

Package com.mapmyjourney.backend.controller

## Class TripMemberController

java.lang.Object  
com.mapmyjourney.backend.controller.TripMemberController

```
@RestController
@RequestMapping("/api/trips/{tripId}/members")
public class TripMemberController
extends Object
```

Controlador REST para gestionar miembros de viajes.

### Constructor Summary

#### Constructors

##### Constructor

##### Description

TripMemberController()

### Method Summary

#### All Methods

#### Instance Methods

#### Concrete Methods

##### Modifier and Type

##### Method

##### Description

org.springframework.http.ResponseEntity<TripMemberDTO>

addMember(Long tripId, @Valid AddMemberRequestDTO request)

1.

org.springframework.http.ResponseEntity<TripMemberDTO>

changeMemberRole(Long tripId, Long userId, @Valid ChangeMemberRoleRequestDTO request)

4.

org.springframework.http.ResponseEntity<TripMemberDTO>

getMember(Long tripId, Long userId)

3.

org.springframework.http.ResponseEntity<List<TripMemberDTO>>

getTripMembers(Long tripId)

2.

org.springframework.http.ResponseEntity<Void>

leaveTrip(Long tripId)

6.

org.springframework.http.ResponseEntity<Void>

removeMember(Long tripId, Long userId)

5.

#### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait

## Constructor Details

### TripMemberController

```
public TripMemberController()
```

## Method Details

### addMember

```
@PreAuthorize("hasRole('USER')")
@PostMapping
public org.springframework.http.ResponseEntity<TripMemberDTO> addMember(@PathVariable
    Long tripId,
    @Valid
    @Valid AddMemberRequestDTO request)
```

1. Agrega un nuevo miembro al viaje. POST /api/trips/{tripId}/members

### getTripMembers

```
@PreAuthorize("hasRole('USER')")
@GetMapping
public org.springframework.http.ResponseEntity<List<TripMemberDTO>> getTripMembers(@PathVariable
    Long tripId)
```

2. Obtiene todos los miembros del viaje. GET /api/trips/{tripId}/members

### getMember

```
@PreAuthorize("hasRole('USER')")
@GetMapping("/{userId}")
public org.springframework.http.ResponseEntity<TripMemberDTO> getMember(@PathVariable
    Long tripId,
    @PathVariable
    Long userId)
```

3. Obtiene un miembro específico. GET /api/trips/{tripId}/members/{userId}

### changeMemberRole

```
@PreAuthorize("hasRole('USER')")
@PutMapping("/{userId}/role")
public org.springframework.http.ResponseEntity<TripMemberDTO> changeMemberRole
(@PathVariable
    Long tripId,
    @PathVariable
    Long userId,
    @Valid
    @Valid ChangeMemberRoleRequestDTO request)
```

4. Cambia el rol de un miembro. PUT /api/trips/{tripId}/members/{userId}/role Solo OWNER puede cambiar roles.

### removeMember

```
@PreAuthorize("hasRole('USER')")
@DeleteMapping("/{userId}")
public org.springframework.http.ResponseEntity<Void> removeMember(@PathVariable
    Long tripId,
    @PathVariable
    Long userId)
```

5. Elimina un miembro del viaje. DELETE /api/trips/{tripId}/members/{userId} Solo OWNER puede remover miembros.

### leaveTrip

```
@PreAuthorize("hasRole('USER')")
@PostMapping("/leave")
public org.springframework.http.ResponseEntity<Void> leaveTrip(@PathVariable
    Long tripId)
```

6. Permite que el usuario abandone el viaje. POST /api/trips/{tripId}/members/leave