[illegible] [illegible] [illegible] का उपयोग

[illegible] [illegible] [illegible] एक बहुत ही अच्छा औजार है जो [illegible]—[illegible]-[illegible], [illegible]-[illegible] [illegible] [illegible]—को अपने [illegible] [illegible] में [illegible] करने के लिए है। यह [illegible] के साथ काम करने को सरल बनाता है, एक [illegible] [illegible]-[illegible] [illegible] और [illegible]-[illegible] [illegible] [illegible] पर [illegible] प्रदान करते [illegible].

1. [illegible] [illegible]

पहले, अपने [illegible] में [illegible] [illegible] [illegible] [illegible] [illegible] करें। अगर आप [illegible] [illegible] कर रहे हैं, तो `pom.xml` में यह [illegible] करें:

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-redis</artifactId>
</dependency>
```

[illegible] के लिए:

```
implementation 'org.springframework.boot:spring-boot-starter-data-redis'
```

यह [illegible] [illegible] (Jedis या [illegible]—[illegible] [illegible] है) [illegible] करता है। अगर आप [illegible] [illegible] करते हैं, तो आप [illegible] [illegible] कर सकते हैं और [illegible] [illegible] कर सकते हैं:

```
<dependency>
    <groupId>redis.clients</groupId>
    <artifactId>jedis</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-redis</artifactId>
    <exclusions>
        <exclusion>
            <groupId>io.lettuce</groupId>
            <artifactId>lettuce-core</artifactId>
        </exclusion>
    </exclusions>
</dependency>
```

2. [illegible] [illegible]

`application.properties` या `application.yml` में [illegible] [illegible] [illegible] करें। एक [illegible] [illegible] [illegible] के लिए, [illegible] [illegible] (6379) पर:

```
spring.redis.host=localhost
spring.redis.port=6379
spring.redis.password= # Optional,      Redis server   password
spring.redis.database=0 # Default database index
```

अगर आप □□□□□□□ □□□□□□ □□□□□□□ या □□□□ □□□□□□□□□□□□□ जैसे □□□□□□□□ □□□□ कर रहे हैं, तो □□□□□ और □□□□□□□□□□□□□ □□□□□□□ करें.

3. □□□□□□ □□□□□□ □□□□□ □□□□□□□□□□□□□□□□

□□□□□□□ □□□□□ □□□□□□ `RedisTemplate` □□□□□□□□ करता है □□□□-□□□□□□ □□□□□□□□□□□□□ के लिए। आप □□□□□ □□□□□□□□□ कर सकते हैं □□□□□□□□ या □□□□□□□□□□□ में:

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.redis.core.RedisTemplate;
import org.springframework.stereotype.Service;

@Service
public class RedisService {
    private final RedisTemplate<String, String> redisTemplate;

    @Autowired
    public RedisService(RedisTemplate<String, String> redisTemplate) {
        this.redisTemplate = redisTemplate;
    }

    public void saveData(String key, String value) {
        redisTemplate.opsForValue().set(key, value);
    }

    public String getData(String key) {
        return redisTemplate.opsForValue().get(key);
    }
}
```

- `RedisTemplate` □□□□□□□□ है: `<String, String>` □□□□□□ □□□□□ और □□□□□□□ □□□□□□□□ हैं। आप □□□□□□ □□□□□□ (□.□., `<String, Object>`) □□□ कर सकते हैं।
- `opsForValue()` □□□□□□□ □□□□-□□□□□□ □□□□□□□□□□□□ के लिए है। अन्य □□□□□□□□ □□□□□□□□ `opsForList()`, `opsForSet()`, `opsForHash()`, □□□., □□□□□□□□□□ □□□□□□ □□□□□ □□□□□□□□□□□□ के लिए.

4. [illegible]

[illegible] [illegible] को [illegible] और [illegible] करने के [illegible], `RedisTemplate` के [illegible] [illegible] के [illegible] [illegible] करें। [illegible] [illegible]-[illegible] के, [illegible] आप [illegible] [illegible] [illegible] [illegible] [illegible]:

```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.data.redis.connection.RedisConnectionFactory;
import org.springframework.data.redis.core.RedisTemplate;
import org.springframework.data.redis.serializer.Jackson2JsonRedisSerializer;
import org.springframework.data.redis.serializer.StringRedisSerializer;

@Configuration
public class RedisConfig {
    @Bean
    public RedisTemplate<String, Object> redisTemplate(RedisConnectionFactory connectionFactory) {
        RedisTemplate<String, Object> template = new RedisTemplate<>();
        template.setConnectionFactory(connectionFactory);
        template.setKeySerializer(new StringRedisSerializer());
        template.setValueSerializer(new Jackson2JsonRedisSerializer<>(Object.class));
        template.afterPropertiesSet();
        return template;
    }
}
```

अब आप [illegible] के [illegible] [illegible] [illegible] [illegible] [illegible]:

```
public class Person {
    private String firstName;
    private String lastName;

    // Default constructor (for deserialization)
    public Person() {}

    public Person(String firstName, String lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
    }

    // Getters and setters
    public String getFirstName() { return firstName; }
```

```
    public void setFirstName(String firstName) { this.firstName = firstName; }
    public String getLastName() { return lastName; }
    public void setLastName(String lastName) { this.lastName = lastName; }
}

@Service
public class PersonRedisService {
    private final RedisTemplate<String, Object> redisTemplate;

    @Autowired
    public PersonRedisService(RedisTemplate<String, Object> redisTemplate) {
        this.redisTemplate = redisTemplate;
    }

    public void savePerson(String key, Person person) {
        redisTemplate.opsForValue().set(key, person);
    }

    public Person getPerson(String key) {
        return (Person) redisTemplate.opsForValue().get(key);
    }
}
```

5. [illegible]

एक [illegible], [illegible] [illegible] [illegible]। [illegible] [illegible]:

```
import org.springframework.data.annotation.Id;
import org.springframework.data.redis.core.RedisHash;

@RedisHash("Person") // Maps to a Redis hash with prefix "Person"
public class Person {
    @Id
    private String id; // Redis key will be "Person:<id>"
    private String firstName;
    private String lastName;

    // Constructors, getters, setters (as above)
}
```

```
import org.springframework.data.repository.CrudRepository;

public interface PersonRepository extends CrudRepository<Person, String> {
}
```

□□□□□ □□□□ □□□□□□□□:

```
@Service
public class PersonService {
    private final PersonRepository repository;

    @Autowired
    public PersonService(PersonRepository repository) {
        this.repository = repository;
    }

    public void savePerson() {
        Person person = new Person("John", "Doe");
        repository.save(person);
        System.out.println("Saved person with ID: " + person.getId());
    }

    public void findPerson(String id) {
        Person person = repository.findById(id).orElse(null);
        if (person != null) {
            System.out.println(person.getFirstName() + " " + person.getLastName());
        }
    }
}
```

- □ `@RedisHash` □□□□□□□ □□ □□ □□□□□□ □□□□□ □□ □□□□□ □□□□ □□□□□□ □□□□.
- □ □□□□□□□□□□□□ □□□□□ □□□□□□□□□□□□□ □□□□ □□ □□□□ □□□□ □□□□□□□□□ □□□□□□ □□□□.

6. □□□□ □□□□□ □□□□□□□□□□□□□□□

□□□□□□□ □□□□□□ □□□□□□□□□ □□□□□□□□□ □□□□ (□.□., □□□□ □□□□□□□□: `docker run -d -p 6379:6379 redis`) □□ □□□□□□□□□ □□□□□□□□ □□ □□□□□□□ □□□□□□□□□□ □□ □□□□□. □□□□□ □□□□□□ □□□□□□□ □□□□□□ □□□□□□□ □□□□□ □□□□:

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
@SpringBootApplication
public class Application {
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

7. □□□□□□□□□ □□□□□□□□□

- □ □□□ **(**□□□□-□□-□□□□**):** □□□□ □□□ □□□□□□□□□□ □□□ □□□□□□ `redisTemplate.expire(key, 10, TimeUnit.MINUTES)` □□ □□□□.
- □ □□□□□□**:** `opsForHash()` □□□ □□□□□□ □□□□□ □□□□□□ □□ □□□□ □□□□□□□□ □□□□ □□□□□ □□ □□□□.
- □ □□□/**/**□□□**:** `RedisMessageListenerContainer` □□□ □□□□□□ □□□□□□□□□ □□ □□□□.

□□□□□□□ □□□□□□□□

1. `Person` □□□□□□ □□ `repository.save()` □□ `redisTemplate.opsForValue().set()` □□ □□□□ □□□□ □□□□□□.
2. □□□□ □□□□□□□□ □□□□□□ `repository.findById()` □□ `redisTemplate.opsForValue().get()` □□ □□□□.
3. □□□□ □□□□□ □□□□□□□□ □□ □□□□□ □□□□□□ (□.□., `redis-cli` □□ □□□□) □□□□ □□□□□□ □□ □□□□.

□□□ □□□□□□□□□□ □□□□□ □□□□□ □□□! □□□□ □□□ □□□□□□□, □□□/□□□, □□ □□□□ □□□□□□□□ □□□□□□□ □□□□□ □□□□□□□ □□□□, □□□ □□□□□□□ □□ □□□ □□□ □□□ □□□ □□□□ □□□□, □□□ □□□□ □□□□□□ □□□□ □□□□□ □□□□. □□□ □□□□□ □□□□ □□□ □□□ □□□□ □□□?