

Курс:  
**«Язык сценариев JavaScript и библиотека jQuery»**

## Модуль 9. ECMAScript

### Задание

У вас недавно было задание по созданию страницы поиска фильмов. Сегодня вам необходимо усовершенствовать это задание.

### Технические требования

1. Логику работы с API необходимо описать в отдельном классе.  
Например, `class MovieService` с методами `search (title, type, page)` и `getMovie (movieId)`.
2. В этих методах необходимо реализовать только (!) обращение к API и получение результатов.
3. Логика изменения DOM должна быть реализована за пределами класса `MovieService`. Это может быть отдельный класс для взаимодействия с DOM или просто набор функций.
4. Методы класса, которые обращаются к API, должны быть асинхронными (используйте конструкцию `async/await`).

Для начала пользователю доступна форма поиска, приведенная на рисунке 1.

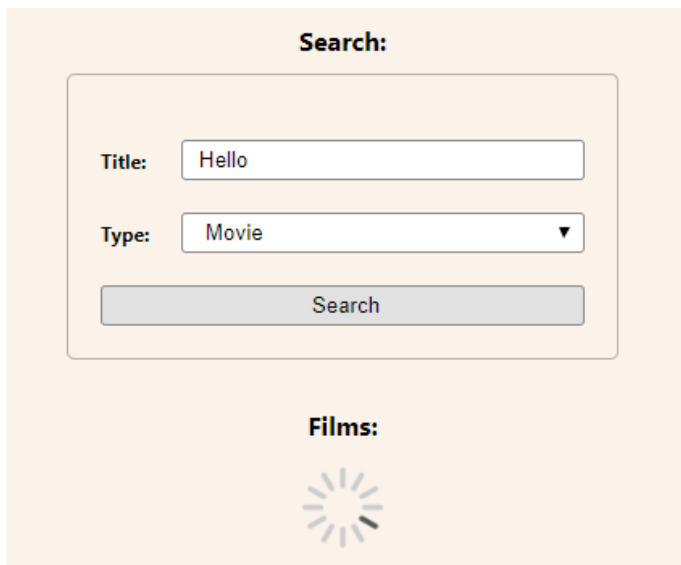
После того как пользователь введет данные и нажмет кнопку **Search** и до того как загрузятся все фильмы, пройдет какое-то время. Ведь будет идти обращение к удаленному серверу, а мы не знаем заранее, как быстро он ответит на наш запрос. В этот промежуток

времени ваш сайт не должен застывать в непонятном состоянии. Поэтому необходимо отображать иконку загрузки до тех пор, пока все фильмы не отобразятся на странице (рис. 2).



The diagram shows a search interface on a light orange background. At the top, the word "Search:" is centered. Below it is a rounded rectangle containing two labels, "Title:" and "Type:", each followed by a white input field. The "Type:" field is a dropdown menu with "Movie" selected and a downward arrow. Below these fields is a wide, light gray button with the text "Search" centered on it.

Рисунок 1



The diagram shows the same search interface as in Figure 1, but with additional content. The "Title:" input field now contains the text "Hello". Below the search form, the word "Films:" is centered. Underneath "Films:" is a circular loading spinner, which is a gray circle with several short lines radiating from it, indicating that data is being loaded.

Рисунок 2

Когда все фильмы (первые 10 штук) загружены и отображены на странице, иконка загрузки пропадает.

Под списком всех фильмов должна быть кнопка **More**, при нажатии на которую загружаются следующие 10 фильмов.

**Обращаем внимание**, что OMDb возвращает только 10 фильмов за 1 запрос и дает возможность указать номер страницы для выборки. Например, если под критерии поиска подходит 50 фильмов, значит будет 5 страниц по 10 фильмов. И при первом обращении (нажатии на кнопку **Search**) вы запрашиваете первую страницу, при следующих нажатиях на кнопку **More** – следующие страницы (рис. 3).

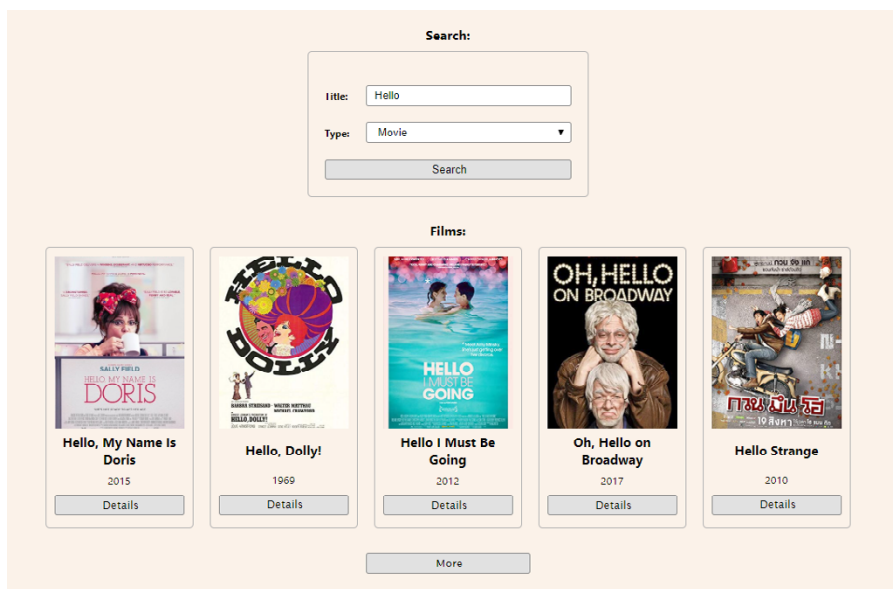


Рисунок 3

После нажатия на кнопку **More**, она блокируется и появляется иконка загрузки, до тех пор пока следующие 10 фильмов не отобразятся на странице (рис. 4).

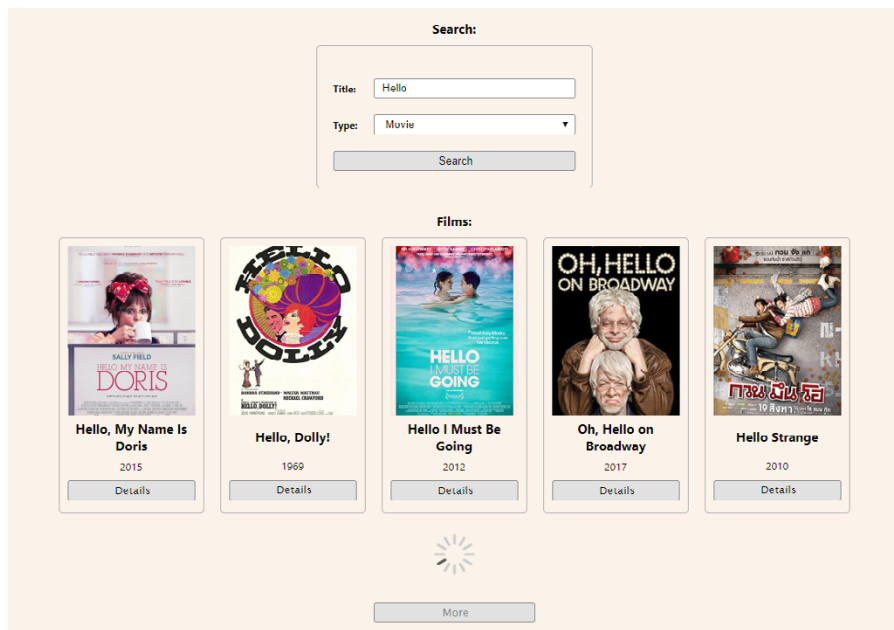


Рисунок 4

Следующие фильмы уже отображены на странице, и кнопка **More** разблокирована (см. рис. 5 на стр. 5).

При нажатии на кнопку **Details** необходимо открыть модальное окно и отображать иконку загрузки, пока данные загружаются на страницу (см. рис. 6 на стр. 5).

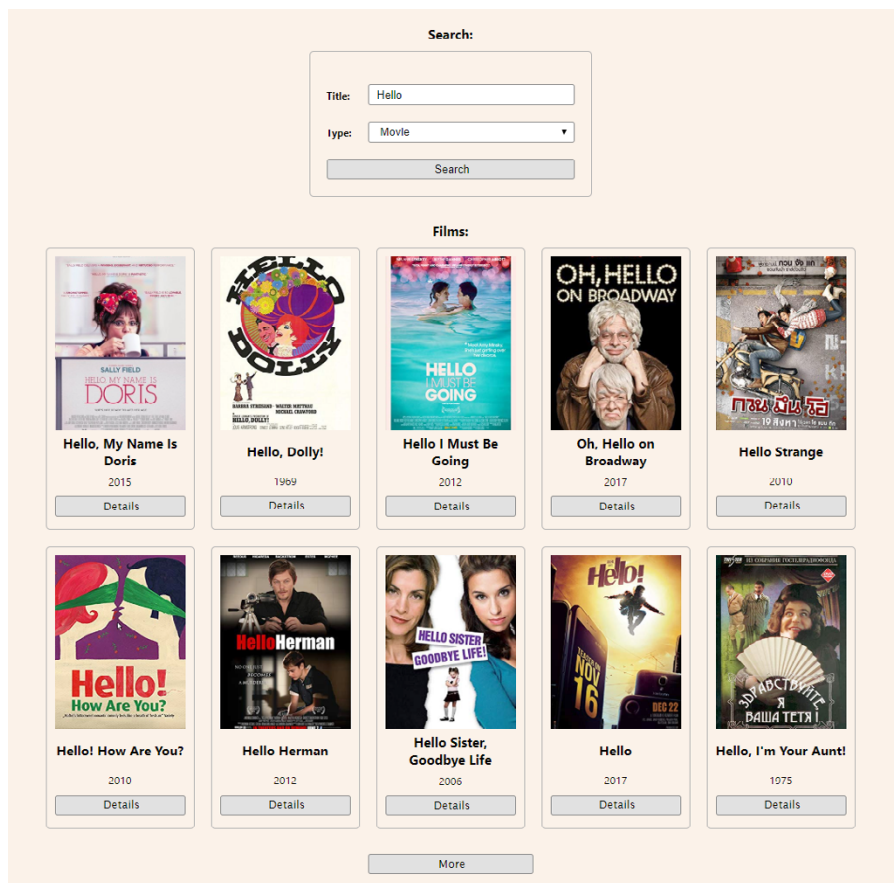


Рисунок 5

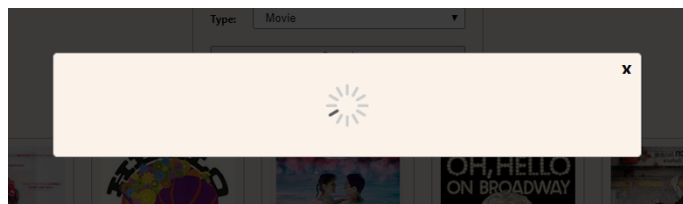


Рисунок 6

Данные о фильме загружены на страницу (рис. 7).



Рисунок 7

Ссылка на API OMDb: <http://www.omdbapi.com/> (необходимо зарегистрироваться для получения API KEY).

Ссылка на альтернативный API с фильмами (для того случая, если OMDb не будет работать): <https://developers.themoviedb.org/3/search/search-movies>.

Если же и этот API не будет работать, то вам придется самостоятельно найти другой доступный ресурс и адаптировать под него задание.