# OpenAPI definition

No description provided (generated by Swagger Codegen https://github.com/swagger-api/swagger-codegen)
More information: https://helloreverb.com
Contact Info: hello@helloreverb.com
Version: v0
All rights reserved
http://apache.org/licenses/LICENSE-2.0.html

## Access

## Methods

[ Jump to Models ]

### Table of Contents

# MovieController

### `GET /movie`

Up

Get Movies playing in the given time intervall. (**getAllPlayingMovies**)

#### Query parameters

**startTime (required)**
*Query Parameter* — start of the time interval in which the Movie is played. format: date-time

**endTime (required)**
*Query Parameter* — end of the time interval in which the Movie is played. format: date-time

### Return type
array[MovieDto]

### Example data
Content-Type: application/json

```
[ {
  "id" : 0,
  "title" : "title"
}, {
  "id" : 0,
  "title" : "title"
} ]
```

### Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses
**200**
Found some (maybe zero) Movies.
**400**
Invalid time intervall.

## `GET /movie/{id}`

Up

Get a Movie by its id. (**getById4**)

### Path parameters

**id (required)**
*Path Parameter* — ID of Movie to be searched. format: int64

### Return type
MovieDto

### Example data
Content-Type: application/json

```
{
  "id" : 0,
  "title" : "title"
}
```

### Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses
**200**
Found the Movie. MovieDto
**400**
Invalid id supplied.

**404**
Movie not found.

### `GET /movie/title/{title}`

[Up](#)

Get a Movie by its title. (**getByTitle**)

#### Path parameters

**title (required)**
*Path Parameter* — Title of Movie to be searched.

#### Return type
[MovieDto](#)

#### Example data
Content-Type: application/json

```
{
  "id" : 0,
  "title" : "title"
}
```

#### Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

#### Responses
**200**
Found the Movie. [MovieDto](#)
**404**
Movie not found.

# ScreeningController

### `GET /screening`

[Up](#)

Get Screenings of the given Movie in the given Theater in the given time intervall. (**getAllScreeningsOfMovieInTheaterPlaying**)

#### Query parameters

**startTime (required)**
*Query Parameter* — start of the time interval in which the Screening is played. format: date-time

**endTime (required)**
*Query Parameter* — end of the time interval in which the Screening is played. format: date-time

**movieId (required)**
*Query Parameter* — The id of the Movie of the Screening. format: int64

**theaterId (required)**
*Query Parameter* — The id of the Theatre of the Screening. format: int64

#### Return type
array[[ScreeningDto](#)]

#### Example data

Content-Type: application/json

```
[ {
  "theaterId" : 1,
  "startTime" : "2000-01-23T04:56:07.000+00:00",
  "movieId" : 6,
  "id" : 0,
  "endTime" : "2000-01-23T04:56:07.000+00:00",
  "movieTitle" : "movieTitle",
  "theaterName" : "theaterName"
}, {
  "theaterId" : 1,
  "startTime" : "2000-01-23T04:56:07.000+00:00",
  "movieId" : 6,
  "id" : 0,
  "endTime" : "2000-01-23T04:56:07.000+00:00",
  "movieTitle" : "movieTitle",
  "theaterName" : "theaterName"
} ]
```

### Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses

**200**
Found some (maybe zero) Screenings.
**400**
Invalid time intervall.
**404**
The given Movie or Theater not found.

## `GET /screening/{id}`

Up

Get a Screening by its id. (**getById3**)

### Path parameters

**id (required)**
*Path Parameter* — ID of Screening to be searched. format: int64

### Return type

ScreeningDto

### Example data

Content-Type: application/json

```
{
  "theaterId" : 1,
  "startTime" : "2000-01-23T04:56:07.000+00:00",
  "movieId" : 6,
  "id" : 0,
  "endTime" : "2000-01-23T04:56:07.000+00:00",
  "movieTitle" : "movieTitle",
  "theaterName" : "theaterName"
}
```

### Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

**Responses**
**200**
Found the Screening. [ScreeningDto]
**400**
Invalid id supplied.
**404**
Screening not found.

## SeatController

### `GET /seat`

[Up]

Get all available Seats for the given Screening. (**getAllAvailableSeatsForScreening**)

**Query parameters**

**screeningId (required)**
*Query Parameter* — ID of Screening we search available Seats for. format: int64

**Return type**
array[SeatDto]

**Example data**
Content-Type: application/json

```
[ {
  "theaterId" : 6,
  "name" : "name",
  "id" : 0,
  "theaterName" : "theaterName"
}, {
  "theaterId" : 6,
  "name" : "name",
  "id" : 0,
  "theaterName" : "theaterName"
} ]
```

**Produces**
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

**Responses**
**200**
Found some available (maybe zero) Seats.
**404**
The given Screening not found.

### `GET /seat/{id}`

[Up]

Get a Seat by its id. (**getById2**)

### Path parameters

**id (required)**
*Path Parameter* — ID of Seat to be searched. format: int64

### Return type
[SeatDto]

### Example data
Content-Type: application/json

```
{
  "theaterId" : 6,
  "name" : "name",
  "id" : 0,
  "theaterName" : "theaterName"
}
```

### Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses
**200**
Found the Seat. [SeatDto]
**400**
Invalid id supplied.
**404**
Seat not found.

# TheaterController

## GET /theater

Up

Get Theaters playing the given Movie in the given time intervall. (**getAllPlayingTheaters**)

### Query parameters

**startTime (required)**
*Query Parameter* — start of the time interval in which the Movie is played. format: date-time

**endTime (required)**
*Query Parameter* — end of the time interval in which the Movie is played. format: date-time

**movieId (required)**
*Query Parameter* — ID of Movie we are looking for. format: int64

### Return type
array[[TheaterDto]]

### Example data
Content-Type: application/json

```
[ {
  "name" : "name",
  "id" : 0
}, {
  "name" : "name",
  "id" : 0
} ]
```

**Produces**
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

**Responses**
**200**
Found some (maybe zero) Theaters.
**400**
Invalid time intervall.
**404**
The given Movie not found.

## `GET /theater/{id}`

Up

Get a Theater by its id. (**getById1**)

**Path parameters**

**id (required)**
*Path Parameter* — ID of Theater to be searched. format: int64

**Return type**
TheaterDto

**Example data**
Content-Type: application/json

```
{
  "name" : "name",
  "id" : 0
}
```

**Produces**
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

**Responses**
**200**
Found the Theater. TheaterDto
**400**
Invalid id supplied.
**404**
Theater not found.

## `GET /theater/name/{name}`

Up

Get a Theater by its name. (**getByName**)

**Path parameters**

**name (required)**
*Path Parameter* — Name of Theater to be searched.

### Return type
TheaterDto

### Example data
Content-Type: application/json

```
{
  "name" : "name",
  "id" : 0
}
```

### Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses
**200**
Found the Theater. TheaterDto
**404**
Theater not found.

## TicketController

### POST /ticket/bookn

Up

Creates the given number of Tickets for the Screening with some available Seats. (**createNTickets**)

### Query parameters

**screeningId (required)**
*Query Parameter* — ID of Screening we are to create Tickets for. format: int64

**numberOfTickets (required)**
*Query Parameter* — Number of Ticket to be created. format: int32

### Return type
array[TicketDto]

### Example data
Content-Type: application/json

```
[ {
  "screeningTheaterName" : "screeningTheaterName",
  "screeningId" : 1,
  "seatId" : 6,
  "seatName" : "seatName",
  "id" : 0,
  "screeningStartTime" : "2000-01-23T04:56:07.000+00:00",
  "screeningMovieTitle" : "screeningMovieTitle",
  "screeningEndTime" : "2000-01-23T04:56:07.000+00:00"
}, {
  "screeningTheaterName" : "screeningTheaterName",
  "screeningId" : 1,
  "seatId" : 6,
  "seatName" : "seatName",
  "id" : 0,
  "screeningStartTime" : "2000-01-23T04:56:07.000+00:00",
  "screeningMovieTitle" : "screeningMovieTitle",
  "screeningEndTime" : "2000-01-23T04:56:07.000+00:00"
} ]
```

**Produces**
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

**Responses**
**201**
Successfully created all Tickets.
**400**
Invalid number of Tickets supplied or there are not enough available Seats.
**404**
The given Screening not found.

## `POST /ticket/booknretry`

Up

Creates the given number of Tickets for the Screening with some available Seats optimistically. (**createNTicketsWithRetry**)

### Query parameters

**screeningId (required)**
*Query Parameter* — ID of Screening we are to create Tickets for. format: int64

**numberOfTickets (required)**
*Query Parameter* — Number of Ticket to be created. format: int32

### Return type
array[TicketDto]

### Example data
Content-Type: application/json

```
[ {
  "screeningTheaterName" : "screeningTheaterName",
  "screeningId" : 1,
  "seatId" : 6,
  "seatName" : "seatName",
  "id" : 0,
  "screeningStartTime" : "2000-01-23T04:56:07.000+00:00",
  "screeningMovieTitle" : "screeningMovieTitle",
  "screeningEndTime" : "2000-01-23T04:56:07.000+00:00"
}, {
  "screeningTheaterName" : "screeningTheaterName",
  "screeningId" : 1,
  "seatId" : 6,
  "seatName" : "seatName",
  "id" : 0,
  "screeningStartTime" : "2000-01-23T04:56:07.000+00:00",
  "screeningMovieTitle" : "screeningMovieTitle",
  "screeningEndTime" : "2000-01-23T04:56:07.000+00:00"
} ]
```

**Produces**
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses
**201**
Successfully created all Tickets.
**400**
Invalid number of Tickets supplied or there are not enough available Seats.
**404**
The given Screening not found.
**409**
Could not create the tickets optimistically.

## `POST /ticket/book`

Up

Creates Tickets for the Screening and Seats. (**createTickets**)

### Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

### Request body

**body TicketOrderDto (required)**
*Body Parameter —*

### Return type
array[TicketDto]

### Example data
Content-Type: application/json

```
[ {
  "screeningTheaterName" : "screeningTheaterName",
  "screeningId" : 1,
  "seatId" : 6,
  "seatName" : "seatName",
  "id" : 0,
  "screeningStartTime" : "2000-01-23T04:56:07.000+00:00",
  "screeningMovieTitle" : "screeningMovieTitle",
  "screeningEndTime" : "2000-01-23T04:56:07.000+00:00"
}, {
  "screeningTheaterName" : "screeningTheaterName",
  "screeningId" : 1,
  "seatId" : 6,
  "seatName" : "seatName",
  "id" : 0,
  "screeningStartTime" : "2000-01-23T04:56:07.000+00:00",
  "screeningMovieTitle" : "screeningMovieTitle",
  "screeningEndTime" : "2000-01-23T04:56:07.000+00:00"
} ]
```

### Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses
**201**
Successfully created all Tickets.
**404**

The given Screening or one of the Seats not found.

## `GET /ticket/{id}`

Up

Get a Ticket by its id. (**getById**)

### Path parameters

**id (required)**
*Path Parameter* — ID of Ticket to be searched. format: int64

### Return type

TicketDto

### Example data

Content-Type: application/json

```
{
  "screeningTheaterName" : "screeningTheaterName",
  "screeningId" : 1,
  "seatId" : 6,
  "seatName" : "seatName",
  "id" : 0,
  "screeningStartTime" : "2000-01-23T04:56:07.000+00:00",
  "screeningMovieTitle" : "screeningMovieTitle",
  "screeningEndTime" : "2000-01-23T04:56:07.000+00:00"
}
```

### Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

### Responses

**200**
Found the Ticket. TicketDto
**400**
Invalid id supplied.
**404**
Ticket not found.

## Models

[ Jump to Methods ]

### Table of Contents

### `MovieDto`

Up

**id**
*Long* format: int64

**title**
*String*

## `ScreeningDto` [Up]

**id**
*Long* format: int64

**startTime**
*Date* format: date-time

**endTime**
*Date* format: date-time

**movieId**
*Long* format: int64

**movieTitle**
*String*

**theaterId**
*Long* format: int64

**theaterName**
*String*

## `SeatDto` [Up]

**id**
*Long* format: int64

**name**
*String*

**theaterId**
*Long* format: int64

**theaterName**
*String*

## `TheaterDto` [Up]

**id**
*Long* format: int64

**name**
*String*

## `TicketDto` [Up]

**id**
*Long* format: int64

**seatId**
*Long* format: int64

**seatName**
*String*

**screeningId**
*Long* format: int64

**screeningMovieTitle**

String

**screeningTheaterName**
String

**screeningStartTime**
Date format: date-time

**screeningEndTime**
Date format: date-time

## TicketOrderDto

Up

The TicketOrder containing the ID of the Screening and the Seats we want to create tickets for.

**screeningId**
Long format: int64

**seatIds**
array[Long] format: int64