

Games Database API

Genres

Path: [/genre/all](#)

Method: GET

Returns: all the genres with **HttpStatus 200**

```
[
  {
    "id": 1,
    "name": "Horror"
  },
]
```

Path: [/genre](#)

Method: GET

Parameters:

- **id** - the id of the searched genre

Returns:

1. The genre with **HttpStatus 200** - if the genre is found

```
{
  "id": 1,
  "name": "Horror"
}
```

2. **HttpStatus 404** - if the genre does not exist
-

Path: [/genre/insert](#)

Method: POST

Parameters:

- **name** - the name of the genre

Returns:

1. The inserted genre with **HttpStatus 201** - if the genre is inserted successfully

```
{
  "id": 1,
  "name": "Horror"
}
```

2. **HttpStatus 406** - if the genre we try to create is with empty name
 3. **HttpStatus 400** - if the genre already exist
 4. **HttpStatus 404** - if the genre is not inserted successfully
-

Path: [/genre/update](#)

Method: PUT

Parameters:

- **id** - the id of the genre that will be updated
- **name** - the name of the genre

Returns:

1. The **updated genre** with **HttpStatus 200** - *if the genre is updated successfully*

```
{  
  "id": 1,  
  "name": "Horror"  
}
```

2. **HttpStatus 404** - *if the genre does not exist*
 3. **HttpStatus 406** - *if the genre we try to update is with empty name*
 4. **HttpStatus 409** - *if there is already genre with this name in the database*
-

Path: </genre/delete>

Method: DELETE

Parameters:

- **id** - *the id of the genre that will be deleted*

Returns:

1. **True** with **HttpStatus 200** - *if the genre is deleted successfully*
2. **False** with **HttpStatus 404** - *if the genre does not exist*

Platforms

Path: </platform/all>

Method: GET

Returns: all the platforms with **HttpStatus 200**

```
[
  {
    "id": 1,
    "name": "PC"
  },
]
```

Path: </platform>

Method: GET

Parameters:

- **id** - the id of the searched platform

Returns:

3. The platform with **HttpStatus 200** - if the platform is found

```
{
  "id": 1,
  "name": "PC"
}
```

4. **HttpStatus 404** - if the genre does not exist
-

Path: </platform/insert>

Method: POST

Parameters:

- **name** - the name of the platform

Returns:

5. The inserted platform with **HttpStatus 201** - if the platform is inserted successfully

```
{
  "id": 1,
  "name": "PC"
}
```

6. **HttpStatus 406** - if the platform we try to create is with empty name
 7. **HttpStatus 400** - if the platform already exist
 8. **HttpStatus 404** - if the platform is not inserted successfully
-

Path: </platform/update>

Method: PUT

Parameters:

- **id** - the id of the platform that will be updated
- **name** - the name of the platform

Returns:

5. The updated platform with **HttpStatus 200** - if the platform is updated successfully

```
{
  "id": 1,
```

```
"name": "PC"
}
```

6. **HttpStatus 404** - *if the platform does not exist*
 7. **HttpStatus 406** - *if the platform we try to update is with empty name*
 8. **HttpStatus 409** - *if there is already platform with this name in the database*
-

Path: </platform/delete>

Method: DELETE

Parameters:

- **id** - *the id of the platform that will be deleted*

Returns:

3. **True** with **HttpStatus 200** - *if the platform is deleted successfully*
4. **False** with **HttpStatus 404** - *if the platform does not exist*

Developers

Path: </developer/all>

Method: GET

Returns: all the developers with **HttpStatus 200**

```
[
  {
    "id": 1,
    "name": "Atari",
    "description": "Arati is ..."
  },
]
```

Path: </developer>

Method: GET

Parameters:

- **id** - *the id of the searched developer*

Returns:

1. **The developer with HttpStatus 200** - *if the developer is found*

```
{
  "id": 1,
  "name": "Atari",
  "description": "Atari is ..."
}
```

2. **HttpStatus 404** - *if the developer does not exist*
-

Path: </developer/insert>

Method: POST

Parameters:

- **name** - *the name of the developer*
- **? description** - *description about the developer*

Returns:

1. **The inserted developer with HttpStatus 201** - *if the developer is inserted successfully*

```
{
  "id": 1,
  "name": "Atari"
  "description": "Atari is ..."
}
```

2. **HttpStatus 406** - *if the developer we try to create is with empty name*
 3. **HttpStatus 400** - *if the developer already exist*
 4. **HttpStatus 404** - *if the developer is not inserted successfully*
-

Path: </developer/update>

Method: PUT

Parameters:

- **id** - *the id of the developer that will be updated*

- **? name** - *the name of the developer*
- **? description** - *description about the developer*

Returns:

1. The **updated developer** with **HttpStatus 200** - *if the developer is updated successfully*

```
{  
  "id": 1,  
  "name": "Atari"  
  "description": "Arati is ..."  
}
```

2. **HttpStatus 404** - *if the developer does not exist*
3. **HttpStatus 406** - *if the developer we try to update is with empty name*
4. **HttpStatus 409** - *if there is already developer with this name in the database*

Path: </developer/delete>

Method: DELETE

Parameters:

- **id** - *the id of the developer that will be deleted*

Returns:

1. **True** with **HttpStatus 200** - *if the developer is deleted successfully*
2. **False** with **HttpStatus 404** - *if the developer does not exist*

Games

Path: [/game/all](#)

Method: GET

Returns: all the games with **HttpStatus 200**

```
[
  {
    "id": 1,
    "name": "Roller Coaster Tycoon 3",
    "description": "RCT 3 is ...",
    "image": "game_cover.png",
    "genres": [
      {
        "id": 74,
        "name": "Tycoon"
      },
      {
        "id": 75,
        "name": "Simulator"
      }
    ],
    "platforms": [
      {
        "id": 1,
        "name": "PC"
      }
    ],
    "developer": {
      "id": 1,
      "name": "Atari",
      "description": "Atari is ..."
    }
  },
]
```

Path: [/game](#)

Method: GET

Parameters:

- **id** - the id of the searched game

Returns:

1. **The game with HttpStatus 200** - if the game is found

```
{
  "id": 1,
  "name": "Roller Coaster Tycoon 3",
  "description": "RCT 3 is ...",
  "image": "game_cover.png",
  "genres": [
    {
```

```

        "id": 74,
        "name": "Tycoon"
    },
    {
        "id": 75,
        "name": "Simulator"
    }
],
"platforms": [
    {
        "id": 1,
        "name": "PC"
    }
],
"developer": {
    "id": 1,
    "name": "Atari",
    "description": "Atari is ..."
}
}

```

2. HttpStatus 404 - if the game does not exist

Path: </game/search>

Method: GET

Parameters:

- **? genres_id_list** - list of genres that we want the searched games to belong
- **? platforms_id_list** - list of platforms that we want the searched games to belong

Returns:

3. All the games that belong to searched genres and/or platforms with HttpStatus 200

```

[
  {
    "id": 1,
    "name": "Roller Coaster Tycoon 3",
    "description": null,
    "image": null,
    "genres": [
      {
        "id": 1,
        "name": "Arcade"
      },
      {
        "id": 2,
        "name": "Action"
      },
      {
        "id": 3,
        "name": "Adventure"
      }
    ]
  }
]

```



```

    }
  ],
  "platforms": [
    {
      "id": 1,
      "name": "PC"
    }
  ],
  "developer": {
    "id": 1,
    "name": "Atari",
    "description": null
  }
},
]

```

Path: </game/insert>

Method: POST

Parameters:

- **name** - the name of the game
- **? description** - description about the game
- **? image** - the image
- **? developer_id** - the developer id
- **? genres_id_list** - the genres id list, example: 1, 2, 3
- **? platforms_id_list** - the platforms id list, example: 1, 2, 3

Notes: If front-end is on JavaScript, because there is image that have to be sent from front-end to back-end, the data have to be sent as object of FormData(); Example:

```

var formData = new FormData();

formData.append("name", $("#name").val())
formData.append("image", $("#image")[0].files[0]);
formData.append("genres_id_list", $("#genres").val());

$.ajax({
  url: '/game/insert',
  type: 'POST',
  processData: false,
  contentType: false,
  cache: false,
  data: formData,
  enctype: 'multipart/form-data',
  complete: function(data) {
    console.log(data);
  }
})

```

```
});
```

Returns:

1. The **inserted game** with **HttpStatus 201** - *if the game is inserted successfully*

```
{
  "id": 1,
  "name": "Roller Coaster Tycoon 3",
  "description": "RCT 3 is ...",
  "image": "game_cover.png",
  "genres": [
    {
      "id": 74,
      "name": "Tycoon"
    },
    {
      "id": 75,
      "name": "Simulator"
    }
  ],
  "platforms": [
    {
      "id": 1,
      "name": "PC"
    }
  ],
  "developer": {
    "id": 1,
    "name": "Atari",
    "description": "Atari is ..."
  }
}
```

2. **HttpStatus 406** - *if the game we try to insert is with empty name*
3. **HttpStatus 400** - *if the game already exist, the developer is not found, some of the genres are not found or some of the platforms are not found*
4. **HttpStatus 404** - *if the game is not inserted successfully*

Path: </game/update>

Method: PUT

Parameters:

- **id** - *the id of the game that will be updated*
- **? name** - *the name of the game*
- **? description** - *description about the game*
- **? image** - *the image*
- **? developer_id** - *the developer id*
- **? genres_id_list** - *the genres id list, example: 1, 2, 3*

Returns:

1. The **updated game** with **HttpStatus 200** - *if the game is updated successfully*

```
{
```

```

    "id": 1,
    "name": "Roller Coaster Tycoon 3",
    "description": "RCT 3 is ...",
    "image": "game_cover.png",
    "genres": [
      {
        "id": 74,
        "name": "Tycoon"
      },
      {
        "id": 75,
        "name": "Simulator"
      }
    ],
    "platforms": [
      {
        "id": 1,
        "name": "PC"
      }
    ],
    "developer": {
      "id": 1,
      "name": "Atari",
      "description": "Atari is ..."
    }
  }
}

```

2. **HttpStatus 404** - if the game does not exist
3. **HttpStatus 406** - if the game we try to update is with empty name
4. **HttpStatus 400** - if the developer is not found, some of the genres are not found or some of the platforms are not found
5. **HttpStatus 409** - if there is already game with this name in the database

Path: </game/delete>

Method: DELETE

Parameters:

- **id** - the id of the game that will be deleted

Returns:

1. **True** with **HttpStatus 200** - if the game is deleted successfully
2. **False** with **HttpStatus 404** - if the game does not exist