Preparing for code generation

OpenAPI & Client Code Generation in Elements

Elements supports **automatic client code generation** using **OpenAPI**. This allows you to expose your Element's API in a standardized format (OAS3 / Swagger), and then generate client libraries for different languages (Java, C#, TypeScript, etc.) without writing them by hand.

What is OpenAPI?

OpenAPI is a standard specification for describing REST APIs.
When your Element exposes endpoints, OpenAPI automatically generates a **JSON schema** (OAS3 file) that describes:

- Available endpoints
- Request/response formats
- Authentication requirements
- Data models

This JSON file is then used by client code generators (e.g. Swagger Codegen, OpenAPI Generator) to produce strongly typed client libraries.

Organizing APIs with `@Tag`

Each **endpoint class** in Elements should be annotated with a `@Tag`.
This tells the generator which **API file** the methods should be grouped into.

For example:

```
import jakarta.ws.rs.GET;
import jakarta.ws.rs.Path;
import io.swagger.v3.oas.annotations.tags.Tag;

@Path("/example")
@Tag(name = "Example")  // <-- Groups this endpoint into ExampleApi
public class ExampleEndpoint {

    @GET
    public String hello() {
        return "Hello from Example!";
    }
}
```

When client code is generated, this endpoint will appear inside an `ExampleApi` file in the generated client.

Warning

Not tagging an endpoint class will put the generated code into a DefaultApi class. If you have multiple DefaultApi classes in the client project, they could overwrite each other.

Registering OpenAPI in Your Application

For OpenAPI code generation to work, you must **register the Swagger OpenAPI JAX-RS resource** (`OpenApiResource`) in your Element's `Application` class.

Example:

```
// Swagger OpenAPI JAX-RS resource
import io.swagger.v3.jaxrs2.integration.resources.OpenApiResource;

import jakarta.ws.rs.core.Application;
import dev.getelements.elements.sdk.annotation.ElementServiceImplementation;
import dev.getelements.elements.sdk.annotation.ElementServiceExport;

@ElementServiceImplementation
@ElementServiceExport(Application.class)
```

```
public class HelloWorldApplication extends Application {

    /**
     * Here we register all the classes that we want
     * to be included in the Element.
     */
    @Override
    public Set<Class<?>> getClasses() {
        return Set.of(
            // Endpoints
            HelloWorld.class,
            HelloWithAuthentication.class,

            // Required for OpenAPI JSON generation
            OpenApiResource.class
        );
    }
}
```

Without `OpenApiResource`, your endpoints will run, but no OpenAPI spec will be generated — which means no client code generation.

Summary

- **OpenAPI** describes your API in a standardized JSON format.
- Use `@Tag(name = "...")` to group endpoints into client-side API files.
- Always register `OpenApiResource` in your `Application` class to enable codegen.

With these pieces in place, you can run OpenAPI tools to generate client libraries for your users, ensuring type-safe access to your API without manual boilerplate.