## Tại sao phải sử dụng WebClient?

WebClient được sinh ra để thay thế cho RestTemplate thì đương nhiên nó phải tốt hơn rồi. Vậy nhũng cải tiến của WebClient là gì?

## Cơ chế Blocking và Non-Blocking

### Blocking

* RestTemplate sử dụng Java Servlet API, dựa trên mô hình thread cho mỗi request. Mỗi thread sẽ chặn cho tới khi nào web client nhận được phản hồi. Khi có số lượng request lớn sẽ làm số lượng thread tăng lên và hao tốn bộ nhớ, giảm performance.

### Non-Blocking

* WebClient được cải tiến bằng cách sử dụng cơ chế không đồng bộ được cung cấp bởi Spring Reactive framework.
* Webclient sẽ tạo ra mỗi task cho 1 sự kiện(request), Spring Reactive hỗ trợ sắp xếp các task đó và chỉ thực thi chúng khi nhận được response thích hợp.
* Reactive framework sử dụng cơ chế *event-driven architecture.* Nó cung cấp những cơ chế không đồng bộ thông qua *Reactive Streams API.*

## So sánh qua Example

Đầu tiên cài đặt thư viện và viết phương thức *getSlowServiceUri()*

*@GetMapping("/slow-service-datas")*

*private List<Entity>* ***getSlowServiceUri****() {*

*Thread.sleep(2000L); // delay*

*return Arrays.asList(*

*new Entity("RestTemplate rules", "@user1"),*

*new Entity("WebClient is better", "@user2"),*

*new Entity("OK, both are useful", "@user1"));*

*}*

### Sử dụng *RestTemplate*

*@GetMapping("/datas-blocking")*

*public List<Entity>* ***getDatasBlocking****() {*

*log.info("Starting BLOCKING Controller!");*

*final String uri = getSlowServiceUri();*

*RestTemplate restTemplate = new RestTemplate();*

*ResponseEntity<List<Entity>> response = restTemplate.exchange(*

*uri, HttpMethod.GET, null,*

*new ParameterizedTypeReference<List<Entity>>(){});*

*List<Entity> result = response.getBody();*

*result.forEach(entity -> log.info(entity.toString()));*

*log.info("Exiting BLOCKING Controller!");*

*return result;*

*}*

**Kết quả:**

*Starting BLOCKING Controller!*

*Entity(text=RestTemplate rules, username=@user1)*

*Entity(text=WebClient is better, username=@user2)*

*Entity(text=OK, both are useful, username=@user1)*

*Exiting BLOCKING Controller!*

### Sử dụng *WebClient*

*@GetMapping(value = "/datas-non-blocking",*

*produces = MediaType.TEXT\_EVENT\_STREAM\_VALUE)*

*public Flux<Tweet>* ***getDatasNonBlocking****() {*

*log.info("Starting NON-BLOCKING Controller!");*

*Flux<Entity> dataFlux = WebClient.create()*

*.get()*

*.uri(getSlowServiceUri())*

*.retrieve()*

*.bodyToFlux(Entity.class);*

*dataFlux.subscribe(entity -> log.info(entity.toString()));*

*log.info("Exiting NON-BLOCKING Controller!");*

*return dataFlux;*

*}*

**Kết quả:**

*Starting NON-BLOCKING Controller!*

*Exiting NON-BLOCKING Controller!*

*Entity(text=RestTemplate rules, username=@user1)*

*Entity(text=WebClient is better, username=@user2)*

*Entity(text=OK, both are useful, username=@user1)*

*Ref:* [*https://www.baeldung.com/spring-webclient-resttemplate*](https://www.baeldung.com/spring-webclient-resttemplate)