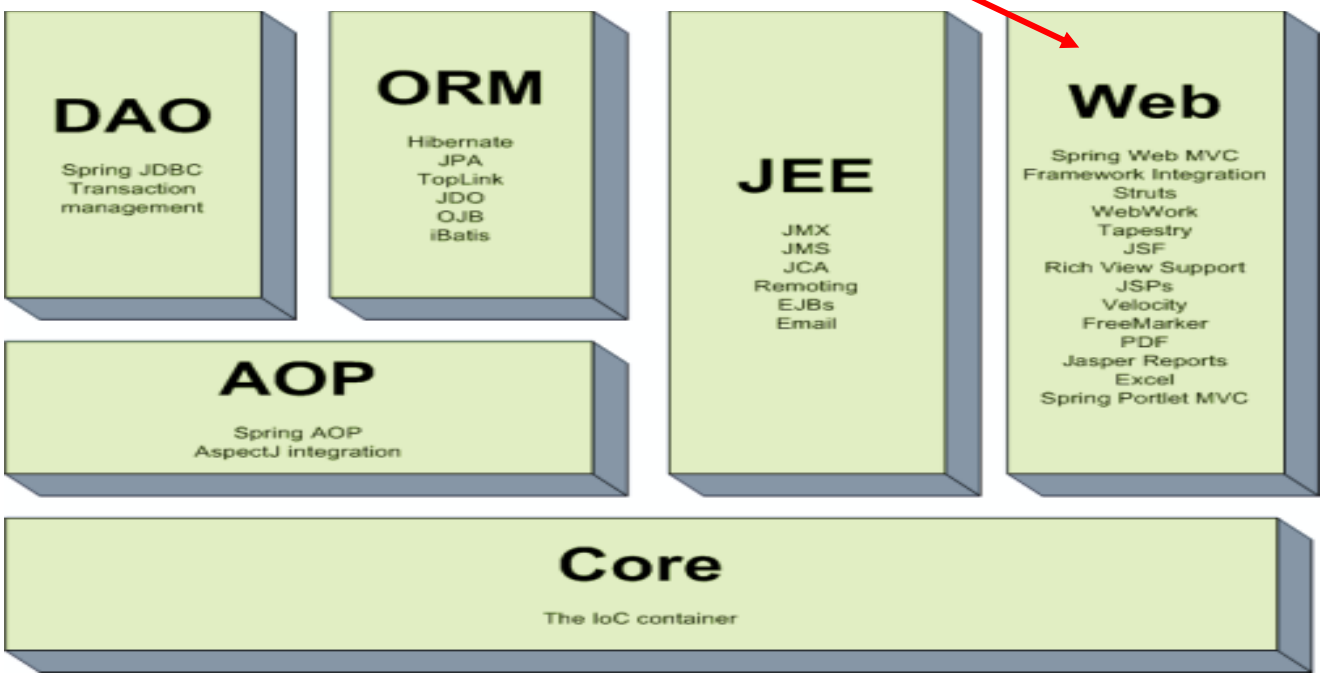
Rest Template

Ci sono molti modi per implementare client Rest in java e uno dei tanti è:

➔ Usare la classe RestTemplate di Spring Framework

Per chi non conosce questa libreria: RestTemplate è un client Rest offerto dal modulo di Spring-web di Spring Framework il quale mette a disposizione classi (come RestTemplate) e metodi per “consumare” dati Rest.



1.0 OTTENERE I DATI DA UN WEB SERVICE REST

In Questo paragrafo vedremo come utilizzare i RestTemplate per ottenere dati utilizzando API di un web server e utilizzando diversi verbi del protocollo HTTP come GET, POST, PUT, ecc.. e per riuscire nel nostro intento andremo avanti per step successivi.

- STEP_1:

  Iniziamo con l'importare il modulo di Spring Framework che ci mette a disposizione la classe RestTemplate.

  Per fare questo importiamo la seguente dipendenza nel file pom.xml:

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-web</artifactId>
</dependency>
```

NOTA BENE:

Se stai usando la dipendenza di springboot il cui artifcatid si chiama

➔ Spring-boot-starter-web

Allora non è necessario importate il modulo spring-web di prima in quanto questo è già contenuto in spring-boot-starter-web.

2.0 CLIENT REST DI UNA API CON VEBO GET DI http

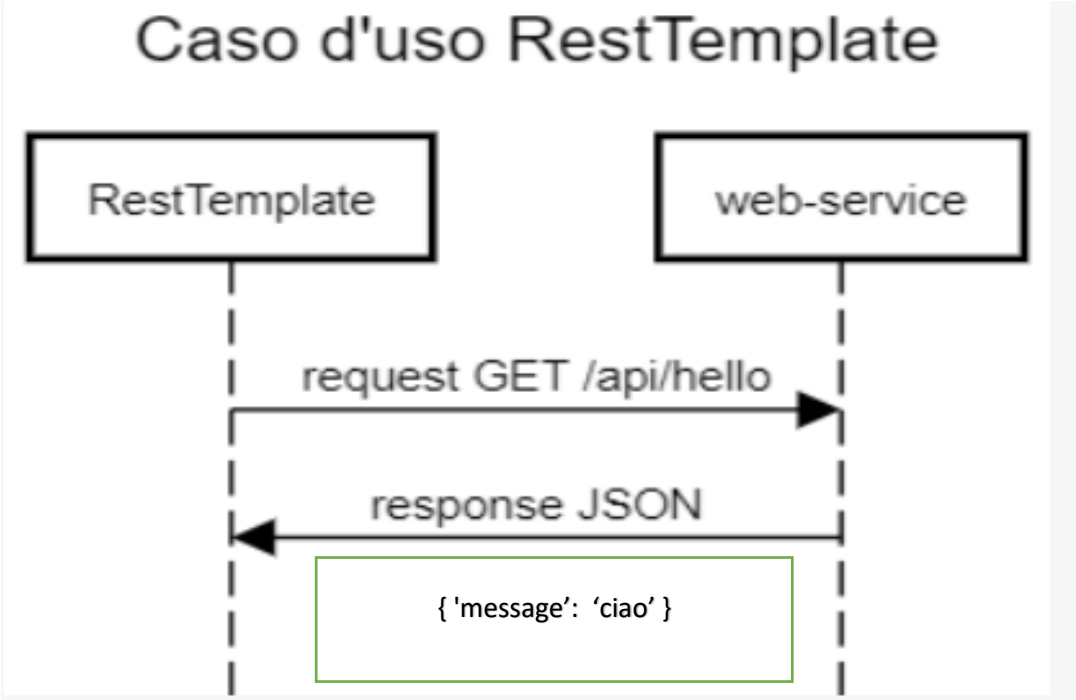
- STEP_2:

  Partiamo implementando un clien rest che effettua una richiesta GET al seguente URL: ‘api/hello’

  Il quale risponde per puro scopo di test con una risorsa di tipo HelloMessage con il seguente JSON:

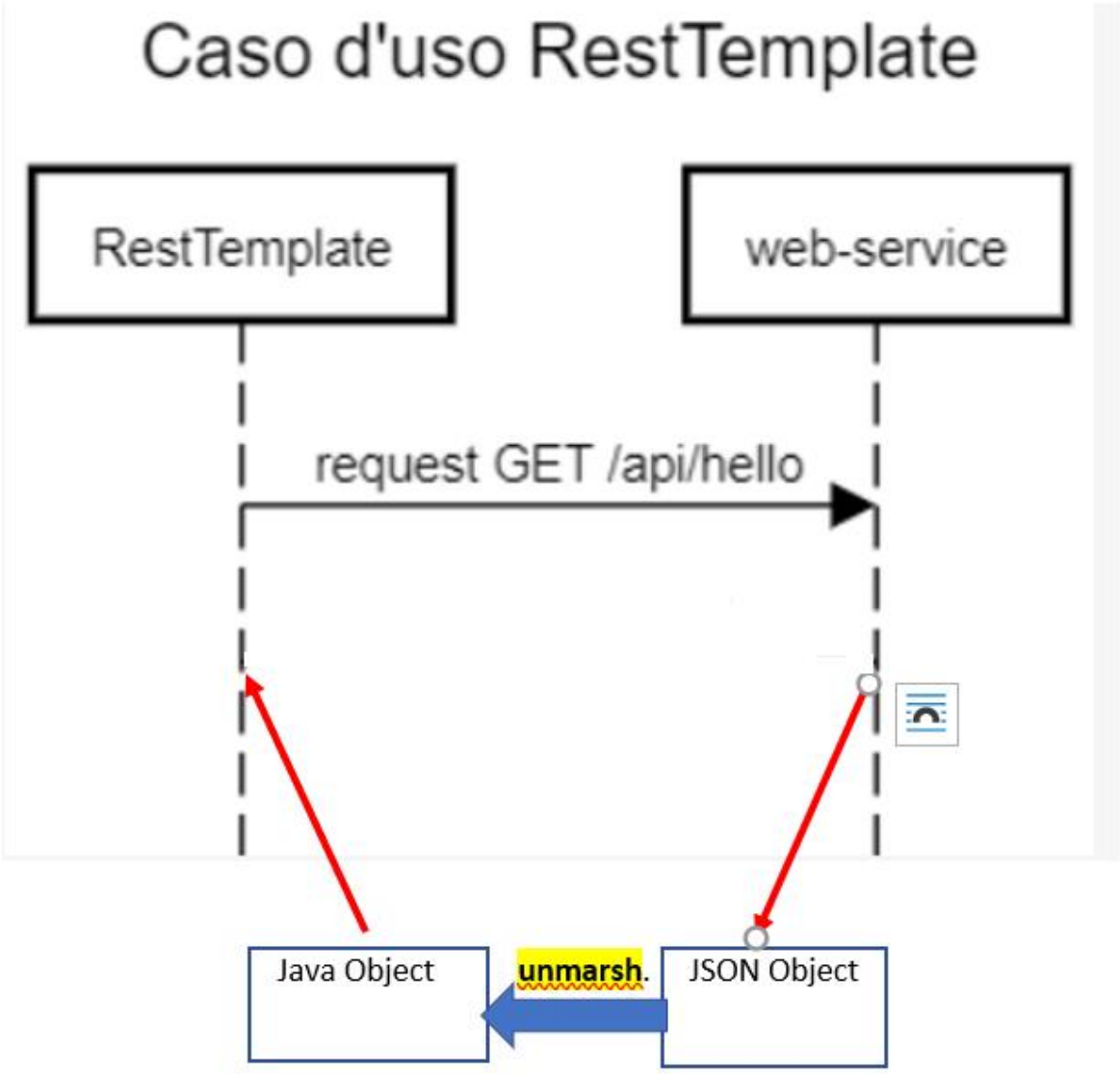
```
{
    "message": "Ciao"
}
```

RestTemplate ci permette di effettuare la richiesta al web service (servizio web) e di convertire automaticamente il JSON di risposta in un oggetto Java.



Quando il web service invia la risposta al nostro applicativo Java (RestTemplate) , in verità non arriva il JSON vero e proprio ma interviene la libreria Jackson di che è contenuta sempre dentro al modulo Spring-web che prende il JSON dalla risposta e lo trasforma in un oggetto Java. Tale processo da JSON a oggetto Java prende il nome di 'deserializzazione' o 'unmarshaling'.

Vicevarsa da da oggetto Java a JSON prende il nome di ‘serializzazioe’ o ‘marshaling’.



RestTemplate ti permette di effettuare la richiesta e di convertire automaticamente il JSON ricevuto in un oggetto Java strutturato in modo equivalente al JSON atteso;

ovvero occorre in tal caso creare una classe Java che abbia tutti i campi previsti dal JSON. Nel nostro esempio ho creato una classe HelloMessage.java la quale mi rappresenta la risposta dal web service.

```java
public class HelloMessage {

    private String message;

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    public HelloMessage(String message){
        this.message = message;
    }

    public HelloMessage(){}

    @Override
    public String toString() {
        return "HelloMessage{" +
                "message='" + message + '\'' +
                '}';
    }
}
```

Notiamo che questa classe contiene come campi di istanza gli stessi del JSON di risposta del web-service (ovvero message).

Ci penserà poi la libreria Jackson a deserializzare il Json di risposta e creare una istanza dell'oggetto Java di tipo HelloMessage con stato uguale ai campi del JSON.

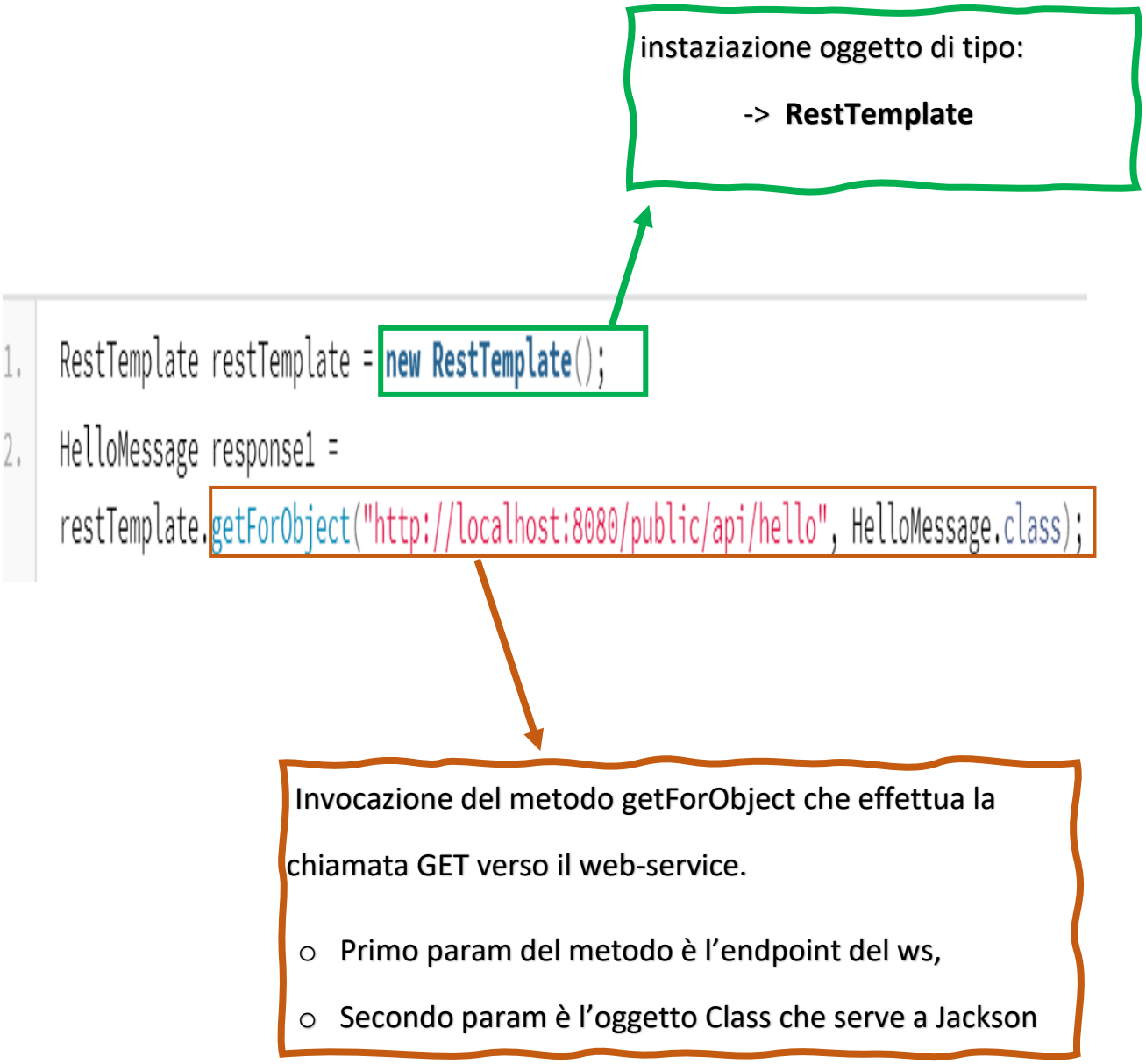
NOTA BENE BENE:

Non dimentichiamo di includere nella classe HelloMessage il costruttore senza parametri, in quanto è utilizzato da Jackson per istanziare l'oggetto.

Se questo costruttore senza parametri non è presente verrà lanciato un errore a runtime proprio da Jackson library.

- STEP_3:

  A questo punto per ottenere dal web-service la risposta bastano poche righe di codice:



  Due istruzioni ed il gioco è fatto: nella variabile response1 verrà salvata la risposta del web-service non come oggetto JSON ma come oggetto Java.

  Per questo abbiamo passato come secondo parametro del metodo getForObject la class Class di HelloMessage, pre fare in modo che tramite la

reflection Jackson instanzi un oggetto di Tipo HelloMessage equivalente al JSON di risposta del web-service.