CyberBionic systematics

FRONTEND DEVELOPER

Практика

Содержание:

Введение 3

Темы для курсового проекта. Курс HTML5 & CSS3 4

Личное резюме 4

Темы для курсового проекта. Курс JavaScript Essential 5

Задача 1 5

Задача 2 5

Задача 3 5

Темы для курсового проекта. Курс JavaScript Advanced 7

Прогноз погоды 7

Темы для курсовых проектов. Курс HTML5 & CSS3 Advanced 9

Сайт пиццерии 9

Темы для курсовых проектов. Курсы Angular/TypeScript 10

Интернет магазин 10

Введение

В этом документе собраны рекомендации и варианты курсовых проектов. Вы можете по желанию выбрать свою тему для курсового проекта согласовав его с тренером.

Вам нужно выбрать и защитить как минимум один проект.

Мы рекомендуем сделать максимальное возможное количество проектов, но реальное количество курсовых проектов будет зависеть от Вашей успеваемости или от количества, времени которое Вы можете выделить на работу над проектами.

Для хранения исходного кода используйте репозиторий на https://github.com

Помните, что для решения многих задач, которые описаны в этом документе и с которыми вы столкнетесь во время работы, вам будут необходимы дополнительные навыки инструменты, изучение применение которых выходит за рамки учебной программы. Самостоятельное изучение и применение техник, описанных в дополнительных материалах к задачам, будут большим плюсом.

Темы для курсового проекта. Курс HTML5 & CSS3

Личное резюме

Цель: Продемонстрировать навыки верстки, применить адаптивную верстку.
Обязательные языки/технологии/библиотеки: HTML, CSS
Дополнительные языки/технологии/библиотеки: SASS, Bootstrap
Дополнительные материалы:
Верстаем сайт правильно
Работа с GIT
Верстка сайта за 30 минут на Flexbox

Требования к готовому решению:

- Исходный код должен быть размещен на github или другом подобном сервисе

- Стили должны быть вынесены в отдельный файл изображения с осмысленными именами должны храниться в отдельной папке рядом с html разметкой.

- Ваша страница должна корректно отображаться на разрешениях 1280 и более. Как усложнение задания реализуйте адаптивную верстку (если выполняете это задание, после освоения этой темы или самостоятельно пытаетесь освоить медиа запросы)
  Основные разрешения:
  - 320 px. – мобильное устройство (портретная ориентация);
  - 480 px – мобильное устройство (альбомная ориентация);
  - 600 px – Небольшие планшеты;
  - 768 px – Планшеты (портретная ориентация);
  - 1024 px – Планшеты (альбомная ориентация);
  - 1280 px и более – PC.

- В резюме необходимо ввести ваши данные.

  Обязательные разделы резюме:
  - Фото и ФИО кандидата
  - Цель
  - Блок с контактными данными
  - Блок с опытом работы - в обратном хронологическом порядке места работы, даты работы, должность, описание обязанностей и т.д.
  - Блок с образованием
  - Блок с сертификатами – с изображениями сертификатов их номерами, ссылками по возможности.
  - Блок с хобби и личными качествами
  - Другие блоки по желанию

- Можно выбрать свой стиль шаблона, но желательно выполнить верстку по макету, который находится в https://drive.google.com/drive/folders/1IdpTV4grpW_TPPlrq5NguY90nEDMpGH_?usp=sharing

Темы для курсового проекта. Курс JavaScript Essential

Цель: Продемонстрировать навыки использования базовых конструкция JavaScript
Обязательные языки: JavaScript

Так как задачи курса JavaScript Essential изучение синтаксиса языка и не включают полноценного взаимодействия с API браузера, задания в этом разделе не являются полноценными курсовыми проектами, а дополнительными задачами для самостоятельной работы. Перед выполнением этих заданий решите все домашние задания, входящие в курс JavaScript Essential

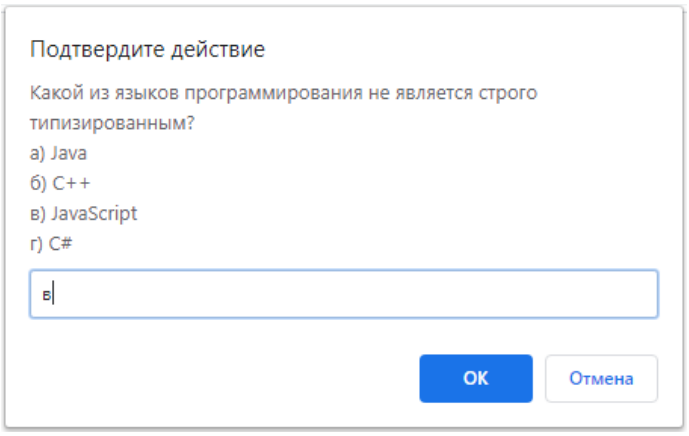
Задача 1
Создайте функцию, которая будет определять является ли введенная пользователем строка палиндромом.
Палиндром - число, буквосочетание, слово или текст слева направо и справа налево читается одинаково. Например, 707, топот, оно, мадам и т.д.

Задача 2
Напишите функцию, которая сравнивает две строки и определяет являются ли они анаграммой (слова состоят из одних и тех же букв). Строки получаем от пользователя с помощью 2-х методов prompt.
Пример:
Первое слово - апельсин
Второе слово - спаниель
Результат - слова "апельсин" и "спаниель" являются анаграммой.

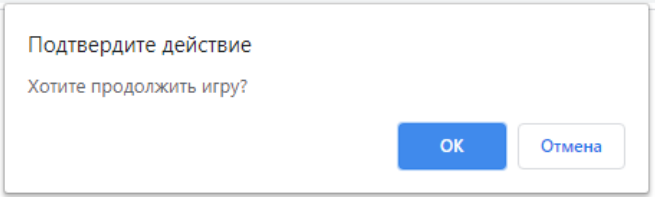
Задача 3
Игра «Как стать миллионером».
Подумайте, как можно реализовать подобный сценарий с использованием массивов и объектов. Помните, что для каждого вопроса нужно прописать: вопрос, 4 варианта ответа, правильный ответ (который будем сравнивать с тем, что ввел пользователь, сумма которую выиграл пользователь). Определите минимум 5 вопросов. Вопросы с вариантами ответов вы можете придумать самостоятельно, а можете воспользоваться поиском

Как должна выглядеть реализация:

1) В окне prompt выводятся вопрос и варианты ответов.



2) Если пользователь вводит правильный ответ (в виде буквы), то выводится окно confirm с вопросом желает ли пользователь продолжить.



3) Если "ОК" - продолжаем, и пользователю выводится следующий вопрос иначе - выводим сколько выиграл пользователь и выходим из игры
4) Если пользователь вводит неверный ответ - выходим пользователю информацию, что он указал не правильный ответ и выходим из игры.

Дополнительное задание
Увеличьте количество вопросов до 15 и добавьте несгораемые суммы на 5 и 10 вопросах ($5000 и $25000 соответственно). Это означает, что если пользователь ответил, например, на 7 вопросов и «заработал» $10000, а на 8-й ответил не верно, то гарантировано получит 5000.

Номер вопроса	Приз
1	500
2	1 000
3	2 000
4	3 000
5	**5 000**
6	8 000
7	10 000
8	13 000
9	15 000
10	**25 000**
11	50 000
12	100 000
13	250 000
14	500 000
15	**1 000 000**

Темы для курсового проекта. Курс JavaScript Advanced

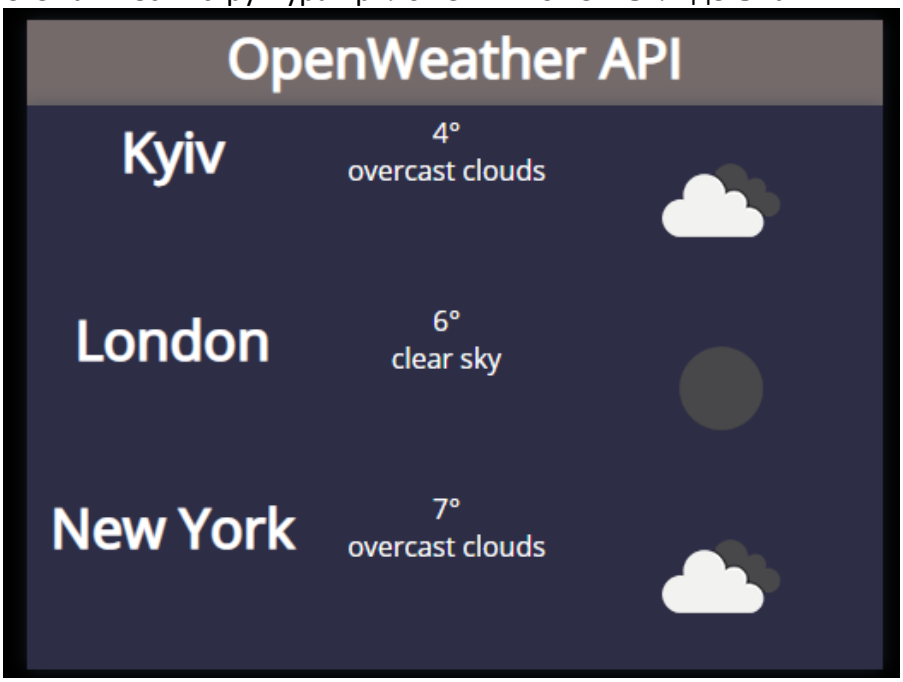
Прогноз погоды

Цель: Применить ООП и другие техники на JavaScript.
Обязательные языки/технологии/библиотеки: HTML, CSS, JavaScript
Дополнительные языки/технологии/библиотеки: SASS, Bootstrap, jQuery, TypeScript
Дополнительные материалы:
JavaScript шаблоны
ECMAScript 6 - новые инструменты для JavaScript разработчика
Асинхронное программирование в JavaScript

Требования к готовому решению:

- Приложение должно показывать текущую погоду минимум по трем городам: Киев, Лондон, Нью Йорк
- Оформление приложения произвольное, акцент нужно сделать на структуру JavaScript кода. Разделите логику работы с API и интерфейсом по отдельным объектам. Не пытайтесь разместить весь код в одном файле, разбейте логику на несколько файлов.
  Преимуществом будет использование шаблона модуль. Подробнее о этом шаблоне вы можете узнать в этом видео уроке
  https://itvdn.com/ru/video/javascript-patterns/create-patterns

- Схематически структура приложения может выглядеть так



- Приложение должно отображать погоду выполняя запросы к погодным сервисам. Подробная документация находится на сайте https://openweathermap.org/

- Ссылки для получения данных
  Киев
  http://api.openweathermap.org/data/2.5/weather?id=703448&appid=bf35cac91880cb98375230fb443a116f

  Лондон

  http://api.openweathermap.org/data/2.5/weather?id=2643743&appid=bf35cac91880cb98375230fb443a116f

  Нью Йорк
  http://api.openweathermap.org/data/2.5/weather?id=5128638&appid=bf35cac91880cb98375230fb443a116f

- Отображайте соответствующую иконку для погоды, список вы можете найти в документации https://openweathermap.org/weather-conditions

  Для размещения изображения используйте обращение по специальному адресу:
  `http://openweathermap.org/img/wn/'+data.weather[0]['icon']+'@2x.png`
  где data – объект, который был возвращен с сервера с информацией о погоде. Можете воспользоваться другой логикой и подобрать свои изображения для погоды.

- Добавьте возможность переключать настройку вывода температуры в градусах Цельсия или по Фаренгейту.

- Дополнительные задачи:

  - Добавьте вывод дополнительных данных о погоде – направление ветра, скорость ветра, давление, время восхода и захода солнца
  - Добавьте возможность просматривать прогноз погоды на следующие несколько дней в табличном представлении для выбранного города. Ознакомьтесь для этого с документацией https://openweathermap.org/. Предусмотрите соответствующие элементы пользовательского интерфейса, которые дадут возможность переключать отображение между текущей погодой и прогнозом погоды.

Темы для курсовых проектов. Курс HTML5 & CSS3 Advanced

Сайт пиццерии

Цель: Продемонстрировать навыки верстки по макету, применить адаптивную верстку, закрепить на практике темы курсов по HTML и JavaScript
Обязательные языки/технологии/библиотеки: HTML, CSS, SASS
Дополнительные языки/технологии/библиотеки: Bootstrap
Дополнительные материалы:
Верстаем сайт правильно
Работа с GIT
Адаптивная верстка с помощью FlexBox и Grid
Верстка сайта за 30 минут на Flexbox
Верстка с использованием Gulp, JavaScript, HTML/CSS Часть 1
Верстка с использованием Gulp, JavaScript, HTML/CSS Часть 2
Верстка с использованием Gulp, JavaScript, HTML/CSS Часть 3

Требования к готовому решению:

- Исходный код должен быть размещен на github или другом подобном сервисе
- Сверстайте сайт пиццерии по макету в файле Pizza-Layout.zip. Все страницы должны быть максимально приближенными к макету.
- Продумайте структуру файлов и папок используйте правильные и понятные имена.
- Используйте адаптивную верстку
- Используйте шрифты https://fonts.google.com/
- Панель навигации (верхнее меню) должна быть закрепленной и двигаться вместе с сайтом
- На странице о нас разместите видео используя video элемент
- Сайт должен состоять из нескольких страниц – главная, меню, заказ, о нас. На каждой странице должен находится хедер с меню и футер как на главной странице. Контент каждой страницы свой и представлен в макете. Меню сайта должно выполнять переходы на соответствующие страницы.
- Форма на странице заказа должна проверять корректность введенных данных и показывать пользователю ошибки, если он их допустил. Отправлять куда либо данные с формы не нужно, в случае корректного ввода, можно просто показать сообщение о успешной отправке формы.
- Не забывайте о семантической верстке, на страницах должен быть один заголовок h1 и html элементы должны соответствовать содержимому, блоки текста – p, списки – ul или ol, статьи – article и т.д.
- Большим преимуществом будет использование SASS. Большим плюсом будет использование Gulp. Дополнительно о Gulp и верстке вы можете узнать в серии вебианров https://itvdn.com/ru/webinars/description/webinar-lending1

Темы для курсовых проектов. Курсы Angular/TypeScript или React

Интернет магазин

Цель: Продемонстрировать навыки создания SPA приложений, закрепление на практике фреймворков и библиотек, изученных на курсе
Обязательные языки/технологии/библиотеки: На выбор – Angular или React
Дополнительные языки/технологии/библиотеки: Gulp, Webpack, Angular CLI

Дополнительные материалы:
Angular 4 Jump Start
Создание первого проекта на Angular

Требования к заданию:

- Реализуйте приложение как SPA
- Приложение должно состоять из разделов – каталог, корзина, панель администрирования
- Интерфейс интернет магазина делайте на свое усмотрение, можете использовать стандартные стили bootstrap или material design. Основной акцент сделайте на описание бизнес логики и структуру проекта.
- Разделите бизнес логику и логику работы с пользовательским интерфейсом. Работа с внешним API должна быть вынесена в отдельные сервисы. Используйте внедрение зависимостей для Angular или импорты для React .
- Выделите модель/компонент для работы с данными. Как минимум у Вас должны быть классы product и cart, но просматривая документацию, возможно вы добавите другие классы, например, user.
- В качестве серверной логики используйте Fake Store API. Ссылка на документацию https://fakestoreapi.com/docs

- **Требования к каталогу**
  - Отображает список продуктов и их цены. Информация по запросу к серверу https://fakestoreapi.com/docs#p-all
  - Позволяет фильтровать каталог по категориям продукта. Выводить продукты по возрастанию и по убыванию цены.
  - Можно просмотреть детали каждого продукта – описание, категорию цену и т.д.
  - Из каталога и из страницы просмотра деталей о продукте можно добавить продукт в корзину
  - Реализуйте пагинацию в каталоге

- **Требования к корзине**
  - корзина должна содержать продукты, которые пользователь добавил из каталога
  - корзина должна содержать сумму всех продуктов, которые были добавлены
  - информация о количестве продуктов в корзине должна отображаться в правом верхнем углу страницы

- **Требования к панели администрирования**
  - этот раздел должен использоваться администратором, но будет доступен всем пользователям. Настройка авторизации пользователей не входит в задание этого проекта.
  - в разделе должны отображаться все корзины всех пользователей. Сумма каждой корзины по отдельности и сумма всех корзин. Документация https://fakestoreapi.com/docs#c-all
  - должна быть возможность просмотреть содержимое одной конкретной корзины (увидеть список всех продуктов, которые были куплены). Документация https://fakestoreapi.com/docs#c-single
  - должна быть возможность фильтровать вывод по датам. Администратор указывает даты с по и информация по корзинам отображается только для корзин, созданных в эти даты. Документация https://fakestoreapi.com/docs#c-date

Онлайн кинотеатр

Цель: Продемонстрировать навыки создания SPA приложений, закрепление на практике фреймворков и библиотек, изученных на курсе
Обязательные языки/технологии/библиотеки: На выбор – Angular или React
Дополнительные языки/технологии/библиотеки: Gulp, Webpack, Angular CLI

Дополнительные материалы:
Angular 4 Jump Start
Создание первого проекта на Angular

Требования к заданию:

- Реализуйте приложение как SPA
- Приложение должно состоять из разделов – каталог, корзина, панель администрирования
- Интерфейс онлайн кинотеатра делайте на свое усмотрение, можете использовать свои стили или стандартные стили bootstrap, или material design. Основной акцент сделайте на описание бизнес логики и структуру проекта.
- В качестве серверной логики используйте The Movie Database. Ссылка на документацию https://developers.themoviedb.org/3/getting-started/introduction

- **Требования к каталогу**
  - Отображает список фильмов и их краткое описание. Информация по запросу к серверу
  - Позволяет отфильтровать каталог по популярности. Выводить фильмы по дате релиза возрастанию и по убыванию.
  - Можно просмотреть детали фильма – описание, категорию и т.д.
  - Из каталога и из страницы просмотра деталей фильма можно добавить его в список желаемых к просмотру

- Реализуйте пагинацию в каталоге

- **Требования к панели администрирования**
  - этот раздел должен использоваться администратором, но будет доступен всем пользователям. Настройка авторизации пользователей не входит в задачи этого проекта.
  - в разделе должны отображаться все фильмы, которые находятся в списке желаемых к просмотру всех пользователей. Документация https://developers.themoviedb.org/3/getting-started/introduction
  - должна быть возможность просмотреть содержимое списка одного пользователя.
  - должна быть возможность фильтровать вывод по датам. Администратор указывает даты с по и информация по спискам отображается только для списков, созданных в эти даты.