
<body>
</body>
</html>
```

3. Внутри тега BODY поместите тег DIV

4. К данному тегу с помощью атрибута `style` примените стиль CSS который будет изменять цвет текста на красный

```
<div style="color: red;"> Практическая работа CSS </div>
```

5. Сохраните и просмотрите результат

6. Далее в части HEAD поместите тег `<style>` и запишите в него следующий код

```
<style>
div {
color: green;
width: 200px;
height: 200px;
background-color: #000006;
}
</style>
```

7. Сохраните и просмотрите результат в браузере

8. Заметьте, что цвет текста в данном случае для одного и того же элемента задается двумя способами сразу.

9. Далее в папке где мы сохранили файл `index.html`, создаем файл `style.css`. и записываем в него следующий код

```
body {
font-size: 20px;
color: yellow;
}
div {color: #887700;
float: right;
text-align: center;
}
```

10. Сохраните файл и возвращаемся к редактированию страницы HTML

11. Теперь необходимо подключить файл со стилями к странице. С помощью кода `<link rel="stylesheet" href="style.css">` (так как файл со стилями у нас находится в той же папке что и файл `index.html`, то мы прописываем относительный путь для файла стилей)

Данный код помещаем в часть HEAD.

Посмотрите на результат.

Обзор основных типов селекторов

Селекторы

Ключевое понятие в CSS – **селектор**, которое представляет собой правило для применения стиля. Браузер для каждого элемента пытается применить стиль в соответствии с заданным правилом. Стиль содержит набор свойств.

Различают простые селекторы, которые будут применены к указанному элементу (в примере к любому заголовку h1, h2, h3):

```
h1 { font-family: sans-serif }
h2 { font-family: sans-serif }
h3 { font-family: sans-serif }
```

Группы селекторов (эквивалентно выше приведённому фрагменту):

```
h1, h2, h3 { font-family: sans-serif }
```

Селекторы класса:

```
*.pastoral { color: green } /* все элементы, имеющие class=pastoral */
```

Или

```
.pastoral { color: green } /* все элементы, имеющие class=pastoral */
```

А также:

```
h1.pastoral { color: green } /* только элементы h1, имеющие class=pastoral */
```

Селекторы идентификатора ID:

```
h1#chapter1 { font-family: sans-serif } /* для <h1 id="chapter">...</h1> */
#chapter1 { font-family: sans-serif } /* для любого элемента с id="chapter1" */
```

Селекторы атрибутов:

```
h1[class] { font-family: sans-serif } /* элемент имеет class */
h1[class="fancy"] { font-family: sans-serif } /* элемент имеет class="fancy" */
*[title] { font-family: sans-serif } /* любой элемент, имеющий заголовок */
```

Селектор потомков (устанавливает иерархию применения):

```
tr h1 { font-family: sans-serif } /* <tr><td><h1>...</h1><td></tr> */
```

Псевдоклассы (особый вид динамически атрибутов, который меняются в зависимости от определенных действий):

```
a:link /* ссылки, которые не были посещены */
a:visited /* посещенные ссылки */
a:hover /* выделенная в данный момент ссылка */
a:active /* активные ссылки */
```

Обзор основных атрибутов

Шрифты

При оформлении страницы доступны следующие семейства шрифтов:

- **Serif** - шрифты с засечками. Обычно используются при бумажной печати. Наиболее используемый шрифт – Times.
- **Sans-serif** - шрифты без засечек. Подходят для заголовков. Наиболее используемые шрифты этого семейства – Arial, Helvetica, Verdana.
- **Monospace** - шрифт, обеспечивающий равную ширину символов. Используется для вывода примеров кода, поскольку внешний вид этого текста будет соответствовать текстовой консоли. Наиболее используемый шрифт – Courier.

- Fantasy, Cursive – декоративные и курсивные шрифты. Не рекомендуются к использованию, поскольку шрифты этой группы не обязательно присутствуют на компьютере, на котором будут просматривать html страницу.

Выбор семейства шрифта осуществляется свойством font-family. Пример отображения семейств стилей:

```
<p style="font-family: serif">Serif: Образец текста</p>
<p style="font-family: sans-serif">Sans-serif: Образец текста</p>
<p style="font-family: cursive">Cursive: Образец текста</p>
<p style="font-family: fantasy">Fantasy: Образец текста</p>
<p style="font-family: monospace">Monospace: Образец текста</p>
```

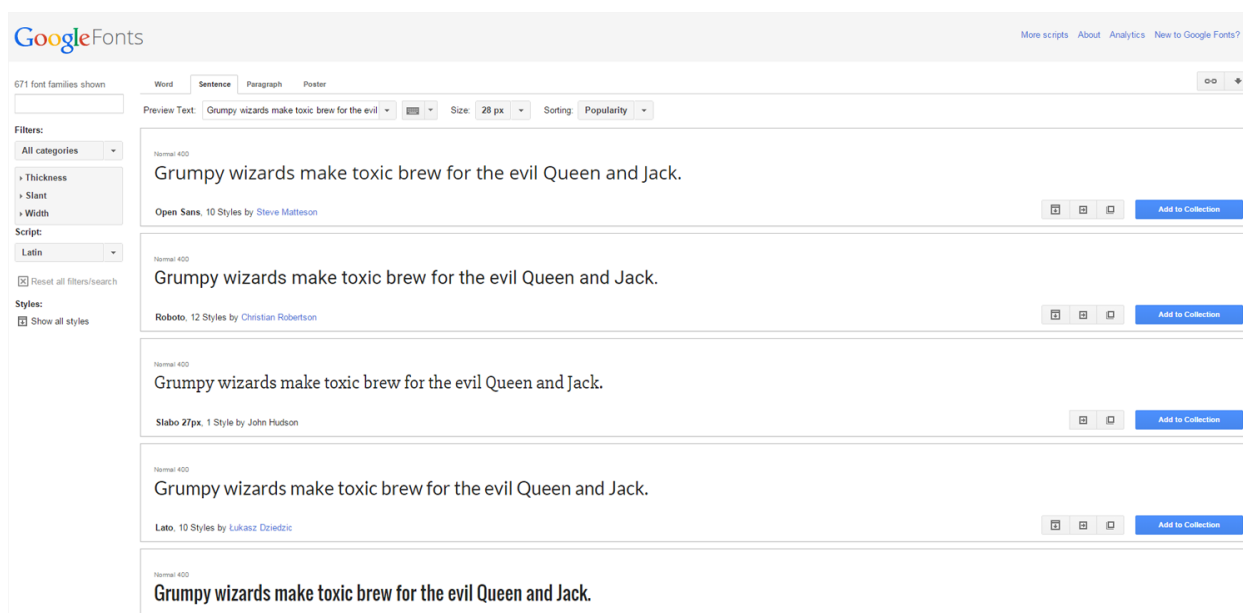
Попробуйте подключить эти шрифты и посмотрите на результат в браузере.

Подключение шрифта с помощью Google fonts

Вышеприведенный способ использования шрифтов имеет огромный минус – вы ограничены в количестве шрифтов. Вам придется довольствоваться лишь теми их вариантами, которые, вероятнее всего, установлены на большинстве компьютеров.

Каким же образом можно увеличить выбор шрифтов, чтобы сделать дизайн страницы индивидуальным, добавить оригинальности? На помощь приходят веб-шрифты.

Компания Google дает возможность легко подключать любой шрифт из коллекции Google Fonts. Всё, что вам необходимо сделать, чтобы начать использовать понравившийся шрифт, – указать несколько настроек на странице шрифта в Google, после чего скопировать специальную ссылку на этот шрифт и добавить в ваш веб-документ.



Чтобы подключить шрифт с сайта Google fonts:

1. Выберите начертание

Первым делом на странице выбранного шрифта отобразятся варианты его начертания, а также иконка спидометра, которая означает не что иное, как скорость загрузки шрифта. Чем больше стилей для шрифта вы выбираете, тем больше времени потребуется на его загрузку. Поэтому рекомендуется выбирать только те варианты начертания, которые планируется использовать.

2. Выберите алфавит

Далее на странице есть возможность выбрать набор символов: латиница, кириллица и т. д. В зависимости от шрифта, в нем могут быть доступны не все варианты алфавита. Аналогично

предыдущему пункту, лучше поставить галочку только напротив того алфавита, который понадобится.

3. Добавьте код на сайт

Далее Google предлагает подключить шрифт одним из нескольких способов: или стандартным, или через директиву `@import`, или с помощью JavaScript. Мы рассмотрим первые два варианта.

Первый способ подразумевает добавление в HTML-код ссылки на сервер Google, откуда и скачивается шрифт. Вам необходимо скопировать уже готовый кусок кода и поместить его между тегами `<head>` `</head>` в вашем HTML-документе. Пример:

```
<head>
<link
href='http://fonts.googleapis.com/css?family=Roboto&subset=latin,cyrillic'
rel='stylesheet' type='text/css'>
```

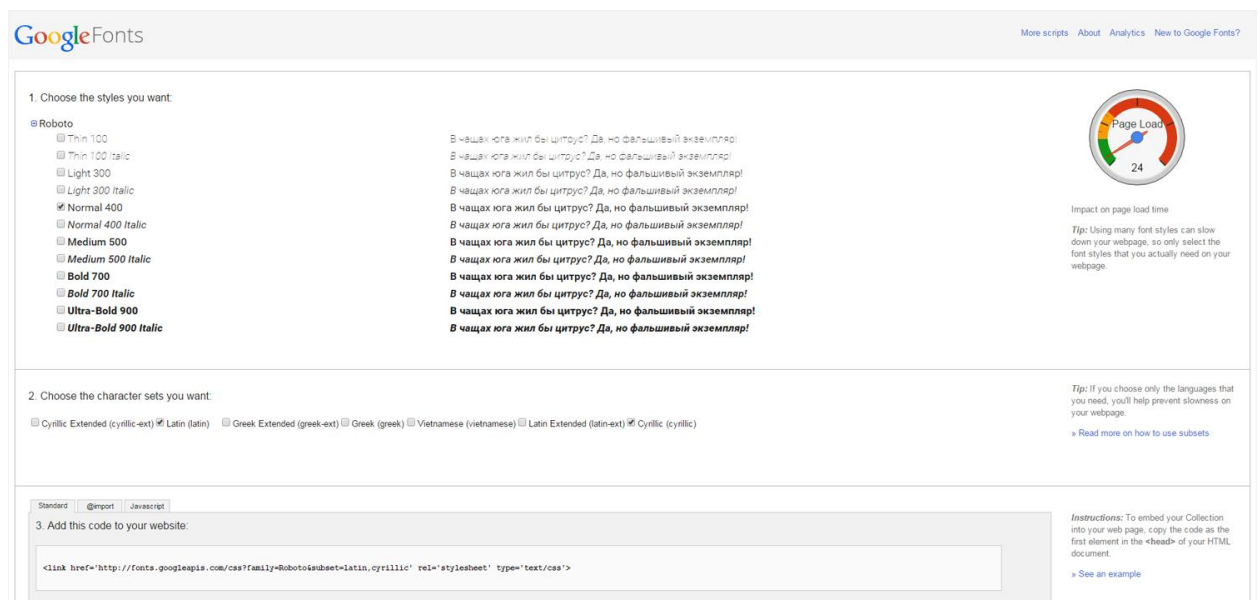
...

```
</head>
```

Второй способ – подключение шрифта с помощью директивы `@import`. Готовый код находится во второй вкладке пункта 3 на странице выбранного Google-шрифта. Его нужно добавить в самое начало вашей таблицы стилей (в противном случае файл не импортируется). Выглядит код примерно так:

```
@import
url(http://fonts.googleapis.com/css?family=Roboto&subset=latin,cyrillic);
```

Особенность первого способа заключается в том, что вам понадобится добавлять ссылку на шрифт в заголовок каждой страницы, где планируется его использовать. Это легко осуществить на сайтах с небольшим количеством страниц, но проблематично для крупных ресурсов. Второй способ удобен тем, что код можно поместить в самое начало внешней таблицы стилей, и тогда все страницы, к которым подключена эта таблица, получат необходимый шрифт, который будет загружаться с помощью директивы `@import`



Google Fonts

More scripts About Analytics New to Google Fonts?

1. Choose the styles you want:

☒ Roboto

- ☐ Thin 100
- ☐ Thin 100 Italic
- ☐ Light 300
- ☐ Light 300 Italic
- ☒ Normal 400
- ☐ Normal 400 Italic
- ☐ Medium 500
- ☐ Medium 500 Italic
- ☐ Bold 700
- ☐ Bold 700 Italic
- ☐ Ultra-Bold 900
- ☐ Ultra-Bold 900 Italic

В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!
В чашках юга жил бы цитрус? Да, но фальшивый экземпляр!

2. Choose the character sets you want:

☐ Cyrillic Extended (cyrillic-ext) ☒ Latin (latin) ☐ Greek Extended (greek-ext) ☐ Greek (greek) ☐ Vietnamese (vietnamese) ☐ Latin Extended (latin-ext) ☒ Cyrillic (cyrillic)

3. Add this code to your website:

```
<link href='http://fonts.googleapis.com/css?family=Roboto&subset=latin,cyrillic' rel='stylesheet' type='text/css'>
```

Instructions: To embed your Collection into your web page, copy the code as the first element in the <head> of your HTML document.

See an example

4. Создайте стиль

После осуществления предыдущих шагов можно начать применять шрифт. Как записывается такое правило CSS, вы уже видели ранее:

```
p {
```

```
font-family: 'Roboto', sans-serif;
}
```

Цвет

Свойство `color` задаёт цвет шрифта. Возможно указание по названию цвета (red, green, lime), так и указание точного его значения в системах RGB, HSL, а также RGBA, HSLA (добавлен канал прозрачности).

Полный перечень допустимых значений см. <http://www.w3.org/TR/css3-color/>

Пример:

```
body {color:blue;}
h1 {color:#00ff00;}
h2 {color:rgb(255,0,0);}
```

Отметим, что выбор цветов, цветовой схемы является очень важным элементом создания дизайна сайта. Существуют определенные методики подбора совместимых цветов. Имеются бесплатные средства в Интернете, например <http://colorscheme.ru/>.

Фон

Фон элементов может быть задан однородным цветом, одиночным или мозаично расположенным изображением. Подробнее см.

http://www.w3schools.com/css/css_background.asp

Используются следующие свойства:

`background-color` – однородный цвет константой или кодом в одной из допустимым систем цветности.

Пример:

```
div {background-color:#b0c4de;}
```

`background-image` – фоновое изображение.

Пример:

```
body {background-image:url('paper.gif');}
```

`background-repeat` – флаг мозаичного размножения изображения.

Пример:

```
body {
background-image:url('gradient2.png');
background-repeat:repeat-x;
}
```

`background-attachment` – указывает, будет ли изображение смещаться при скроллинге или будет оставаться на месте.

Пример:

```
background-attachment:fixed;
```

`background-position` – указывает позицию размещения изображения на устройстве отображения.

Пример:

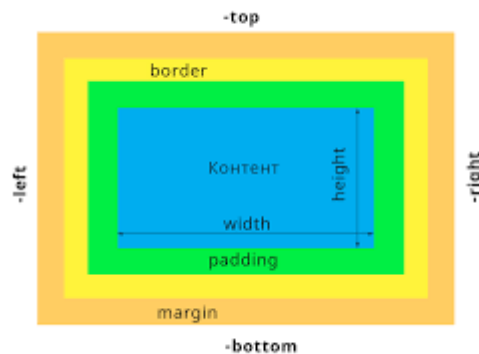
```
body {
background-image:url('img_tree.png');
background-repeat:no-repeat;
background-position:right top;
}
```

Блочная модель

Блочная модель лежит в основе модели визуализации элементов и описывает прямоугольники, формирующиеся вокруг всех элементов в соответствии с их иерархией в дереве элементов документа. См. подробнее в <http://www.w3.org/TR/CSS2/box.html>

Отступы и границы

В соответствии с блочной моделью, для любого элемента имеется область самого элемента (content), внутренние поля (padding), рамка или граница (border), внешние границы (margin). Для каждой области может быть задан размер. Наличие внутреннего поля позволяет сформировать рамку на заданном расстоянии от содержимого элемента. Наличие внешнего поля позволяет установить отступ между рядом расположенными элементами. На рисунке представлена схема расположения этих составных частей элемента. Обратите внимание на то, что рамка может иметь толщину, задаваемую соответствующим свойством CSS и также участвует в расчёте внешнего размера элемента.



В CSS любой элемент имеет свойства `width` и `height`, которые устанавливают размер «содержимого» в процентах, пикселах или автоматически вычисляемые. Размер отступа задаётся свойствами `'padding-top'`, `'padding-right'`, `'paddingbottom'`, `'padding-left'` или единственным свойством `padding`, которому указывается один общий размер отступа или последовательно отступ сверху, справа, снизу, слева.

```
div { padding: 10px }
blockquote { padding-top: 0.3em }
h1 {
background: white;
padding: 1em 2em;
}
```

Позиционирование

CSS поддерживает 4 вида позиционирования:

- Статическое (static)
- Абсолютное (absolute)
- Относительное (relative)
- Фиксированное (fixed)

Используются следующие термины:

- Нормальный поток – который означает обычное поведение браузера при отображении данных.
- Окно просмотра браузера – окно браузера, в котором отображается содержимое документа.

Элементы контейнеры могут быть размечены с использованием позиционирования. В качестве элементов-контейнеров может быть любой элемент, однако обычно используется специальный элемент `div`. Все элементы, включенные в элемент-контейнер, будут размещены в его границах.

Статическое позиционирование назначается у всех элементов по умолчанию и означает нормальное следование элементов. В явном виде спецификатор `static` применяется для перекрытия унаследованных стилей.

Абсолютное позиционирование

Абсолютное позиционирование подразумевает указание расположения относительно его блока-контейнера или корневого элемента `html`. При этом, как только появляется абсолютное позиционирование, элемент выпадает из нормального потока и всегда будет позиционироваться относительно контейнера, независимо от остального содержимого страницы. Рассмотрим простейший пример позиционирования. В данном случае позиция будет совпадать с левым верхним отступом от окна отображения.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<style type="text/css">
    #content {
        position: absolute;
        left: 200px;
        top: 100px;
        border: 1px solid green;
    }
</style>
</head>
<body>
    <div id="content">
        <p>Некоторый текст для проверки размещения элемента.
    </p></div>
</body>
</html>
```

Рассмотрим другой пример, включающий абсолютное позиционирование относительно другого блока.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<style type="text/css">
    #main {
        position: absolute;
        left: 100px;
        top: 50px;
        border: 1px solid black;
        padding: 0 100px 100px 0 }
    #content {
        position: absolute;
        left: 50px;
        top: 20px;
        border: 1px solid green;
    }
</style>
</head>
<body>
<div id="main">Главное меню:
    <div id="content">
        <p>Некоторый текст для проверки размещения элемента.</p>
    </div>
</div>
</body>
</html>
```


В данном случае элемент с идентификатором content смещен относительно элемента с идентификатором main. Обратите внимание на то, что его смещение не зависит от текста, который непосредственно помещен в <p>, а зависит только от заданной позиции в стиле.

Относительное позиционирование

Несмотря на то, что позиционирование контрастирует в названии с абсолютным позиционированием, следует помнить, что смещение в этом случае вычисляется не относительно соседних элементов, а относительно нормального потока. В следующем примере блок с идентификатором content смещен относительно нормального потока, но элемент, расположенный за ним, будет отображен так, как будто никаких изменений в потоке не было. Обратите внимание на то, что этот блок будет менять положение при изменении размеров окна браузера.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<style type="text/css">
    #content {
        position: relative;
        left: 50px;
        top: 20px;
        border: 1px solid green; }
</style>
</head>
<body>
    <p>Проект Mozilla официально выпустил релиз web-браузера Firefox 15. Кроме
    того, выпущен корректирующий релиз ветки с длительным сроком поддержки –
    Firefox 10.0.7, в котором отмечается только исправление уязвимостей и
    серьезных
    ошибок.</p>
    <div id="content">
        <p>В ближайшие дни на стадию бета-тестирования перейдёт
        ветка Firefox 16 и будет отделена aurora-ветка Firefox 17.</p>
    </div>
    <p>В соответствии с шестинедельным циклом разработки, релиз Firefox 16 намечен
    на 9 октября, а Firefox 17 на 20 ноября.</p>
</body>
</html>
```

Представленный пример также позволяет легко понять различие во влиянии на нормальный поток абсолютного и относительного позиционирования. Замените здесь position на absolute. Часто применяется комбинирование абсолютного и относительного позиционирования. Следующий пример иллюстрирует относительное позиционирование внутри блока с абсолютным позиционированием. Обратите внимание на то, каким образом браузер отображает внешний блок при изменении размера окна.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<style type="text/css">
    #main {
        position: absolute;
        left: 100px;
        top: 50px;
        border: 1px solid black;
        padding: 0 100px 100px 0
    }
    #content {
```

```

        position: relative;
        left: 20px;
        top: 10px;
        border: 1px solid green;
    }
</style>
</head>
<body>
<div id="main">Главное меню:
  <div id="content"><p>Некоторый текст для проверки размещения
элемента.</p></div>
  </div>
</body>
</html>

```

Фиксированное позиционирование

В отличие от абсолютного позиционирования, фиксированное позволяет закрепить элемент относительно окна просмотра, а не элемента-контейнера. Это позволяет разместить элементы, которые не будут подвергаться прокрутке, например постоянно отображаемый блок меню.

Плавающие элементы

Плавающее размещение не является схемой позиционирования. Оно было введено как средство, позволяющее получить обтекание элементов, но не для создания макета страницы. Например следующие: стиль обеспечит отображение изображений в правой части страницы, а все остальные элементы будут размещать в свободном пространстве слева.

```
img { float: right; padding: 15px; }
```

Плавающее размещение иногда применяют к блокам, содержащим меню и прочие средства навигации.

В современной вёрстке, применение float крайне нерекомендовано. Вёрстка на float плохо адаптируется под разные экраны.

Управление отображением элемента

Для управления отображением элемента используется свойство display.

none Элемент не будет отображен

block Элемент отображается как блок (например как <div> или как <p>). Блок имеет отступы над и под собой, а также отделяется от следующих за ним HTML элементов.

inline Режим по умолчанию. Встроенный элемент отображается как часть строки (подобно span) и не разрывает строку перед и после себя. Отделяется от следующих за ним элементов.

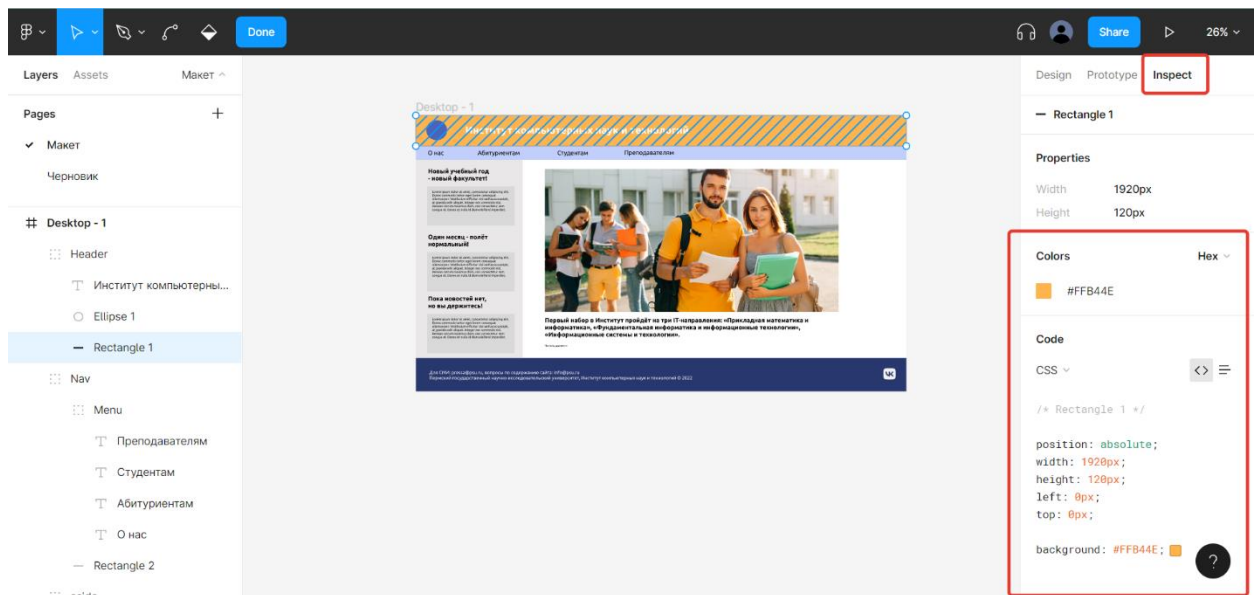
inline-block Элемент встраивается в строку, но ведёт себя как блок.

inherit Значение будет унаследовано у родительского элемента

Именно с помощью этих свойств нужно стремиться выстраивать макет сайта, так как это наиболее современный подход.

Figma и CSS

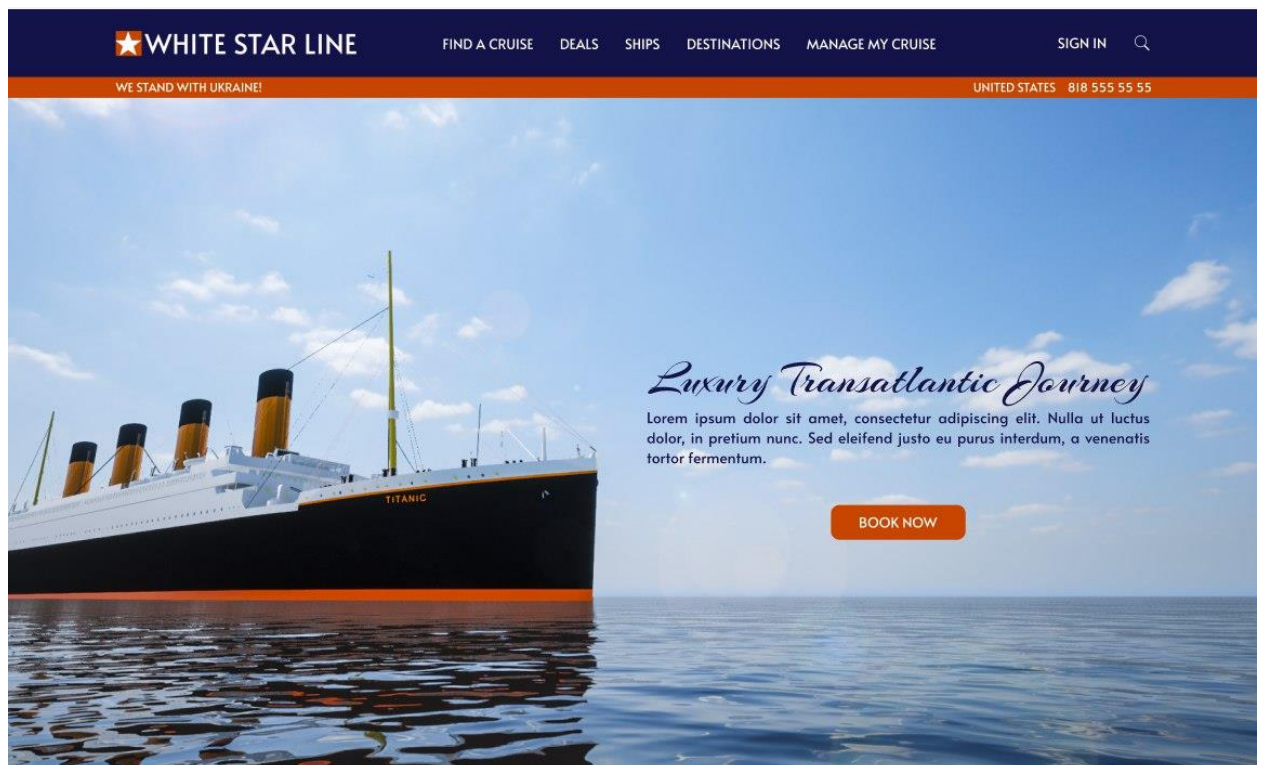
Графический редактор Figma позволяет генерировать базовые стили CSS непосредственно в макете. Для того, чтобы увидеть стиль элемента необходимо выбрать этот элемент с помощью курсора и в правом меню переключиться на вкладку "Inspect":



В это разделе Вы увидите размеры выделенного объекта, а также цвет в удобном формате, для размещения в CSS документе.

Мастер-класс по вёрстке статической страницы.

Возьмите макет страницы из открытых источников и попробуйте вместе со студентами разобраться, как верстать такую страницу.



[https://www.figma.com/file/nPdJk4jpHTRuwAynef52mJ/Titanic-Website-\(Community\)?node-id=0%3A1](https://www.figma.com/file/nPdJk4jpHTRuwAynef52mJ/Titanic-Website-(Community)?node-id=0%3A1)