Лабораторная работа №2
Асинхронность в javascript. Работа с API

***Цель работы*:** научиться получать данные от API и отображать их на web-странице.

Порядок выполнения

1 Создйте GitHub-репозиторий.

2 В процессе работы над проектом ведите историю коммитов. История коммитов должна отображать процесс разработки приложения. Следуйте гайдлайну. История должна содержать не менее трех коммитов.

3 Изучить основы работы с API-запросами и асинхронным JS:

– Using the Fetch API. Режим доступа: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch**;**

– Практическое ES6 руководство, о том, как сделать HTTP запрос с помощью Fetch API Режим доступа: https://jem-space.ru/praktichieskoie-es6-rukovodstvo-o-tom-kak-sdielat-http-zapros-s-pomoshchiu-fetch-api/

4 Изучить работу с json-данными:

– Working with JSON Режим доступа: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON ;

– Что такое JSON Режим доступа: https://habr.com/ru/post/554274/

5 Реализовать приложение. В лабораторной работе предложено четыре варианта приложений. Варианты заданий смотрите в google docs. По согласованию с преподавателем можно реализовать свой вариант приложения.

6 Проверить работу приложения для Google Chrome.

7 Проверить, не «рушатся» ли стили в других браузерах.

Требования к верстке

– вёрстка адаптивная. Приложение хорошо выглядит при ширине страницы от 1920px до 768px;

– интерактивность элементов, с которыми пользователи могут взаимодействовать, изменение внешнего вида самого элемента и состояния курсора при наведении, использование разных стилей для активного и неактивного состояния элемента, плавные анимации;

– в футере приложения есть ссылка на GitHub автора приложения, год создания приложения;

– представленные варианты страниц не для копирования, а как пример того, что можно сделать.

Технические требования

- работа приложения проверяется в браузере Google Chrome последней версии;
- можно использовать bootstrap, material design, css-фреймворки, html и css препроцессоры;
- не разрешается использовать jQuery, другие js-библиотеки и фреймворки;
- js-код приложения должен быть читаемым.

Варианты приложений

Вариант 1. Галерея изображений

Unsplash один из крупнейших поставщиков фотографий в Интернете. Unsplash API позволяет получать бесплатные фото, тематику которых можно указать. Другим известнейшим и популярнейшим API, позволяющим получить фото указанной тематики, является Flickr.

Задача

Разработать приложение отображающее полученные от API фото. Добавить приложению поиск. При вводе поискового запроса изменяются фото, которые отображаются в приложении.

Требования

- на странице есть несколько фото, и строка поиска. Каждое фото содержит снимок, описание и еще одно поле на ваше усмотрение;
- в футере приложения есть ссылка на GitHub автора приложения, год создания приложения;
- при загрузке приложения на странице отображаются полученные от API изображения;
- если в поле поиска ввести слово и отправить поисковый запрос, на странице отобразятся изображения соответствующей тематики, если такие данные предоставляет API;
- при открытии приложения курсор находится в поле ввода;
- есть placeholder;
- автозаполнение поля ввода отключено (нет выпадающего списка с предыдущими запросами);
- поисковый запрос можно отправить нажатием клавиши Enter;
- после отправки поискового запроса и отображения результатов поиска, поисковый запрос продолжает отображаться в поле ввода;

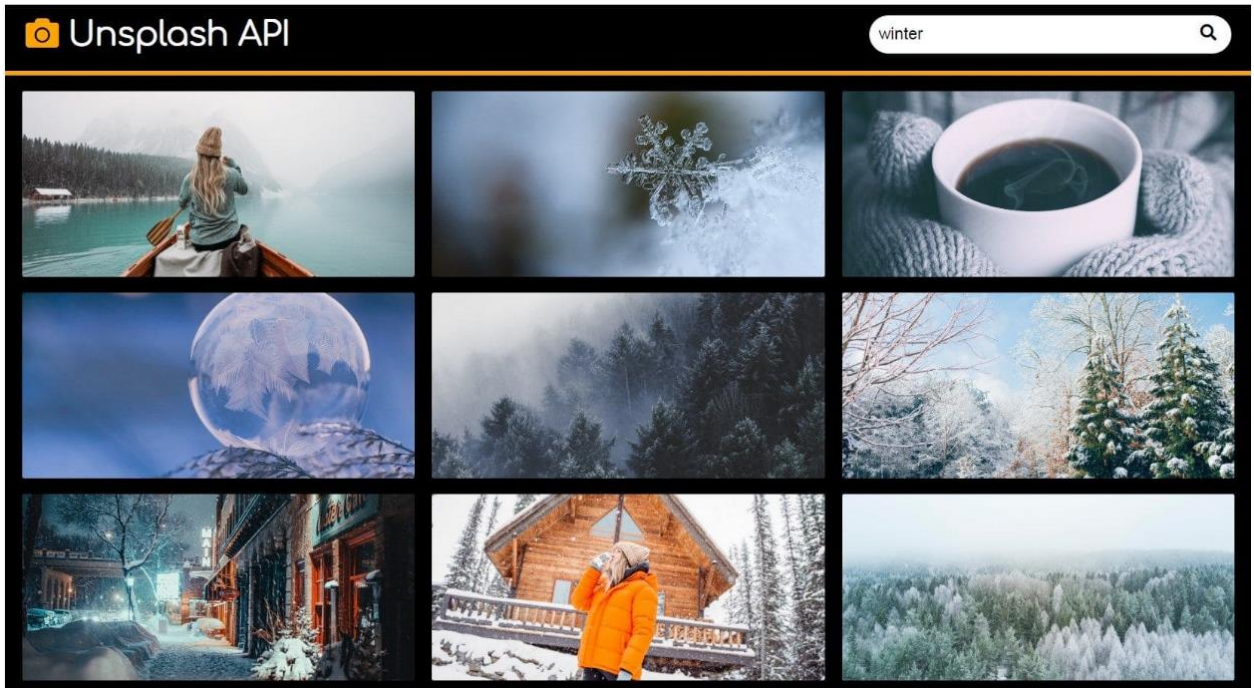
– в поле ввода есть крестик при клике по которому поисковый запрос из поля ввода удаляется и отображается placeholder.

Ссылки на API

- Unsplash API
  - сайт https://unsplash.com/developers
  - документация: https://unsplash.com/documentation
- Flickr API
  - сайт: https://www.flickr.com/services/
  - документация

https://www.flickr.com/services/api/flickr.photos.search.html

Пример



Вариант 2 Поисковик по видео

Существуют API, которые позволяют получить информацию о фильмах. Например. к ним относятся OMDb RESTful API, The Movie Database (TMDB), также при создании приложения можно использовать другие подходящие API.

Задача

Необходимо создать приложение, отображающее информацию о фильмах по запросу пользователя.

Требования

– на странице есть несколько карточек фильмов и строка поиска. На каждой карточке фильма есть постер и название фильма. Также на карточке может быть другая информация, которую предоставляет API, например, описание фильма, его рейтинг на IMDb и т.д.;

– при загрузке приложения на странице отображаются карточки фильмов с полученными от API данными;

– если в поле поиска ввести слово и отправить поисковый запрос, на странице отобразятся карточки фильмов, в названиях которых есть это слово, если такие данные предоставляет API;

– в футере приложения есть ссылка на GitHub автора приложения, год создания приложения;

– при открытии приложения курсор находится в поле ввода;

– есть placeholder;

– автозаполнение поля ввода отключено (нет выпадающего списка с предыдущими запросами);

– поисковый запрос можно отправить нажатием клавиши Enter;

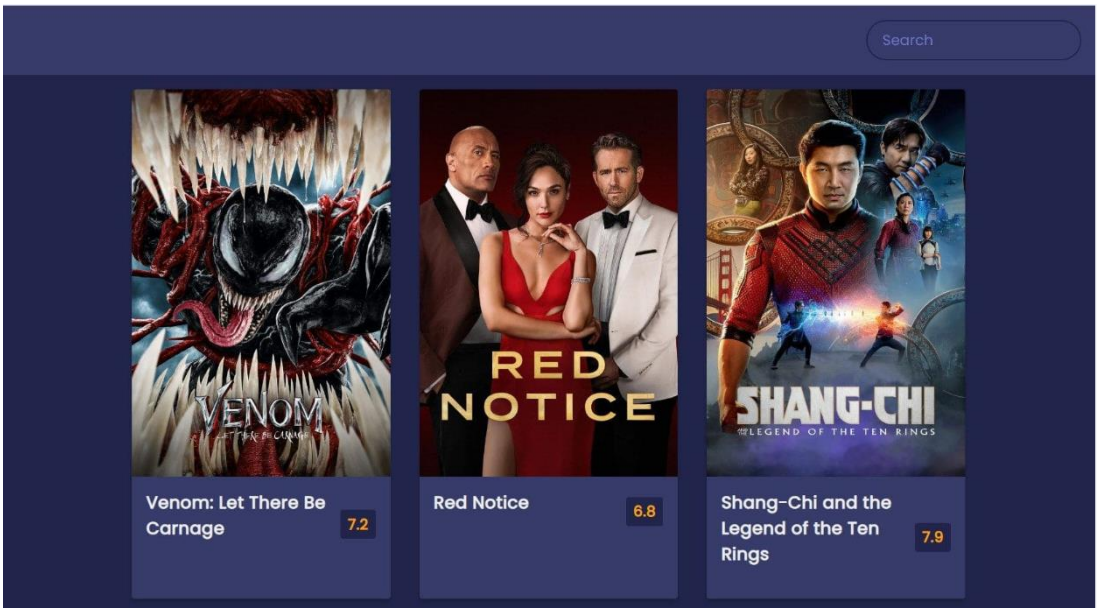
– после отправки поискового запроса и отображения результатов поиска, поисковый запрос продолжает отображаться в поле ввода;

– в поле ввода есть крестик при клике по которому поисковый запрос из поля ввода удаляется и отображается placeholder.

Ссылки на API

- OMDb API http://www.omdbapi.com/
- TMDB API https://www.themoviedb.org/documentation/api

Пример



Вариант 3. Поисковик книг

Существуют API, которые позволяют получить информацию о книгах.

Задача

Необходимо создать приложение, отображающее информацию о книгах по запросу пользователя.

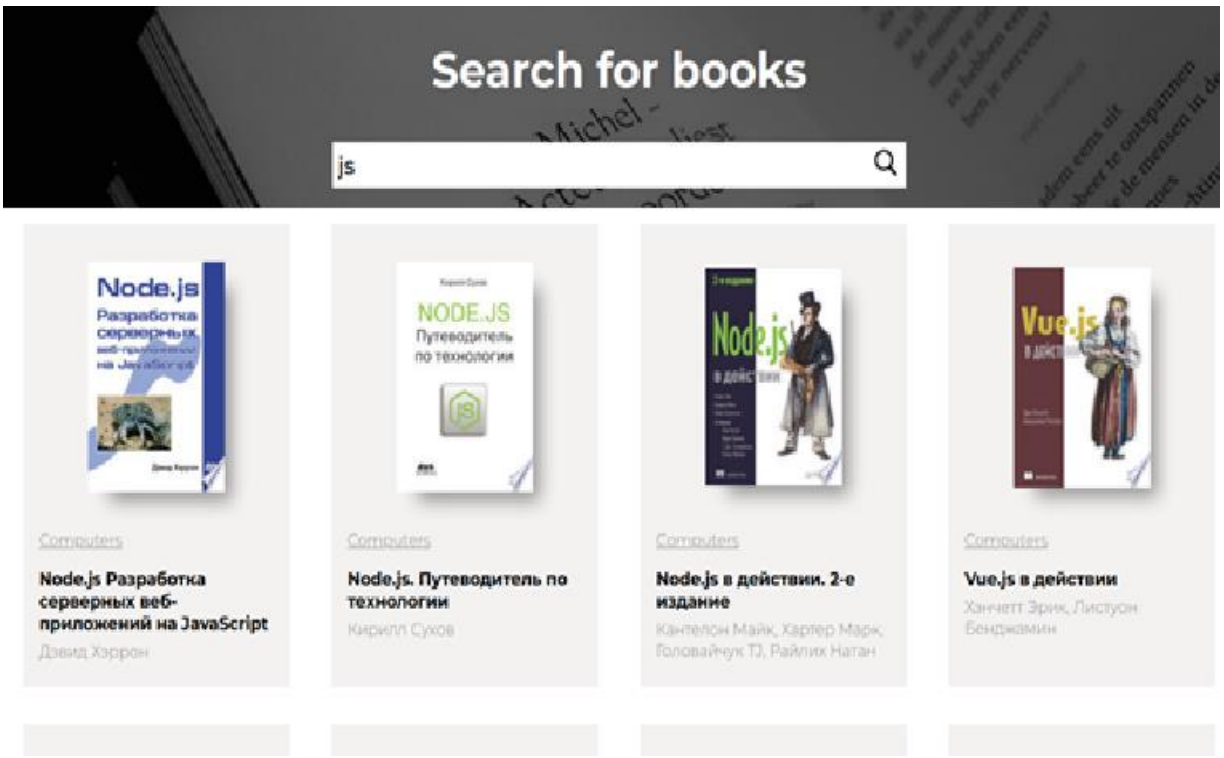
Требования

– на странице есть несколько карточек книг и строка поиска. На каждой карточке отображается обложка книги, название книги, название категории и имена авторов. Если для книги приходит несколько категорий, то отображается только первая. Авторы отображаются все. Если не приходит какой-либо части данных, то вместо нее просто пустое место;
– при загрузке приложения на странице отображаются карточки книг с полученными от API данными;
– если в поле поиска ввести слово и отправить поисковый запрос, на странице отобразятся карточки книг, в названиях которых есть это слово, если такие данные предоставляет API;
– в футере приложения есть ссылка на GitHub автора приложения, год создания приложения;
– при открытии приложения курсор находится в поле ввода;
– есть placeholder;
– автозаполнение поля ввода отключено (нет выпадающего списка с предыдущими запросами);
– поисковый запрос можно отправить нажатием клавиши Enter;
– после отправки поискового запроса и отображения результатов поиска, поисковый запрос продолжает отображаться в поле ввода;
– в поле ввода есть крестик при клике по которому поисковый запрос из поля ввода удаляется и отображается placeholder.

Ссылки на API

Документация: https://developers.google.com/books/docs/v1/using .
Для авторизации запросов к API выбрать способ с предоставлением API key (https://developers.google.com/books/docs/v1/using#APIKey)

Пример



Вариант 4. Галерея персонажей

Задача

Необходимо создать приложение, отображающее список персонажей в виде списка карточек «аватар + имя».

Требования

– при первой отрисовке показываем только первых 20 персонажей;
– реализована пагинация по принципу «load more». Ниже блока с карточками находится кнопка «Load more», по клику на нее к уже загруженным персонажам подгружаются еще. Шаг пагинации – 20;
– фильтрация по категориям. Выше списка карточек располагается селект «gender» с категориями: all, female, male, genderless, unknown. Если выбрано "all" (выбрано изначально), то поиск производится по всем категориям;
– в футере приложения есть ссылка на GitHub автора приложения, год создания приложения;
– при открытии приложения курсор находится в поле ввода;
– есть placeholder;
– автозаполнение поля ввода отключено (нет выпадающего списка с предыдущими запросами);
– поисковый запрос можно отправить нажатием клавиши Enter;

– после отправки поискового запроса и отображения результатов поиска, поисковый запрос продолжает отображаться в поле ввода;

– в поле ввода есть крестик при клике по которому поисковый запрос из поля ввода удаляется и отображается placeholder.

Ссылка на API

https://rickandmortyapi.com/documentation/#rest

Пример