Spring REST servisi

Sadržaj

- REST arhitekturalni stil
- Kreiranje REST kontrolera u Spring Boot aplikaciji
- Testiranje REST servisa
- Dokumentovanje REST servisa

Representational State Transfer (REST)

- Roy Fielding u svojoj disertaciji predstavlja REST kako “arhitekturalni stil”
- REST definiše skup ograničenja koja omogućavaju da korisnici servisa stupaju u interakciju sa sistemom koji pruža servis
- Najznačajnija REST ograničenja su:
  - klijent - server arhitektura
  - nema čuvanja stanja, što znači da naredni zahtev ne zavisi od podataka iz prethodnog
  - uniforman interfejs - svaki resurs ima svoj jedinstveni identifikator

REST principi

- **Resursi** se isporučuju preko lako razumljive strukture direktorijuma URI (Uniform Resource Identifier)
- **Reprezentacije** predstavljaju zahteve ili odgovore u JSON, XML i nekom drugom formatu
- **Poruke** koriste HTTP metode (GET, POST, PUT, DELETE)
- ***Stateless* interakcija** ne smešta na server kontekst klijenta, već klijent sam čuva svoje stanje sesije

Korišćenje HTTP metoda

- GET - čitanje

  ```
  GET /stores
  GET /stores/1
  ```

- POST - upis, šalje se reprezentacija objekta u telu zahteva

  ```
  POST /stores
  ```

- PUT - modifikacija, šalje se reprezentacija objekta u telu zahteva

  ```
  PUT /stores/1
  ```

- DELETE - brisanje

  ```
  DELETE /stores/1
  ```

HTTP kodovi statusa

- Opisuju rezultat izvršenja HTTP zahteva
- 1XX - informativni
- 2XX - uspešno izvršenje
- 3XX - preusmeravanje
- 4XX - greška na klijentskoj strani
- 5XX - greška na serverskoj strani

Sadržaj HTTP poruke

- HTTP poruka sadrži zaglavlje (engl. *header*) i telo (engl. *body*)
- Zaglavlje sadrži metapodatke poruke u obliku parova *key - value*
- Primeri metapodataka u zaglavlju:
  - Klijent može da postavi *key* `Accept` na vrednost `application/json` i time specificirati da zahteva odgovor od servera u JSON formatu
  - Kada šalje podatke, klijent može da postavi *key* `Content-Type` na vrednost `application/xml` i time ukazati da su podaci koje šalje zapisani u XML formatu

Spring REST kontroler (1)

```
@RestController
@RequestMapping("/stores")
public class TStoreController {

  @Autowired
  private TStoreService tStoreService;

  @GetMapping
  public List<TStore> findAll() {
    return tStoreService.findAll();
  }

  @GetMapping("/{storeId}")
  public TStore findStore(@PathVariable Integer storeId) {
    return tStoreService.findById(storeId);
  }
  …
```

Spring REST kontroler (2)

```
  @PostMapping
  public TStore insertStore(@RequestBody TStore tStore) {
    return tStoreService.insertStore(tStore);
  }

  @DeleteMapping("/{storeId}")
  public void deleteStore(@PathVariable Integer storeId) {
    tStoreService.deleteStore(storeId);
  }

  @PutMapping("/{storeId}")
  public TStore updateStore(@RequestBody TStore tStore, @PathVariable Integer storeId) {
    return tStoreService.updateStore(tStore, storeId);
  }
}
```

Vraćanje `ResponseEntity` objekta

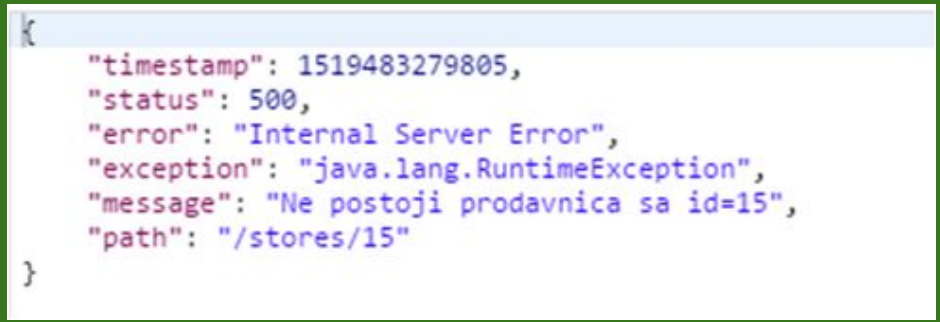
- Omogućava vraćanje detaljnih informacija o izvršenju operacije

```
@PostMapping
public ResponseEntity<Void> insert(@RequestBody TStore tStore) {
  if (tStoreService.exists(tStore.getId())) {
    return new ResponseEntity<Void>(HttpStatus.CONFLICT);
  } else {
    tStoreService.insertStore(tStore);
    return new ResponseEntity<Void>(HttpStatus.OK);
  }
}
```

Bacanje izuzetka

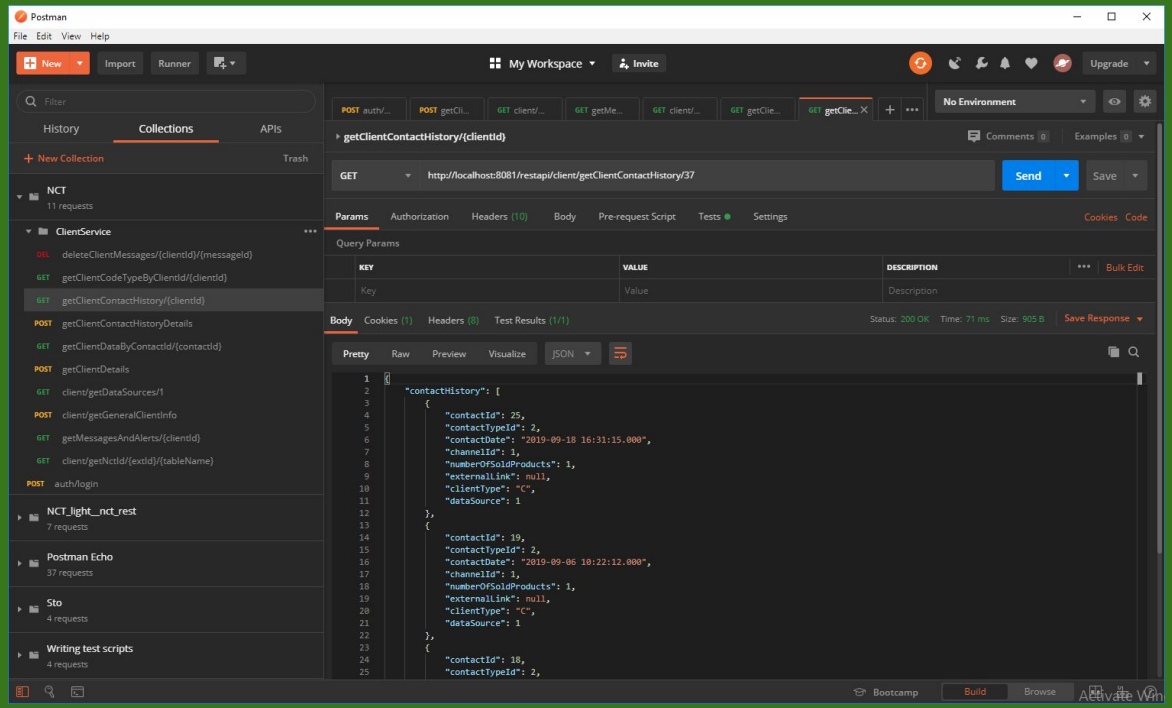
- Ukoliko se želi baciti izuzetak, ne treba da bude *checked exception*

```
@GetMapping("/{storeId}")
public TStore findStore(@PathVariable Integer storeId) {
  TStore tStore = storeService.findById(storeId);
  if (tStore == null) {
    throw new RuntimeException("Ne postoji prodavnica sa id=" + storeId);
  }
  return tStore;
}
```

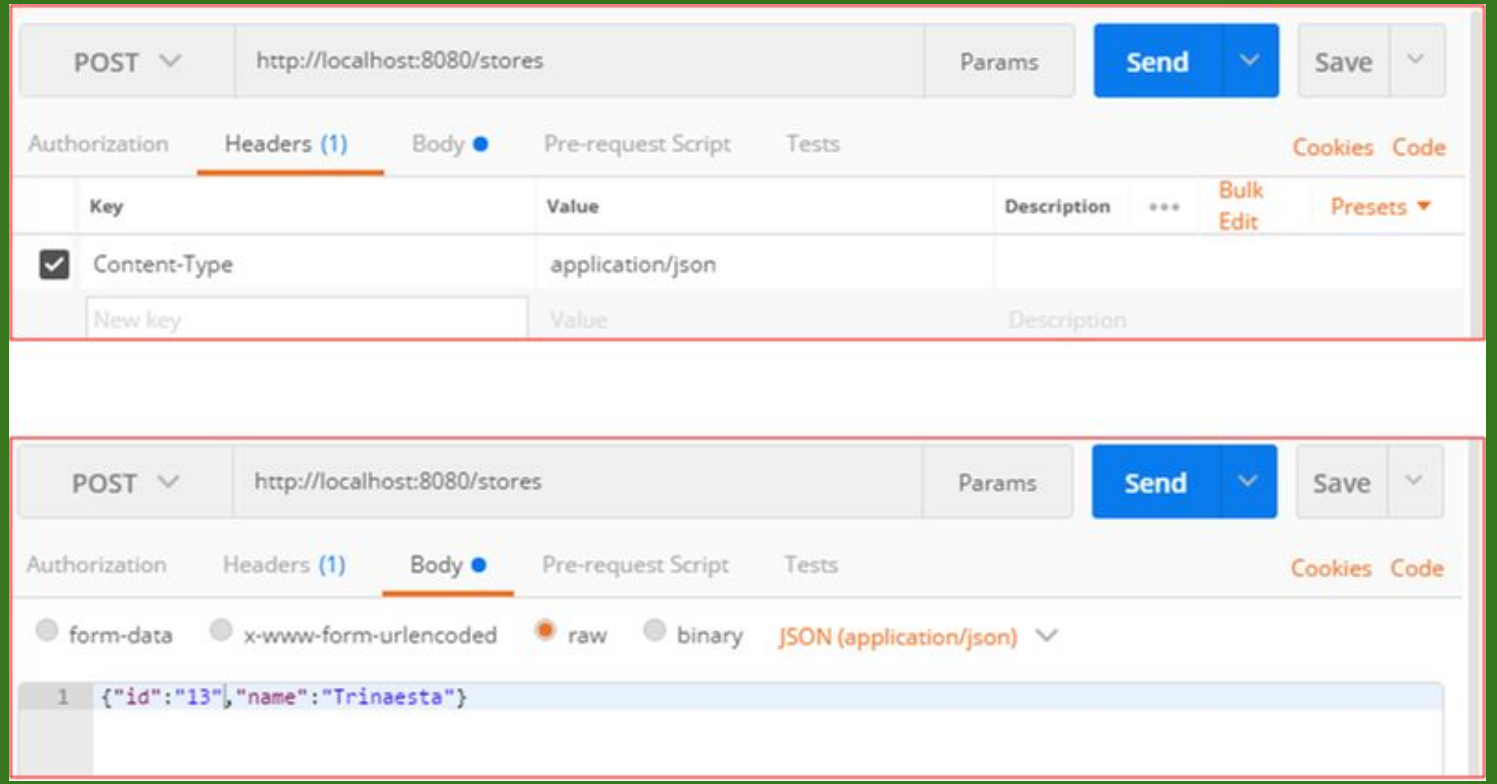


Testiranje REST servisa pomoću Postman-a

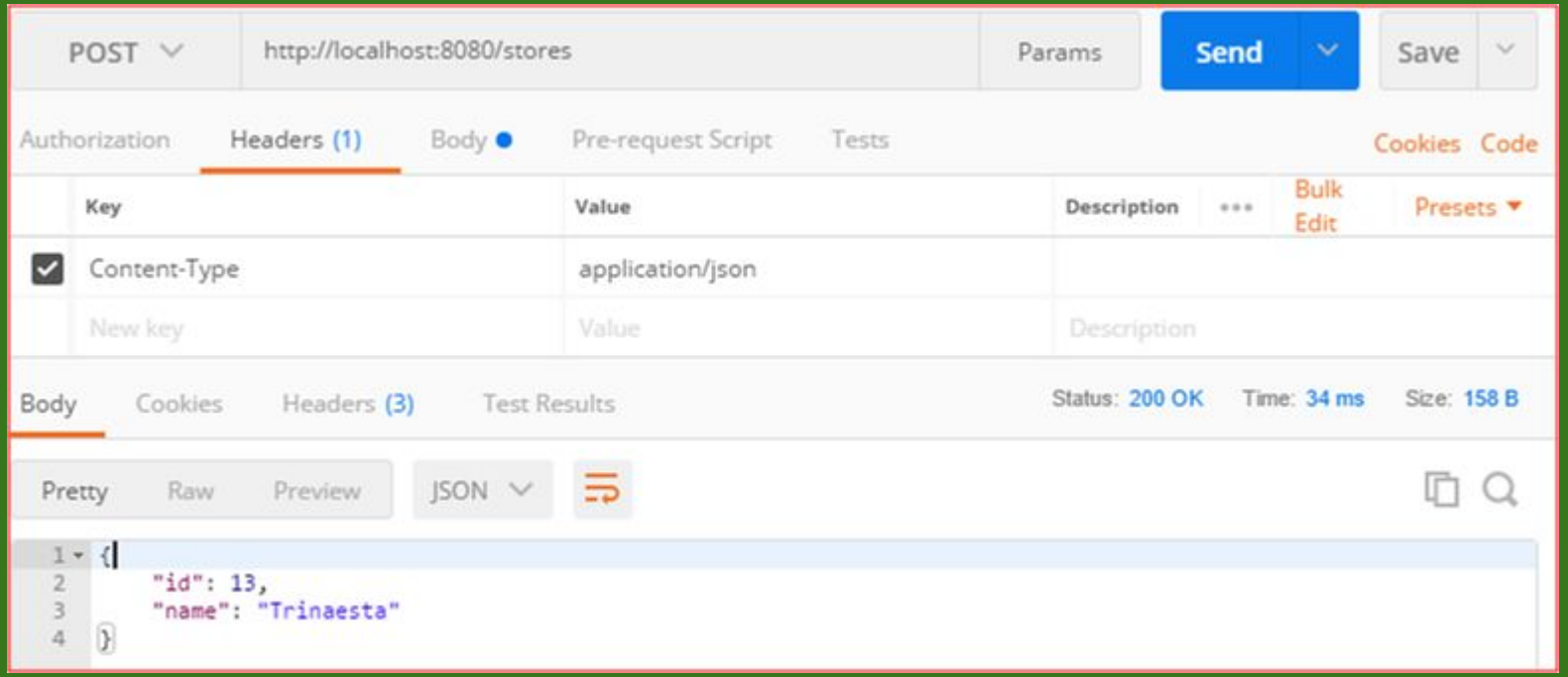
- Dostupan na https://www.postman.com/



Slanje POST zahteva



Odgovor na POST zahtev



Dokumentovanje REST servisa

- Swagger omogućava generisanje dokumentacije za REST servise
  - Format čitljiv za ljude i mašine
  - Lakši razvoj, otkiravnje i integracija
- Springfox Swagger zahteva sledeće zavisnosti:

```
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger2</artifactId>
  <version>2.9.0</version>
</dependency>
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger-ui</artifactId>
  <version>2.9.0</version>
</dependency>
```

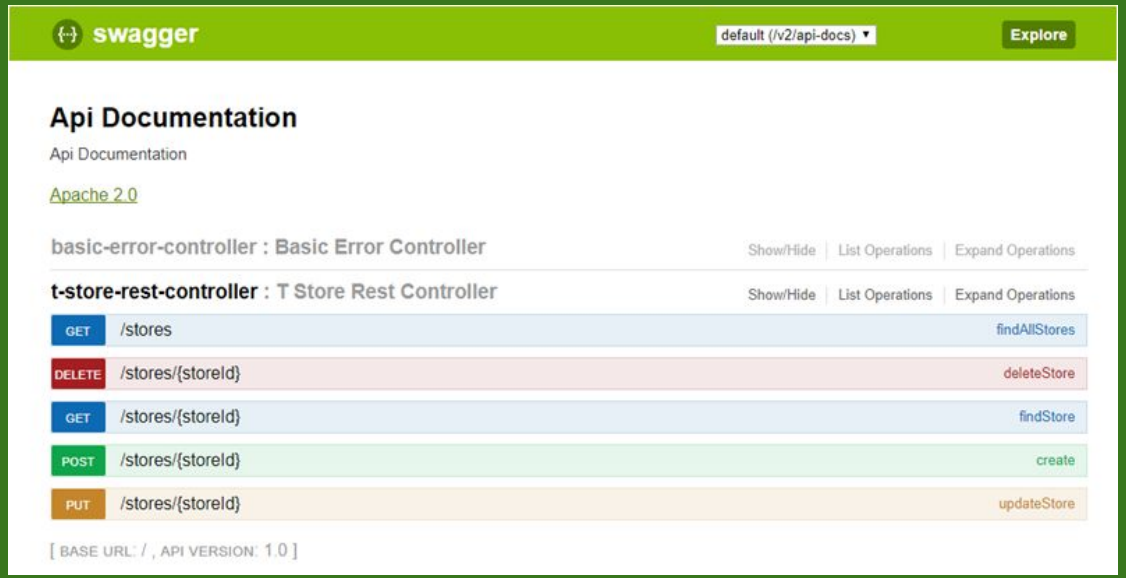
Konfiguracija aplikacije za Swagger

```
@Configuration
@EnableSwagger2
public class SwaggerConfig {

  @Bean
  public Docket api() {
    return new Docket(DocumentationType.SWAGGER_2)
      .select()
          .apis(RequestHandlerSelectors.any())
          .paths(PathSelectors.any())
          .build();
    }
  }
}
```

Prikaz Swagger dokumentacija

- localhost:8080/v2/api-docs prikazuje dokumentaciju kao JSON
- localhost:8080/swagger-ui.html obezbeđuje html pregled



Rezime

- REST servisi su tačka interakcija klijenta sa aplikacijom
- Postman omogućava testiranje REST servisa
- Swagger obezbeđuje dokumentovanje REST servisa