# Описание программы

Для работы с react необходимо установить node js. Через команду «npx create-react-app WebLesson» создастся новый проект со стандартной структурой create react app.

Вся разработка ведется в папке «src». Входной файл «index.js», запускает react и делает первичную отрисовку «app.js» (рис. 4.1).

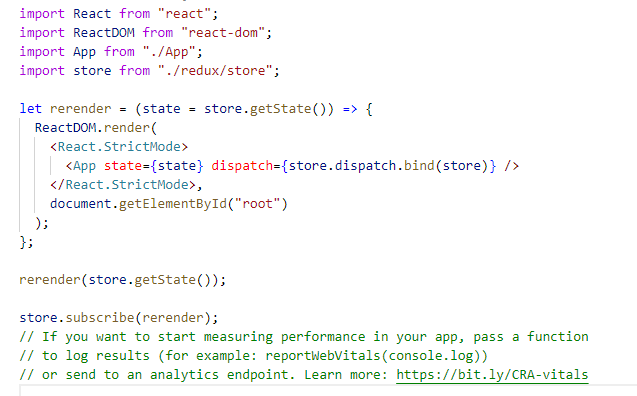


Рис. 4.1 Содержимое файла «index.js»

К файлу «app.js» можно подключить css файл, в котором будут прописаны стили отображения всех элементов приложения. «app.js» - это большая компонента которая содержит в себе несколько других компонент (рис. 4.2).

Все компоненты кроме app будут иметь расширение «jsx» вместо обычного «js». Это позволит удобно писать jsx разметку в среде разработки visual studio code. IDE будет включать синтаксис html. Все компоненты находятся в папке «moduls»

Глобально приложение можно разделить на 2 части: меню и контент. Соответственно можно создать 2 компоненты «menu.jsx» и «content.jsx».



Рис. 4.2 Содержимое компоненты «app.js»

Все компоненты кроме app будут иметь расширение «jsx» вместо обычного «js». Это позволит удобно писать jsx разметку в среде разработки visual studio code. IDE будет понимать когда включать синтаксис html а когда обычный js. Все такие компоненты будут находиться в папке «moduls»

Глобально приложение можно разделить на 2 части: меню и контент. Созданы 2 компоненты «menu.jsx» и «content.jsx».

Компонент «menu.jsx» можно поделить на меню с разделами «menuTheme.jsx» и главами по разделам «menuStep.jsx». их содержимое генерируется на ходу, в зависимости от state.

Для того чтобы добавить возможность свободно расширять содержимое программы, добавляя новые разделы и подразделы, необходимо вывести все данные о них в отдельный файл. Этот файл называется «theme1.json» и находится в папке public, для того чтобы удобно было к нему добраться после работы webpack в конечном electron приложении.

Библиотека react предназначена для отрисовки страницы, за ее наполнение отвечает сторона бизнес логики (BLL). Суть бизнес логики в том, чтобы взять данные из файла «theme1.json» и перенести в state, отвечающий за состояние страницы. В react существует внутренний локальный state, но есть проблема с их использованием при работе с функциональными компонентами. Для более удобной работы со state используется библиотека Redux.

Все дополнительные инструменты скачиваются и интегрируются в проект с помощью пакетного менеджера NPM.

Необходимо получить данные из «theme1.json» и поместить их в компоненты «menu» и «content». Так как компонента – это функция которая возвращает jsx разметку. Туда можно передать дополнительные параметры props. В props можно передать текущее состояние компоненты state.

В дальнейшем программа при взаимодействии с пользователем должна будет менять свой внешний вид. React должен перерисовать изменившиеся компоненты с обновленным state. При взаимодействии с элементом меняется его state. State меняется с помощью dispatch.

В компонент «menu» параметром приходят state со всеми данными. Задача данной компоненты отрисовать элементы на экране (рис. 4.3).

Слева можно увидеть разделы, а справа поле для поиска, шторку для изменения параметров шрифта, кнопку для скрытия меню, переключатель для изменения цветовой темы и внизу подразделы выбранного раздела.

За отрисовку разделов отвечает компонента «MenuTheme». За отрисовку подразделов отвечает компонента «menuStep».

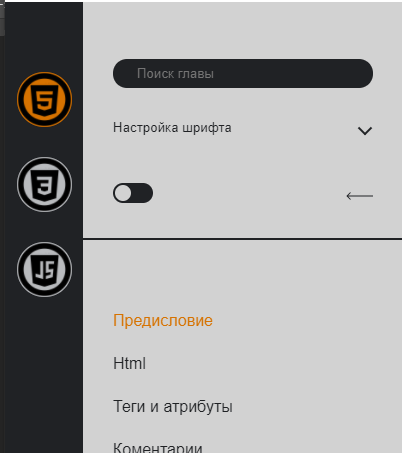


Рис. 4.3 Итог отрисовки компоненты «menu.jsx»

Элементы input и кнопки, запускает dispatch, который принимает в параметры объект с обязательным свойством «type». Благодаря этому свойству запускается нужный сценарии для изменения state. Все сценарии содержаться в reducers. Программе 2 Reducers.

Пример работы метода dispatch можно рассмотреть на кнопке скрытия меню. По нажатию на данную стрелочку выполняется функция (рис. 4.4).

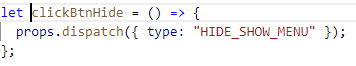


Рис. 4.4 Функция кнопки скрытия меню

Тем временем, в reducer меню «menuReducer.js» выполняется следующий сценарий (рис. 4.5).

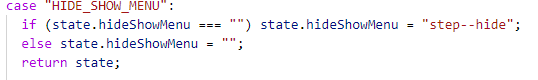


Рис. 4.5 Пример изменения state

По такому же принципу работают все интерактивные элементы.

«menuTheme.jsx» как параметр принимает название раздела и возвращает ее разметку. С помощью цикла map программа проходиться по массиву всех разделов и при каждой итерации вызывает данный компонент (рис. 4.6).



Рис. 4.6 Вызов «menuTheme.jsx»

«menuStep.jsx» похожим образом принимает параметр название подраздела и возвращает ее разметку (рис. 4.7).



Рис. 4.7 Вызов «menuStep.jsx»

Content генерирует свое наполнение благодаря функции «drowThemeContent» (рис. 4.8).

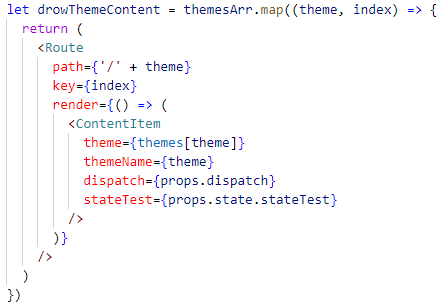


Рис. 4.8 Функция «drowThemeContent»

Создается новый route. Система route имитирует переход на другую страницу. React рисует интерфейсы не перезагружая страницу, в одном html файле, поэтому программа используется route для логичного удобного разделения контента.

Задача route поменять содержимое страницы в зависимости от URL. С помощью navLink’ов можно меняется URL.

На (рис. 4.8) видно, что вызывается компонента «ContentItem», которая принимает в качестве параметра данные выбранного раздела и возвращает готовую разметки целой страницы.

«ContentItem» пробегает по содержимому раздела, разбивая ее на подразделы и после разбивает их на элементы абзаца, определения, картинки или видео youtube. Делает она это с помощью функции «contentParse» (рис. 4.9).

В файле «theme1.json» данные хранятся в виде текста. Чтобы конвертировать элементы из строк в html разметку, необходимо узнать конечный элемент. Для этого используется функция «stepItemsParse» (рис. 4.10).

После определения типа данных, отрисовывает его.

Если элемент является картинкой, то запускается сценарий функции «imgParse» (рис. 4.11).

Если элемент является ссылкой, то запускается сценарий функции «linkParce» (рис. 4.12).

Если элемент является определением, то запускается сценарий функции «specialTextParse» (рис. 4.13).

Если элемент является ссылкой на видео youtube, то запускается сценарий функции «youtubeVideoParse» (рис. 4.14).



Рис. 4.9 Функция «contentParse»



Рис. 4.10 Функция «stepItemsParse»

Если просто текст, то возвращается соответствующая разметка без использования дополнительной функции.

В результате можно увидеть контент (рис. 4.15).

Программа так же поддерживает тесты. Чтобы их добавить, нужно создать объект в файле с данными (рис. 4.16).

Если в разделе находиться такой объект, то при его отрисовки создается определенная система route. Она позволит отбразить вопросы при нажатии на кнопку (рис. 4.17).

В конечном итоге программа выглядит так (рис. 4.18).

Так программа выглядит с темной темой и скрытым меню (рис. 4.19).

При наведении на тему будет отображаться ее название (рис. 4.21).



Рис. 4.11 Функция «imgParse»



Рис. 4.12 Функция «linkParce»

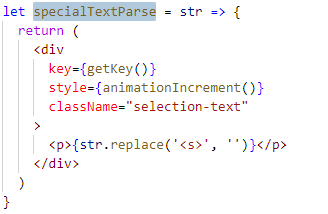


Рис. 4.13 Функция «specialTextParse»



Рис. 4.14 Функция «youtubeVideoParse»

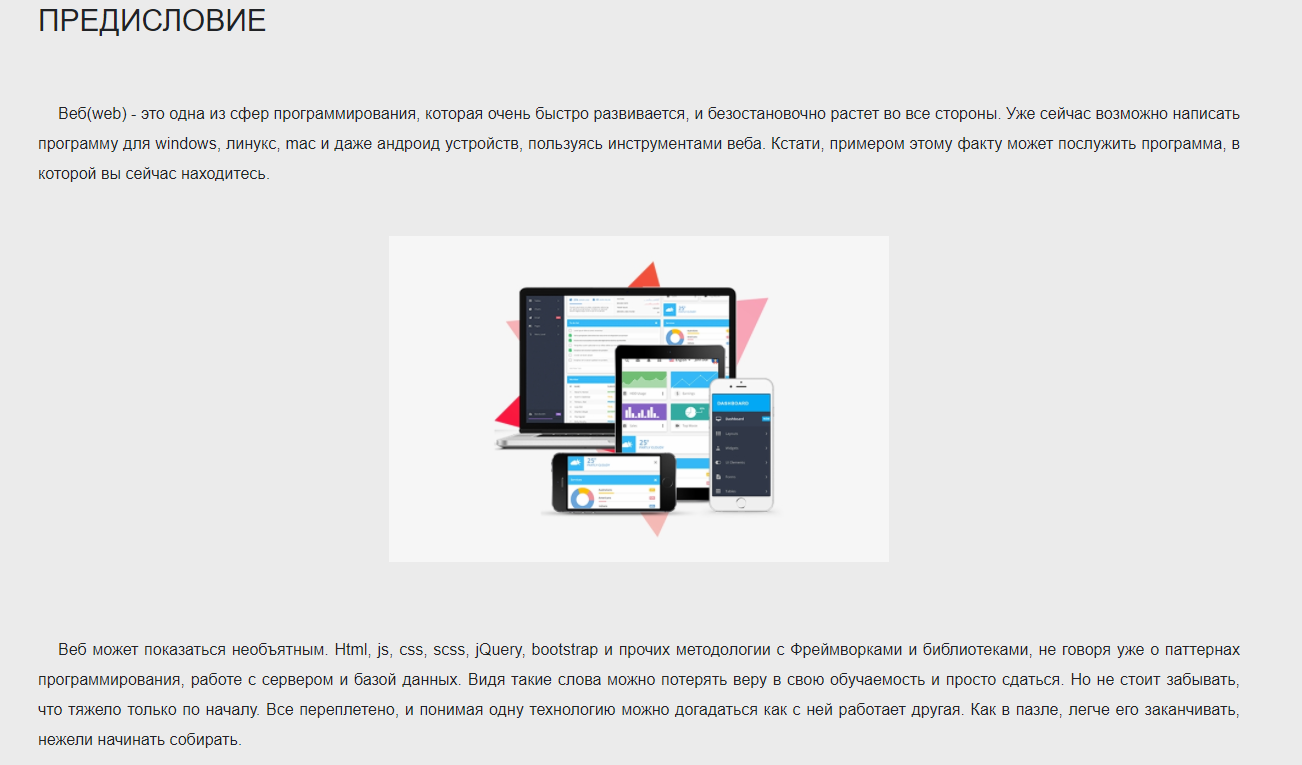


Рис. 4.15 Результат парсинга

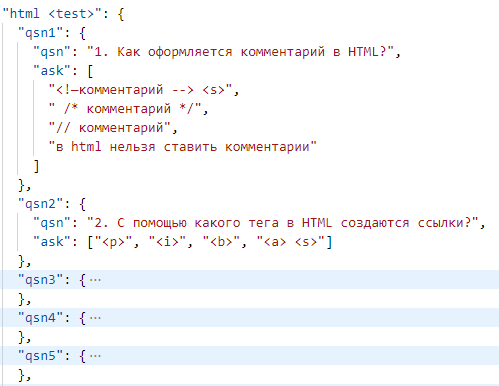


Рис. 4.16 Заполнение теста



Рис. 4.17 Кнопка теста

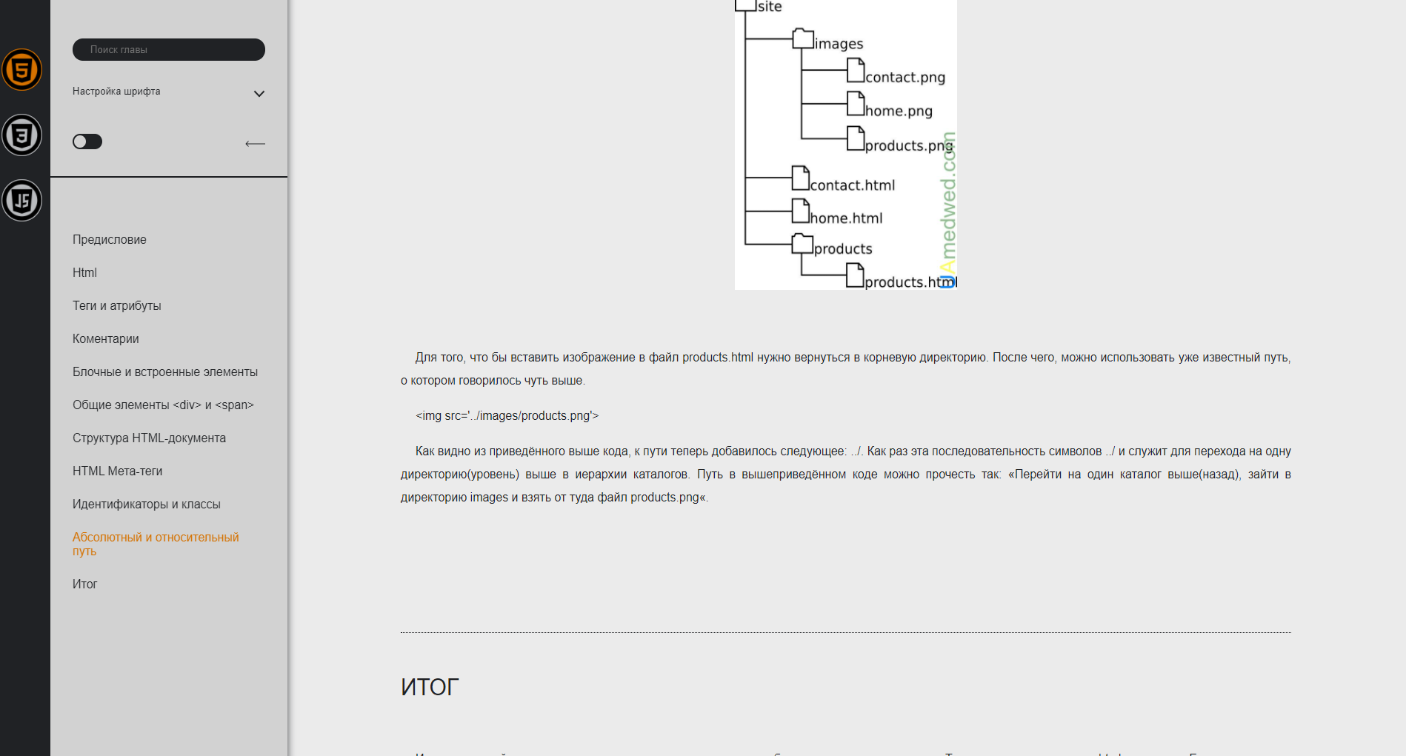


Рис. 4.18 Как выглядит программа в результате



Рис. 4.21 Наведение на тему

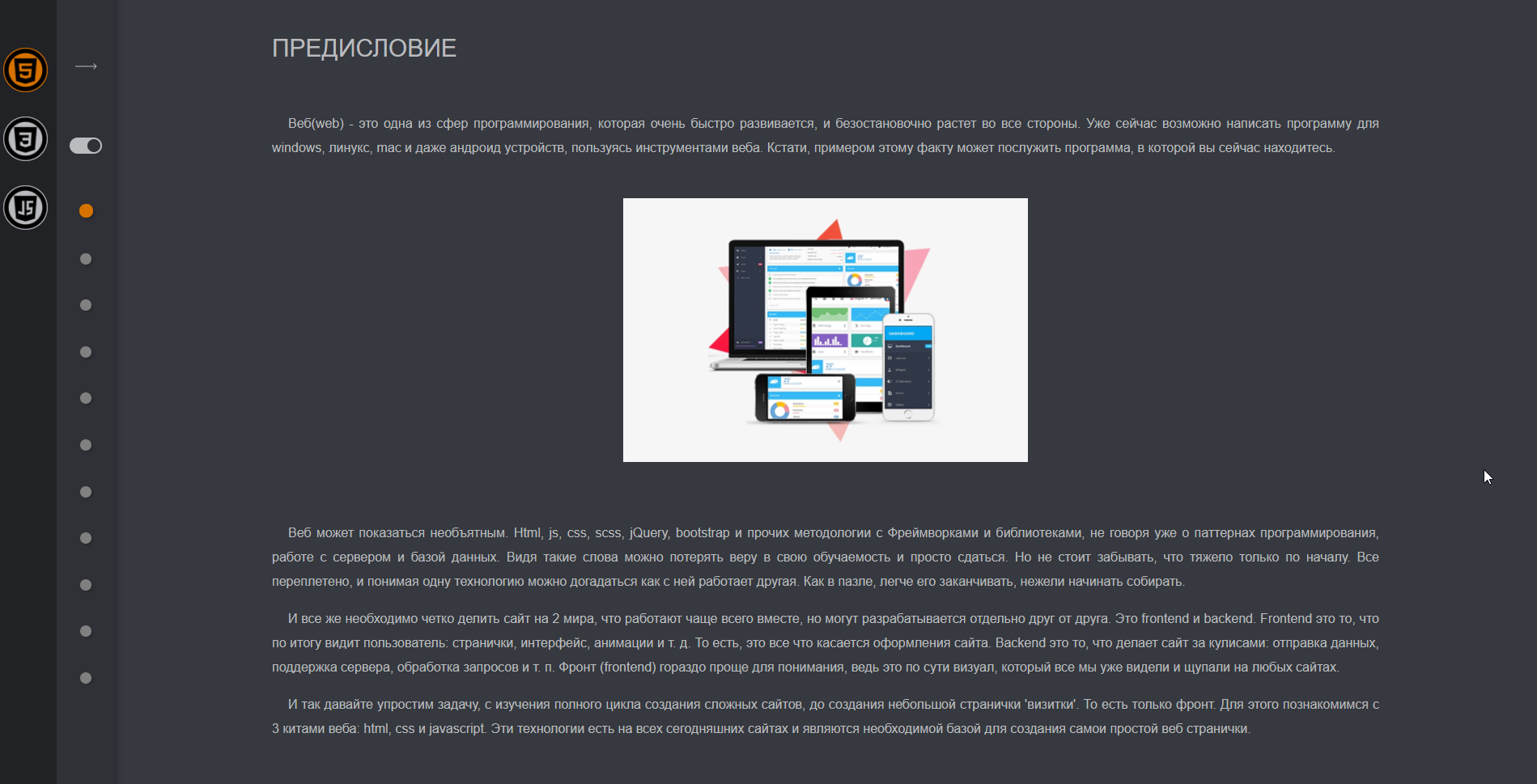


Рис. 4.19 Темная тема

При нажатии на картинку она развернется на полный экран, для более удобного просмотра (рис. 4.20).

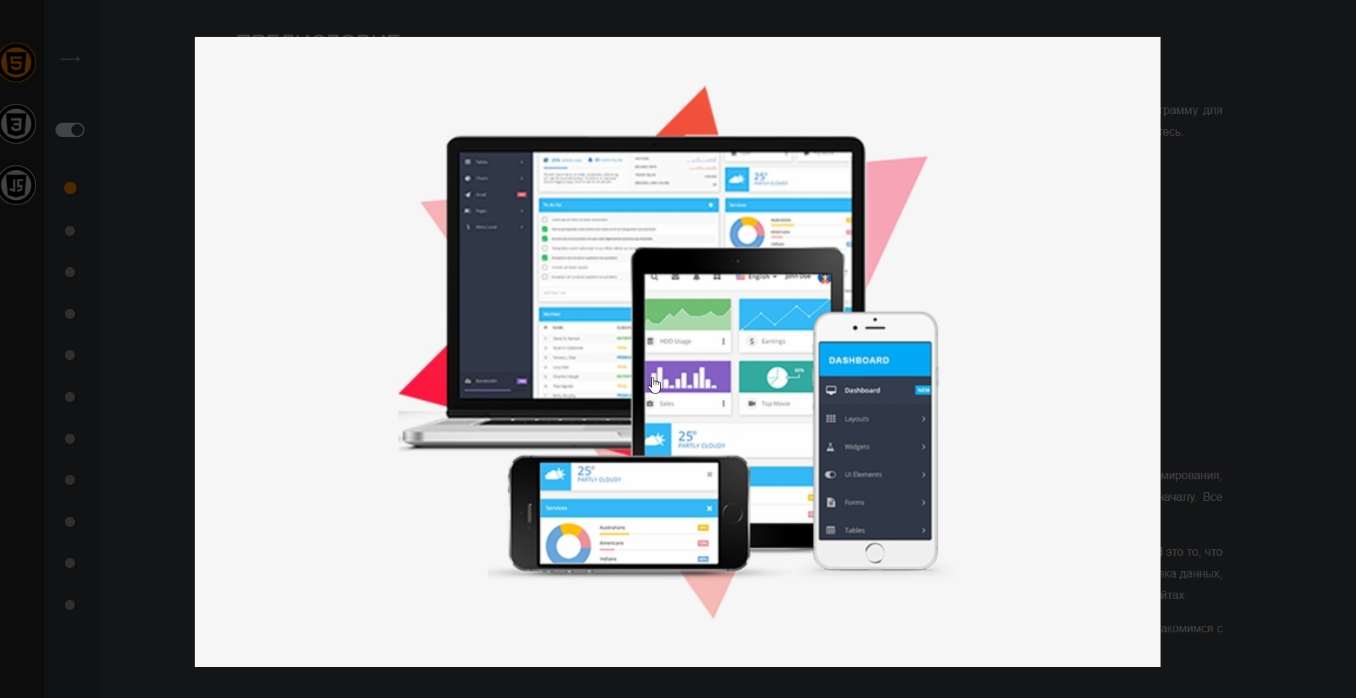


Рис. 4.20 Расширенная фотография

Так же в программе реализованы подсказки. При наведении на любой интерактивный объект и по истечении 1 секунды появиться подсказка о том, что этот элемент делает (рис. 4.22).



Рис. 4.22 Подсказка

Страница с тестами выглядит так (рис. 4.23).

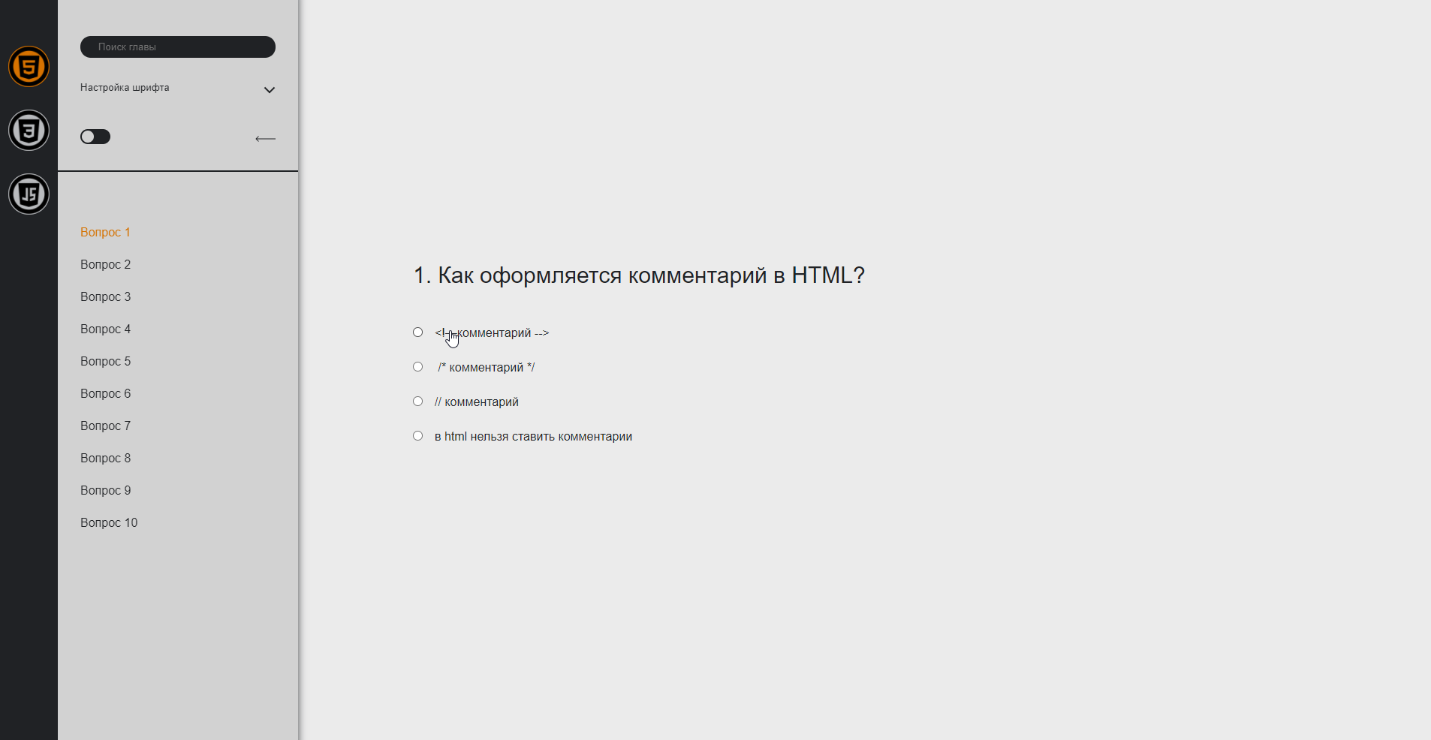


Рис. 4.23 Страница с вопросами