

Package com.mapmyjourney.backend.controller

Class TripController

java.lang.Object[↗]
com.mapmyjourney.backend.controller.TripController

```
@RestController
@RequestMapping("/trips")
public class TripController
extends Object↗
```

Controlador REST para gestionar viajes.

Constructor Summary

| Constructors |
|------------------|
| Constructor |
| Description |
| TripController() |

Method Summary

| All Methods | Instance Methods | Concrete Methods |
|---|------------------|------------------|
| Modifier and Type | | |
| Method | | |
| Description | | |
| org.springframework.http.ResponseEntity<TripDTO> createTrip(@Valid TripCreateRequestDTO request) | | |
| 1. | | |
| org.springframework.http.ResponseEntity<Void [↗] > deleteTrip(Long [↗] tripId) | | |
| 6. | | |
| org.springframework.http.ResponseEntity<List [↗] <TripDTO>> getMyTrips(int page, int size) | | |
| 3. | | |
| org.springframework.http.ResponseEntity<TripDTO> getTripByCode(String [↗] tripCode) | | |
| 4. | | |
| org.springframework.http.ResponseEntity<TripDTO> getTripById(Long [↗] tripId) | | |
| 2. | | |
| org.springframework.http.ResponseEntity<TripDTO> updateTrip(Long [↗] tripId, @Valid TripCreateRequestDTO request) | | |
| 5. | | |
| Methods inherited from class java.lang.Object [↗] | | |
| clone [↗] , equals [↗] , finalize [↗] , getClass [↗] , hashCode [↗] , notify [↗] , notifyAll [↗] , toString [↗] , wait [↗] , wait [↗] , wait [↗] | | |

Constructor Details

TripController

```
public TripController()
```

Method Details

createTrip

```
@PostMapping
@PreAuthorize("hasRole('\\USER\\')")
public org.springframework.http.ResponseEntity<TripDTO> createTrip(@Valid @RequestBody
                                                                    @Valid TripCreateRequestDTO request)
```

1. Crea un nuevo viaje. POST /api/trips

getTripById

```
@GetMapping("/{tripId}")
@PreAuthorize("hasRole('\\USER\\')")
public org.springframework.http.ResponseEntity<TripDTO> getTripById(@PathVariable
                                                                    Long tripId)
```

2. Obtiene un viaje por ID. GET /api/trips/{tripId}

getMyTrips

```
@GetMapping("/my-trips")
@PreAuthorize("hasRole('\\USER\\')")
public org.springframework.http.ResponseEntity<List<TripDTO>> getMyTrips(@RequestParam(defaultValue="0")
                                                                    int page,
                                                                    @RequestParam(defaultValue="10")
                                                                    int size)
```

3. Obtiene todos los viajes del usuario logueado (con paginación). GET /api/trips/my-trips? page=0&size=10&sort=createdAt,desc

getTripByCode

```
@GetMapping("/code/{tripCode}")
@PreAuthorize("hasRole('\\USER\\')")
public org.springframework.http.ResponseEntity<TripDTO> getTripByCode(@PathVariable
                                                                    String tripCode)
```

4. Obtiene un viaje por código (para invitaciones). GET /api/trips/code/{tripCode}

updateTrip

```
@PutMapping("/{tripId}")
@PreAuthorize("hasRole('\\USER\\')")
public org.springframework.http.ResponseEntity<TripDTO> updateTrip(@PathVariable
                                                                    Long tripId,
                                                                    @Valid
                                                                    @Valid TripCreateRequestDTO request)
```

5. Actualiza un viaje. PUT /api/trips/{tripId} Solo el OWNER puede actualizar.

deleteTrip

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

6. Elimina un viaje. DELETE /api/trips/{tripId} Solo el OWNER puede eliminar.

