

Rest API PUTを実装する ～ @PutMapping

商品情報を更新する場合は、HTTP PUTを利用した、更新対象のIDを特定するAPIを作成します。Body部にjsonデータを設定してデータを送信します。Controllerクラスには、@PutMappingアノテーションでAPIのURLを設定します。また、@RequestBodyアノテーションでjson形式で受け取り、@PathVariableアノテーションで指定された更新キーを取得します。

PUT: 商品情報を更新

http://localhost:8080/items/**10001**

Request Body (json形式で設定)

```
{  
  "itemId": "10001",  
  "itemName": "サファイアリング",  
  "itemCategory": "ジュエリ"  
}
```

Controllerクラス

```
36  
37  
38 @PutMapping("/items/{itemId}")  
39 public void updateItem(@RequestBody Item item,  
40     @PathVariable("itemId") String itemId) {  
41     itemService.updateItem(itemId, item);  
42 }  
43
```

【@PutMapping】

- ✓ HTTP PUTリクエスト用のアノテーション
- ✓ value属性 (省略可)にURLを指定

実機演習の流れ

■ 前回の演習で作成したRest APIを拡張します。以下①～④の手順で演習を行います。

- ① サービスに商品更新処理を追加
- ② RestControllerクラスに商品更新Rest API (PUT)を追加
- ③ Clientからデータを更新
- ④ 更新された事をRest API GETを利用して確認

controller パッケージ

```
@RestController
ItemControllerクラス

@PutMapping("/items/{itemId}")
public void updateItem(@RequestBody
Item item,
@PathVariable("itemId") String itemId) {
itemService.updateItem(itemId, item);
}
```

service パッケージ

```
@Service
ItemServiceクラス

public void
updateItem(String
itemId, Item item)
```

Rest
Client

PUT: /items/10001

```
1 {
2   "itemId": "10001",
3   "itemName": "サファイアリング",
4   "itemCategory": "ジュエリ"
5 }
6
```

GET: /items

商品情報

use