Learn Swagger and the Open API Specification

Open API Specification Continued

Security, errors, content types, and operation IDs

SDK BRIDGE

Introduction

- Covers
  - Security
  - Error Conditions
  - Content types (JSON, JPEG, etc.)
  - Operation IDs

Security

- Security means what kind of authentication or authorization is required
- Authentication: the user has correct username and password
- Authorization: the user has access to this API and data

Security types

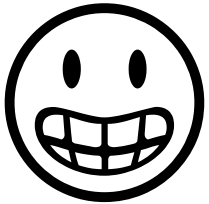
- None
  - Used for getting publically available information
- API key
  - Indicates that the app has permission to use the API
- Basic Authentication
  - Username and password is included in a header
- OAuth
  - Complex issuance of temporary token

How security is indicated

- Each operation has a security key
  - Contains an array of security definition objects
  - Often just one element in the array
- Security Definitions
  - The file contains a **securityDefinitions** key
  - Typically at the end
  - Contains security objects
- Security object
  - Contains information needed for that type of security

None

- When you do not have security…
- …you don’t need to do anything!



API key

- Add security key to each operation
  - Use dash to indicate an array
  - Create name for definition and use empty bracket, since no data is needed
- Add security definition
  - Add definition for that name in securityDefinition section
  - **type: apiKey**
  - **name:** name of the header or query parameter to be used
  - **in: query** or **header**

API key example

- Put a security key in the **get** section and **securityDefinitions** at the end of the file

```
    security:
      - api_key: [ ]
```

https://…?token=23a645ga2798

```
securityDefinitions:
  api_key:
    type: apiKey
    name: application-key
    in: header
```

```
securityDefinitions:
  api_key:
    type: apiKey
    name: token
    in: query
```


SDK BRIDGE

Basic authentication

- Add security key to an operation
  - Use dash to indicate an array
  - Create name for definition and use empty bracket, since no data is needed
- Add security definition
  - Add definition for that name in securityDefinition section
  - **type: basic**

Basic auth example

- Put a security key in the **get** section and **securityDefinitions** at the end of the file

```
  security:
    - basic_auth: [ ]
```

```
securityDefinitions:
  basic_auth:
    type: basic
```


SDK BRIDGE

OAuth

- OAuth is too complicated to explain here
- Add security key to request, like before
  - However, now you add scopes in the array
- Add security definition
  - Add definition for that name in securityDefinition section
  - **type: oauth2**
  - **authorizationUrl:** URL where credentials are entered
  - **tokenUrl:** URL for the token
  - **flow:** OAuth 2 flow (**implicit**, **password**, **application** or **accessCode**.)
  - **scopes:** list of scopes with descriptions

OAuth example

- Put a security key in the **get** section and **securityDefinitions** at the end of the file

```
  security:
    - oauth_example:
      - write:albums
```

```
securityDefinitions:
  oauth_example:
    type: oauth2
    authorizationUrl: http://example.com/authenticate
    flow: implicit
    scopes:
      write:albums: modify albums in your account
      read:albums: read your albums
```

Errors

- Errors are simply different response codes
- Often APIs return a special structure with errors
- Example 401 (Unauthorized) code returned

```
{"errorCode": 13,  "message": "Username not found"}
```

- Should include schema for every potential status code
- Refer to this in response

Error example

```yaml
responses:
  # Response code
  200:
    description: Successful response

  401:
    description: Unauthorized
    schema:
      $ref: '#/definitions/error'
```

```yaml
# Error info
error:
  properties:
    code:
      type: integer
    message:
      type: string
```

Content Types

- Content types indicate the format of the data in the request and response bodies
- This is done through the **Content-Type** header
- You can specify this for:
  - The whole API
  - Individual operations
- Use the **consumes** and **produces** keys
  - **consumes** for requests, **produces** for responses
  - Use the **Content-Type** value (for example, **application/json**)

Example Content-Type

```yaml
# URL data
host: api.example.com
basePath: /photo
schemes:
  - https

consumes:
  - application/json
produces:
  - application/json
```

```yaml
consumes:
  - application/json
produces:
  - application/json

# Incomplete response (to finish later)
responses:
  # Response code
  200:
    description: Successful response
```

Operation ID

- Although it doesn't show up in the documentation, you can optionally add an operation ID to each request
- Some tools will use this

```
paths:
  # Photo albums
  /album:
    # Get one or more albums
    get:
      operationId: getAlbums

      # Query parameters
      parameters:
        # Starting date
        - name: start
```