The QuoteServices App Case Study: Structure & Functionality of Client Classes

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in this Part of the Lesson

• Understand the QuoteClient structure & functionality that uses HTTP interface to send/receive HTTP GET/POST requests/responses to/from QuoteServices microservices

See [github.com/douglascraigschmidt/LiveLessons/tree/master/WebFlux/ex3](https://github.com/douglascraigschmidt/LiveLessons/tree/master/WebFlux/ex3)
The Structure & Functionality of QuoteClient Classes
The Structure & Functionality of QuoteClient Classes

- The QuoteClient classes performs asynchronous remote method invocations on the API Gateway microservice to obtain Handey & Zippy quotes

```java
@Component
public class QuoteClient {
  @Autowired HandeyQuoteAPI mHandleQuoteAPI;

  @Autowired ZippyQuoteAPI mZippyQuoteAPI;
  ...

  public Flux<Quote> postQuotes(String routename, List<Integer> quoteIds) {
    return routename.equals(HANDEY) ? mHandleQuoteAPI.postQuotes(quoteIds) : mZippyQuoteAPI.postQuotes(quoteIds);
  }
  ...
}
```

The Structure & Functionality of QuoteClient Classes

- The QuoteClient classes performs asynchronous remote method invocations on the API Gateway microservice to obtain Handey & Zippy quotes

```java
@Component
public class QuoteClient {
    @Autowired HandeyQuoteAPI mHandleQuoteAPI;
    @Autowired ZippyQuoteAPI mZippyQuoteAPI;

    public Flux<Quote> postQuotes(String routename, List<Integer> quoteIds) {
        return routename.equals(HANDEY)
            ? mHandeyQuoteAPI.postQuotes(quoteIds)
            : mZippyQuoteAPI.postQuotes(quoteIds);
    }
}
```

Enables auto-detection & wiring of dependent implementation classes via classpath scanning

See www.baeldung.com/spring-component-repository-service
The Structure & Functionality of QuoteClient Classes

- The QuoteClient classes performs asynchronous remote method invocations on the API Gateway microservice to obtain Handey & Zippy quotes.

```java
@Component
public class QuoteClient {
    @Autowired HandeyQuoteAPI mHandleQuoteAPI;
    @Autowired ZippyQuoteAPI mZippyQuoteAPI;
    ...
    public Flux<Quote> postQuotes(String routename, List<Integer> quoteIds) {
        return routename.equals(HANDEY)  
            ? mHandeyQuoteAPI.postQuotes(quoteIds)  
            : mZippyQuoteAPI.postQuotes(quoteIds);
    }
    ...
}
```

See [www.baeldung.com/spring-autowire](http://www.baeldung.com/spring-autowire)
The Structure & Functionality of QuoteClient Classes

- The QuoteClient classes performs asynchronous remote method invocations on the API Gateway microservice to obtain Handey & Zippy quotes.

```java
@Component
public class QuoteClient {
  @Autowired HandeyQuoteAPI mHandleQuoteAPI;

  @Autowired ZippyQuoteAPI mZippyQuoteAPI;

  ...

  public Flux<Quote> postQuotes(String routename, List<Integer> quoteIds) {
    return routename.equals(HANDEY) ? mHandeyQuoteAPI.postQuotes(quoteIds) : mZippyQuoteAPI.postQuotes(quoteIds);
  }

  ...
}
```

Get a List of Quotes based on their quote Ids from a designated microservice.
The Structure & Functionality of the QuoteAPI Interface
The Structure & Functionality of the *QuoteAPI Classes

- The *QuoteAPI classes uses the Spring 6 HTTP interface to abstract details of making async remote method invocations using HTTP

```java
@HttpExchange(url = ZIPPY + "/")
public interface ZippyQuoteAPI {
    @GetExchange(GET_ALL_QUOTES)
    Flux<Quote> getAllQuotes();

    @PostExchange(POST_QUOTES)
    Flux<Quote> postQuotes(@RequestBody List<Integer> quoteIds);

    @PostExchange(POST_SEARCHES)
    Flux<Quote> search(@RequestBody List<String> queries);

    ...
}
```

See WebFlux/ex3/client/src/main/java/edu/vandy/quoteservices/client/ZippyQuoteAPI.java
The Structure & Functionality of the *QuoteAPI Classes

• The *QuoteAPI classes uses the Spring 6 HTTP interface to abstract details of making async remote method invocations using HTTP

@HttpExchange(url = ZIPPY + "/")
public interface ZippyQuoteAPI {
    @GetExchange(GET_ALL_QUOTES)
    Flux<Quote> getAllQuotes();

    @PostExchange(POST_QUOTES)
    Flux<Quote> postQuotes(@RequestBody List<Integer> quoteIds);

    @PostExchange(POST_SEARCHES)
    Flux<Quote> search(@RequestBody List<String> queries);

    ...

These proxy methods mimic the signature of controller methods
The Structure & Functionality of the *QuoteAPI Classes

• The *QuoteAPI classes uses the Spring 6 HTTP interface to abstract details of making async remote method invocations using HTTP

```java
@HttpExchange(url = ZIPPY + "/")
public interface ZippyQuoteAPI {
    @GetExchange(GET_ALL_QUOTES)
    Flux<Quote> getAllQuotes();

    @PostExchange(POST_QUOTES)
    Flux<Quote> postQuotes(@RequestBody List<Integer> quoteIds);

    @PostExchange(POST_SEARCHES)
    Flux<Quote> search(@RequestBody List<String> queries);

    ...
}
```

Even the annotations are the same as the Spring controller annotations!

There's a similar HTTP interface that defines the HandeyQuoteAPI
The Structure & Functionality of the ClientBeans Class
The Structure & Functionality of the ClientBeans Class

- The ClientBeans class contains factory method beans that create the Zippy QuoteAPI & HandeyQuoteAPI proxies

```java
@Component
public class ClientBeans {

    @Bean
    public ZippyQuoteAPI getZippyQuoteAPI() {
        var webClient = WebClient.builder()
            .baseUrl(GATEWAY_BASE_URL).build();

        return HttpServiceProxyFactory
            .builder(WebClientAdapter
                .forClient(webClient))
            .build()
            .createClient(ZippyQuoteAPI.class);

    }

    ...

This @Bean annotation can be injected into classes using Spring’s @Autowired annotation

See WebFlux/ex3/client/src/main/java/edu/vandy/quoteservices/common/ClientBeans.java
```
The Structure & Functionality of the QuoteProxy Class

- The ClientBeans class contains factory method beans that create the ZippyQuoteAPI & HandeyQuoteAPI proxies

@Component
public class ClientBeans {
    @Bean
    public ZippyQuoteAPI getZippyQuoteAPI() {
        var webClient = WebClient.builder()
            .baseUrl(GATEWAY_BASE_URL).build();

        return HttpServiceProxyFactory
            .builder(WebClientAdapter
                .forClient(webClient))
            .build()
            .createClient(ZippyQuoteAPI.class);

    }

    ...

Create a ZippyQuoteAPI HTTP client that’s used to make HTTP requests to the GatewayApplication microservice

There’s a similar factory method bean that creates a HandeyQuoteAPI instance
The Structure & Functionality of the QuoteProxy Class

• The ClientBeans class contains factory method beans that create the Zippy QuoteAPI & HandeyQuoteAPI proxies

```java
@Component
public class ClientBeans {
    @Bean
    public ZippyQuoteAPI getZippyQuoteAPI() {
        var webClient = WebClient.builder()
            .baseUrl(GATEWAY_BASE_URL).build();

        return HttpServiceProxyFactory
            .builder(WebClientAdapter
                .forClient(webClient))
            .build()
            .createClient(ZippyQuoteAPI.class);
    }

    ...
}
```
The Structure & Functionality of the QuoteProxy Class

- The ClientBeans class contains factory method beans that create the ZippyQuoteAPI & HandeyQuoteAPI proxies

```java
@Component
public class ClientBeans {
    @Bean
    public ZippyQuoteAPI getZippyQuoteAPI() {
        var webClient = WebClient.builder()
            .baseUrl(GATEWAY_BASE_URL).build();

        return HttpServiceProxyFactory
            .builder(WebClientAdapter
                .forClient(webClient))
            .build()
            .createClient(ZippyQuoteAPI.class);
    }
}
```

Return a new instance of ZippyQuoteAPI that makes HTTP requests via WebClient.
End of the QuoteServices App Case Study: Structure & Functionality of Client Classes