

Практическая работа 5. Верстка по макету Figma

Тема:

Верстка статического сайта по макету Figma.

Цель:

Научиться преобразовывать макет из Figma в валидный, семантический HTML и CSS-код. Закрепить навыки работы с **Flexbox/Grid**, позиционированием, работой с изображениями и шрифтами. Освоить методологию проверки качества верстки с помощью специализированных сервисов.

Теоретическая часть:

- Figma для разработчика:** Figma — это инструмент для проектирования интерфейсов. Разработчики используют режим Inspect (правая панель) для получения точных значений отступов, размеров, цветов, стилей шрифтов и CSS-кода.
- Семантическая верстка:** Использование правильных HTML-тегов (`<header>`, `<main>`, `<section>`, `<article>`, `<footer>`) для улучшения доступности, SEO и читаемости кода.
- Методологии верстки:**
 - Flexbox:** Одномерная модель layout для распределения пространства вдоль одной оси (горизонтально или вертикально). Идеально подходит для компонентов, навигации.
 - CSS Grid:** Двумерная система для создания сложных сеточных макетов. Идеально подходит для общих макетов страницы.
- Позиционирование:** Понимание различий между `static`, `relative`, `absolute`, `fixed` и `sticky`.
- Работа с шрифтами:** Подключение веб-шрифтов (через Google Fonts или прямо из Figma), настройка `font-weight`, `line-height`, `letter-spacing`.
- Адаптивность:** Базовая адаптивность достигается за счет относительных единиц (`%`, `rem`, `vw`), медиа-запросов (`@media`) и гибких сеток.

Ход работы:

- Получите макет:** Преподаватель предоставляет [ссылку](#) на макет в Figma.
- Шаги
 - В файле `index.html` изменить атрибут `lang` у тега `html` с `en` на `ru`, изменить `title` на `Taskana App`.
 - В `assets` положить общие стили приложения, обязательно выделить `global.css` и `variables.css`, а также `normalize.css` или `reset.css` при использовании.
 - В `variables.css` в `:root` с помощью css переменных добавить цветовую палитру в соответствии с макетом, даже если они пока не используются.
 - Подключить шрифты локально, разместив их в `assets` в папке `fonts`. Допустимый формат шрифтов — `woff2`.

- Ссылку на скачивание шрифтов смотри в **макете** во вкладке компоненты. Для конвертации формата шрифта используй любой доступный сервис.
- Общие стили разместить в папке **assets**. Шрифты импортировать в файле **global.css**
- Изображения и иконки достаточно хранить в папке **public** или **assets**.

Согласно макету:

- Реализовать компонент **Header** в соответствии с макетом, он должен включать в себя дополнительные компоненты **Logo** и **Button**.
- Добавить необходимые компоненты в **AppLayout**. Он должен занимать 100% доступной высоты и ширины, **Header** всегда прижат к верхней части. Вертикальный скролл появляется при высоте экрана меньше чем контент внутри одной из секций: **NavBar**, **SideBar** или **TaskList**. Все скроллбары приложения дефолтные (без дополнительных кастомизаций) и не зависят друг от друга, т.е. возможен вариант когда в одной секции есть вертикальный скролл, а в других нет.
- Минимальная ширина экрана **1280px**, т.е. при меньшей ширине экрана появляется горизонтальный скролл.
- При ширине экрана больше чем **1280px** контент страницы растягивается на всю ширину, **NavBar** и правая часть страницы имеют фиксированную ширину. Центральная часть занимает все оставшееся пространство, её контент центрирован по горизонтали и имеет максимальную фиксированную ширину.
- При разработке приложения придерживаться **семантической** верстки. Для навигации использовать тег **nav**, для списков **ul**, для кнопок **button**, для страниц приложения **NavBar** использовать тег **a**.

Замечание: структурно в рамках текущей практической работы будем придерживаться компонентного подхода, предполагая, что каждый компонент представляет собой отдельный контейнер

2. Структура проекта:

```
my-figma-layout-project/  
├──  
├── index.html  
├── styles.css  
└── images/  
    ├── result-screenshot-1.png  
    └── result-screenshot-2.png
```

3. **Проанализируйте макет:** Внимательно изучите свой вариант в Figma. Обратите внимание на отступы, шрифты, цвета, размеры.
4. **Напишите код:** Сверстайте страницу согласно вашему варианту.
5. **Проверьте качество:** Загрузите свою готовую страницу (через Live Server) на один из сервисов для проверки:

- [Screenfly](#) (для проверки на разных разрешениях) Сделайте 2-3 скриншота с наложением макета для демонстрации точности.
 - 6. **Создайте репозиторий:** Инициализируйте локальный git-репозиторий, добавьте файлы, сделайте коммит.
 - 7. **Загрузите на Gogs:** Запустите репозиторий на ваш локальный сервер Gogs.
 - 8. **Заполните README.md:** Оформите файл [README.md](#) по предложенному шаблону.
-

Критерии оценки:

- **5 (Отлично):**

- Полное соответствие макету (проверено через сервис наложения).
- Чистый, семантический и валидный HTML-код.
- Использование современных подходов (Flexbox/Grid).
- Корректно подключены шрифты и изображения.
- Аккуратный и структурированный CSS-код.
- Репозиторий оформлен корректно, README.md содержит всю требуемую информацию.
- Работа залита на Gogs.

- **4 (Хорошо):**

- Есть незначительные отклонения от макета (1-3 мелких неточности).
- Код в целом семантический, но есть небольшие недочеты (например, избыток div).
- Верстка выполнена корректно, но могли быть использованы не самые оптимальные методы.
- Все файлы присутствуют, README.md заполнен, но, возможно, не так подробно.

- **3 (Удовлетворительно):**

- Есть заметные отклонения от макета.
- В коде присутствуют семантические ошибки.
- Верстка "ломается" при незначительном изменении размера экрана.
- Нарушена структура проекта или README.md заполнен не полностью.

- **2 (Неудовлетворительно):**

- Верстка кардинально не соответствует макету.
 - Грубые ошибки в HTML/CSS.
 - Работа не выполнена или сдана не в полном объеме.
-

Контрольные вопросы:

1. Какая панель в Figma является основной для разработчика и почему?
2. Объясните разницу между `display: flex` и `display: grid`. В каких случаях вы бы использовали каждый из них?
3. Что такое "семантическая верстка" и каковы ее преимущества?
4. Как правильно подключить пользовательский шрифт, который используется в макете?
5. Какие единицы измерения в CSS вы знаете и когда стоит использовать `px`, `%`, `rem`?

6. Какой CSS-свойство отвечает за плавное изменение элемента при наведении курсора?
7. Для чего нужен `box-sizing: border-box`?
8. Опишите процесс проверки верстки с помощью сервиса PixelParallel (или аналога).

Шаблон для файла `README.md` в репозитории студента

Студенты должны заполнить этот шаблон и положить его в корень своего проекта.

```
# Практическая работа № : Верстка по макету Figma

## Задание
Сверстать <a
href="https://www.figma.com/design/H778y2QQLV8iJEZDfIdzIv/TaskanaApp.-1-sprint.-1-task?node-id=218-4147&p=f&t=0qtjeUr8aZ3K3375-0">макет</a>макет Figma. Макет должен
быть `pixel-perfect`. Результат проверить с помощью сервиса наложения макета.

## Код проекта

### HTML (index.html)
```html
<!DOCTYPE html>
<html lang="ru">
<head>
 <!-- Ваш код -->
</head>
<body>
 <!-- Ваш код -->
</body>
</html>
```


### CSS (styles.css)

```
/* Ваши стили */
* {
 margin: 0;
 padding: 0;
 box-sizing: border-box;
}
/* ... */
```

## Результат

Ниже представлены скриншоты выполненной работы с наложением макета Figma для проверки точности.

 Скриншот результата 1 Пояснение к скриншоту 1 (например, "Общий вид секции")

 Скриншот результата 2 *Пояснение к скриншоту 2 (например, "Проверка отступов с наложением макета")*

