REST API - Read Only

REST API

© luv2code LLC

REST API

- Spring Data REST will expose ALL these endpoints for free!

© luv2code LLC

REST API

- Spring Data REST will expose ALL these endpoints for free!

HTTP Method		CRUD Action
POST	/products	**C**reate a new product
GET	/products	**R**ead a list of products
GET	/products/{id}	**R**ead a single product
PUT	/products/{id}	**U**pdate an existing product
DELETE	/products/{id}	**D**elete an existing product

© luv2code LLC

REST API

I don't want that!
I want the REST API as READ-ONLY

- Spring Data REST will expose ALL these endpoints for free!

HTTP Method		CRUD Action
POST	/products	**C**reate a new product
GET	/products	**R**ead a list of products
GET	/products/{id}	**R**ead a single product
PUT	/products/{id}	**U**pdate an existing product
DELETE	/products/{id}	**D**elete an existing product

© luv2code LLC

REST API

I don't want that!
I want the REST API as READ-ONLY

- Spring Data REST will expose ALL these endpoints for free!

HTTP Method		CRUD Action
POST	/products	**C**reate a new product
GET	/products	**R**ead a list of products
GET	/products/{id}	**R**ead a single product
PUT	/products/{id}	**U**pdate an existing product
DELETE	/products/{id}	**D**elete an existing product

© luv2code LLC

REST API

I don't want that!
I want the REST API as READ-ONLY

- Spring Data REST will expose ALL these endpoints for free!

HTTP Method		CRUD Action
POST	/products	**C**reate a new product
GET	/products	**R**ead a list of products
GET	/products/{id}	**R**ead a single product
PUT	/products/{id}	**U**pdate an existing product
DELETE	/products/{id}	**D**elete an existing product

© luv2code LLC

REST API

I don't want that!
I want the REST API as READ-ONLY

- Spring Data REST will expose ALL these endpoints for free!

HTTP Method		CRUD Action
POST	/products	**C**reate a new product
GET	/products	**R**ead a list of products
GET	/products/{id}	**R**ead a single product
PUT	/products/{id}	**U**pdate an existing product
DELETE	/products/{id}	**D**elete an existing product

© luv2code LLC

REST API

I don't want that!
I want the REST API as READ-ONLY

- Spring Data REST will expose ALL these endpoints for free!

HTTP Method		CRUD Action
✗ POST	/products	**C**reate a new product
✓ GET	/products	**R**ead a list of products
✓ GET	/products/{id}	**R**ead a single product
✗ PUT	/products/{id}	**U**pdate an existing product
DELETE	/products/{id}	**D**elete an existing product

© luv2code LLC

REST API

I don't want that!
I want the REST API as READ-ONLY

- Spring Data REST will expose ALL these endpoints for free!

HTTP Method		CRUD Action
✗ POST	/products	**C**reate a new product
✓ GET	/products	**R**ead a list of products
✓ GET	/products/{id}	**R**ead a single product
✗ PUT	/products/{id}	**U**pdate an existing product
✗ DELETE	/products/{id}	**D**elete an existing product

© luv2code LLC

Possible Solutions

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST
    1. Manually create our own @RestController

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST
    1. Manually create our own @RestController
    2. Manually define methods for access: @GetMapping

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST
    1. Manually create our own @RestController
    2. Manually define methods for access: @GetMapping
    3. But we lose the Spring Data REST support for paging, sorting etc :-(

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST
    1. Manually create our own @RestController
    2. Manually define methods for access: @GetMapping
    3. But we lose the Spring Data REST support for paging, sorting etc :-(

2. Option 2: Use Spring Data REST

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST
    1. Manually create our own @RestController
    2. Manually define methods for access: @GetMapping
    3. But we lose the Spring Data REST support for paging, sorting etc :-(

2. Option 2: Use Spring Data REST
    1. Configure to disable certain HTTP methods: POST, DELETE etc...

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST
    1. Manually create our own @RestController
    2. Manually define methods for access: @GetMapping
    3. But we lose the Spring Data REST support for paging, sorting etc :-(

2. Option 2: Use Spring Data REST

Choose this one!

    1. Configure to disable certain HTTP methods: POST, DELETE etc...

© luv2code LLC

Possible Solutions

1. Option 1: Don't use Spring Data REST
    1. Manually create our own @RestController
    2. Manually define methods for access: @GetMapping
    3. But we lose the Spring Data REST support for paging, sorting etc :-(

2. Option 2: Use Spring Data REST

Choose this one!

    1. Configure to disable certain HTTP methods: POST, DELETE etc...

We will cover REST API Security later in the course

© luv2code LLC

Spring Data REST Configuration

© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

}
```


© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

From Spring Data REST

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

}
```


© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

    @Override
    public void configureRepositoryRestConfiguration(RepositoryRestConfiguration config) {

        HttpMethod[] theUnsupportedActions = {HttpMethod.PUT, HttpMethod.POST, HttpMethod.DELETE};

        // disable HTTP methods for Product:  PUT, POST and DELETE
        config.getExposureConfiguration()
                .forDomainType(Product.class)
                .withItemExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions))
                .withCollectionExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions));


    }
}
```


© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

HTTP methods to disable

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

    @Override
    public void configureRepositoryRestConfiguration(RepositoryRestConfiguration config) {

        HttpMethod[] theUnsupportedActions = {HttpMethod.PUT, HttpMethod.POST, HttpMethod.DELETE};

        // disable HTTP methods for Product:  PUT, POST and DELETE
        config.getExposureConfiguration()
                .forDomainType(Product.class)
                .withItemExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions))
                .withCollectionExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions));


    }
}
```


© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

    @Override
    public void configureRep                                    estConfiguration config) {

        HttpMethod[] theUnsupporte Actions = {HttpMethod.PUT, HttpMethod.POST, HttpMethod.DELETE};

        // disable HTTP methods for Product:  PUT, POST and DELETE
        config.getExposureConfiguration()
                .forDomainType(Product.class)
                .withItemExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions))
                .withCollectionExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions));


    }
}
```

Apply to Repo for: Product

© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

    @Override
    public void configureRep                                    estConfiguration config) {

        HttpMethod[] theUnsupporte Actions = {HttpMethod.PUT, HttpMethod.POST, HttpMethod.DELETE};

        // disable HTTP methods for Product:  PUT, POST and DELETE
        config.getExposureConfiguration()
                .forDomainType(Product.class)
                .withItemExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions))
                .withCollectionExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions));


    }
```

Apply to Repo for: Product

Single item

Collection

© luv2code LLC

Spring Data REST Configuration

- Disable HTTP methods: POST, PUT, DELETE

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

    @Override
    public void configureRepositoryRestConfiguration(RepositoryRestConfiguration config) {

        HttpMethod[] theUnsupportedActions = {HttpMethod.PUT, HttpMethod.POST, HttpMethod.DELETE};

        // disable HTTP methods for Product:  PUT, POST and DELETE
        config.getExposureConfiguration()
                .forDomainType(Product.class)
                .withItemExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions))
                .withCollectionExposure((metdata, httpMethods) -> httpMethods.disable(theUnsupportedActions));


    }
```

Single item

Collection

Disable

© luv2code LLC