**03. Movies**

You can check your solutions in [Judge.](https://judge.softuni.org/Contests/Practice/Index/4698#2)

**Your Task**

Using **Mocha** and **Chai** write **JS Unit Tests** to test a variable namedmovieService, which represents an object. You may use the following code as a template:

|  |
| --- |
| describe(**"*Tests* …"**, **function**() {  describe(**"*TODO* …"**, **function**() {  ***it***(**"*TODO …*"**, **function**() {  *//* ***TODO:*** …  });  });  *//* ***TODO:*** …  }); |

The movieService object that should have the following functionality:

* getMovies**() -** A function that returns an object with a response from a movie service.
  + If the request is **successful**, return an object with status **200** and an array of **3 movies**, each containing **name**, **genre**, **year**, **director, rating**, **duration**, **language**, **desc** and **id**.
* addMovie**(movie) -** A function that adds a new movie to the list of movies. movie is an object that must contain the fields **name**, **genre**, **year**, **director, rating**, **duration**, **language**, **desc** and **id**.
  + If the movie is **successfully added**, return an object with status **201** and a message: "**Movie added successfully**."
  + If any of the **fields are missing or invalid**, return an object with status **400** and an error message: "**Invalid Movie Data**!"
* deleteMovie**(**movieId**)** - A function that deletes a movie by id.
  + movieId is a string with the name of the movie to be deleted.
  + If the movie is **successfully deleted**, return an object with status **200** and a message: "**Movie deleted successfully.**"
  + If the movie is **not found**, return an object with status **404** and an error message: "**Movie Not Found**!".
* updateMovie**(**oldName**,** newMovie**) -** A function that updates the information of an existing movie.
  + oldName is a string with the name of the movie to be updated.
  + newMovie is an object containing the new movie data.
  + If the movie is **successfully updated**, return an object with status **200** and a message: "**Movie updated successfully**."
  + If the movie is **not found**, return an object with status **404** and an error message: "**Movie Not Found**!"
  + If the **new movie data is invalid**, return an object with status **400** and an error message: "**Invalid Movie Data**!"

**JS Code**

To ease you in the process, you are provided with an implementation that meets all of the specification requirements for the movieServiceobject:

|  |
| --- |
| movieService.js |
| const movieService = {  movies: [  {  id: "1",  name: "Inception",  genre: "Sci-Fi",  year: 2010,  director: "Christopher Nolan",  rating: 8.8,  duration: 148,  language: "English",  desc: "A thief who steals corporate secrets through the use of dream-sharing technology."  },  {  id: "2",  name: "The Matrix",  genre: "Action",  year: 1999,  director: "Lana Wachowski, Lilly Wachowski",  rating: 8.7,  duration: 136,  language: "English",  desc: "A computer hacker learns about the true nature of reality and his role in the war against its controllers."  },  {  id: "3",  name: "Interstellar",  genre: "Adventure",  year: 2014,  director: "Christopher Nolan",  rating: 8.6,  duration: 169,  language: "English",  desc: "A team of explorers travel through a wormhole in space in an attempt to ensure humanity's survival."  }  ],  getMovies() {  return {  status: 200,  data: this.movies  };  },  addMovie(movie) {  if (!movie.id || !movie.name || !movie.genre || !movie.year || !movie.director || !movie.rating || !movie.duration || !movie.language || !movie.desc) {  return {  status: 400,  error: "Invalid Movie Data!"  };  }  this.movies.push(movie);  return {  status: 201,  message: "Movie added successfully."  };  },  deleteMovie(movieId) {  const movieIndex = this.movies.findIndex(movie => movie.id === movieId);  if (movieIndex === -1) {  return {  status: 404,  error: "Movie Not Found!"  };  }  this.movies.splice(movieIndex, 1);  return {  status: 200,  message: "Movie deleted successfully."  };  },  updateMovie(oldName, newMovie) {  const movieIndex = this.movies.findIndex(movie => movie.name === oldName);  if (movieIndex === -1) {  return {  status: 404,  error: "Movie Not Found!"  };  }  if (!newMovie.id || !newMovie.name || !newMovie.genre || !newMovie.year || !newMovie.director || !newMovie.rating || !newMovie.duration || !newMovie.language || !newMovie.desc) {  return {  status: 400,  error: "Invalid Movie Data!"  };  }  this.movies[movieIndex] = newMovie;  return {  status: 200,  message: "Movie updated successfully."  };  }  }; |

**Submission**

Submit your tests inside a describe**()** statement, as shown above.