# Exam Preparation II

**03. Games**

You can check your solutions in [Judge](https://judge.softuni.org/Contests/Practice/Index/4699#2).

**Your Task**

Using **Mocha** and **Chai** write **JS Unit Tests** to test a variable namedgameService, 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 gameServiceobject that should have the following functionality:

* getGames() **-** A function that returns an object with a response from a game service.
  + If the request is successful, return an object with status **200** and an array of **3 games**, each containing **id**, **title**, **genre**, **year**, **developer** and **description**.
* addGame(game)- A function that adds a new game to the list of games.
  + game is an object that must contain the fields **id**, **title**, **genre**, **year**, **developer** and **description**.
  + If all the required fields (**id**, **title**, **genre**, **year**, **developer**, **description**) are provided and **valid**, the game is added to the list and the method returns an object with a **status** of **201** (**created**) and a **success message**.
  + If any of the fields are missing or invalid, return an object with status **400** and an error message: "**Invalid Game Data**!"
* deleteGame(gameId)- A function that deletes a game by its id.
  + gameId is a string with the **ID** of the game to be deleted.
  + If the gameId is found in the list, the game is **deleted**, and an object with a status of **200** and a **success message**  "**Game deleted successfully**." is returned.
  + If the game is **not found**, return an object with status **404** and an error message: "**Game Not Found!**".
* updateGame(oldTitle, newGame) - A function that updates the information of an existing game.
  + oldId is a string with the id of the game to be updated.
  + newGame is an object containing the new game data.
* If the game with the given **oldId** is found and the **newGame** object contains all the necessary fields (**id**, **title**, **genre**, **year**, **developer**, **description**), the game is updated, and an object with a status of **200** and a success message "**Game updated successfully**" is returned.
  + If the game is **not found**, return an object with status **404** and an error message: "**Game Not Found!**".
  + If the new game data is **invalid**, return an object with status **400** and an error message: "**Invalid Game Data!**".

**JS Code**

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

|  |
| --- |
| gameService.js |
| const gameService = {  games: [  { id: "1", title: "The Legend of Zelda: Breath of the Wild", genre: "Action-adventure", year: 2017, developer: "Nintendo", description: "An action-adventure game in an open world." },  { id: "2", title: "God of War", genre: "Action-adventure", year: 2018, developer: "Santa Monica Studio", description: "An action-adventure game set in Norse mythology." },  { id: "3", title: "The Witcher 3: Wild Hunt", genre: "RPG", year: 2015, developer: "CD Projekt Red", description: "An RPG set in a fantasy open world." }  ],  getGames() {  return {  status: 200,  data: this.games  };  },  addGame(game) {  if (!game.id || !game.title || !game.genre || !game.year || !game.developer || !game.description) {  return {  status: 400,  error: "Invalid Game Data!"  };  }  this.games.push(game);  return {  status: 201,  message: "Game added successfully."  };  },  deleteGame(gameId) {  const gameIndex = this.games.findIndex(game => game.id === gameId);  if (gameIndex === -1) {  return {  status: 404,  error: "Game Not Found!"  };  }  this.games.splice(gameIndex, 1);  return {  status: 200,  message: "Game deleted successfully."  };  },  updateGame(oldId, newGame) {  const gameIndex = this.games.findIndex(game => game.id === oldId);  if (gameIndex === -1) {  return {  status: 404,  error: "Game Not Found!"  };  }  if (!newGame.id || !newGame.title || !newGame.genre || !newGame.year || !newGame.developer || !newGame.description) {  return {  status: 400,  error: "Invalid Game Data!"  };  }  this.games[gameIndex] = newGame;  return {  status: 200,  message: "Game updated successfully."  };  }  }; |

**Submission**

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