Put의개념

HTTP PUT 메서드는 요청 페이로드를 사용해 새로운 리소스를 생성하거나, 대상 리소스를 나타내는 데이터를 대체합니다. PUT과 POST의 차이는 멱등성으로, PUT은 멱등성을 가집니다. PUT은 한 번을 보내도, 여러 번을 연속으로 보내도 같은 효과를 보입니다. 즉, 부수 효과가 없습니다.

※ 멱등성
동일한 요청을 한 번 보내는 것과 여러 번 연속으로 보내는 것이 같은 효과를 지니고, 서버의 상태도 동일하게 남을 때, 해당 HTTP 메서드가 멱등성을 가졌다고 말합니다. 다른 말로는, 멱등성 메서드에는 통계 기록 등을 제외하면 어떠한 부수 효과(side effect)도 존재해서는 안됩니다. 올바르게 구현한 경우 GET, HEAD, PUT, DELETE 메서드는 멱등성을 가지며, POST 메서드는 그렇지 않습니다. 모든 안전한 메서드는 멱등성도 가집니다.

예제

요청

```
PUT /new.html HTTP/1.1
Host: example.com
Content-type: text/html
Content-length: 16

<p>New File</p>
```

응답

데이터가 생성된 경우

```
HTTP/1.1 201 Created
Content-Location: /new.html
```

데이터가 없는경우는 200(OK) 또는 204(No Content) 응답

```
HTTP/1.1 204 No Content
Content-Location: /existing.html
```

Kotlin - SpringBoot

Controller 예제

```
@RestController
@RequestMapping("/api")
```

```
class PutApiController {

    //비추천 - @PutMapping 사용
    @RequestMapping(method = [RequestMethod.PUT], path = ["/request-mapping"])
    fun requestMapping(): String{
        return "request-mapping - put method"
    }


    //추천
    @PutMapping("/put-mapping")
    fun putMapping(@RequestBody svcDto: UserRequestDto): UserResponseDto {
        return UserResponseDto().apply {
            this.result = ResultDto().apply {
                this.resultCode = "OK"
                this.resultMessage = "성공"
            }
        }.apply {
            this.description = "~~~~~~~~~~~~~~"
        }.apply {
            this.userList.add(svcDto)

            this.userList.add(UserRequestDto().apply {
                this.name = "Steve"
                this.age = 22
            })

            this.userList.add(UserRequestDto().apply {
                this.name = "Ah~~~~"
                this.age = 18
            })
        }
    }
}
```

Dto - UserRequestDto

```
@JsonNaming(PropertyNamingStrategies.SnakeCaseStrategy::class)
//@JsonNaming(PropertyNamingStrategy.SnakeCaseStrategy::class) //deprecated
data class UserRequestDto(
    var name: String?=null,
    var age: Int?=null,
    var email: String?=null,
    var address: String?=null,
)
```

Dto - UserResponseDto

```kotlin
@JsonNaming(PropertyNamingStrategies.SnakeCaseStrategy::class)
//@JsonNaming(PropertyNamingStrategy.SnakeCaseStrategy::class) //deprecated
data class UserResponseDto(
    var result:ResultDto?=null,
    var description:String?=null,

    @JsonProperty("user")
    var userList: MutableList<UserRequestDto> = mutableListOf(),
)


data class ResultDto (
    var resultCode: String ?= null,
    var resultMessage: String ?= null,
)
```

출처

- PUT의 개념, 예제 - https://developer.mozilla.org/ko/docs/Web/HTTP/Methods/PUT
- 멱등성 - https://developer.mozilla.org/ko/docs/Glossary/Idempotent
- PropertyNamingStrategies.SnakeCaseStrategy::class - https://zzang9ha.tistory.com/380
- 인프런: 스프링부트-코틀린 - https://www.inflearn.com/course/%EC%8A%A4%ED%94%84%EB%A7%81%EB%B6%80%ED%8A%B8-%EC%BD%94%ED%8B%80%EB%A6%B0