

# webclient [] [] reactive [][][]

- [] []
  - doOnError() : [] [] [], [] [] [] [] []
    - [] [], [] [] [] [] [] [] [] []
    - 2xx [] [] [] Exception [] [] call []
  - doOnSuccess() : [] [] [], [] []
  - onErrorReturn : [] [] [] [] [] Return []
    - [] [] [] [] [] [] [] [] [] []
  - onErrorResume : [] [] [] [] [] Flux[] Return []
    - [] [] [] [] [] [] [] [] [] []
  - onErrorContinue : [] [] [] [] [] [] skip[] [] [].
    - [] [] [] [] [] [] [] [] [] []

```
public Mono<Object> doExec(
    ServerHttpRequest request, ServerHttpResponse
    clientResponse, RbcAuth rbcAuth) {

    String finalAuthToken = ...;

    try{
        return webClient.get()
            .uri(rbcUrl.getCategories())
            .accept(MediaType.APPLICATION_JSON)
            .headers(httpHeaders ->
httpHeaders.setBearerAuth(finalAuthToken))
            .retrieve()
            .onStatus(HttpStatus::isError,
res -> doError(res))
            .bodyToMono(CustomResult.class)
            .doOnError(ex ->
doExcept(clientResponse, ex))
            .flatMap(res ->
doSucc(clientResponse, res))
            .doOnSuccess(res ->
doFinish(clientResponse, res))
            .subscribe()
            .log()
    }
```

```

        ;
    }catch (Exception e){
        log.error("webClient fail. e={}",
e.getMessage(), e);
        return doExcept(clientResponse, e);
    }
}

private Mono<Object> doFinish(ServerHttpResponse response,
CustomResult res) {
    ...
    return Mono.just("finish");
}

private Mono<Object> doSucc(ServerHttpResponse response,
CustomResult res) {
    ...
    return Mono.just("success");
}

private Mono<Object> doExcept(ServerHttpResponse response,
Throwable ex) {
    ...
    return Mono.just("fail. e={}", e.getMessage());
}

private Mono<Exception> doError(ClientResponse res) {
    if (res.statusCode().isError()) {
        return Mono.error(new
RuntimeException(res.toString()));
    }
    else{
        return Mono.error(new
RuntimeException(res.toString()));
    }
}
}

```