Note: the slides (including speaker notes) are available on GitHub:

github.com/sjaensch/swagger_talk
BUILDING SERVICE INTERFACES WITH OPENAPI / SWAGGER

Stephan Jaensch
sjaensch@yelp.com / @s_jaensch
Yelp Stats (Q1 2016)

- 90M
- 102M
- 70%
- 32
WHAT THIS TALK IS ABOUT

- What is OpenAPI / Swagger
- Short introduction of some of the available libraries
- The things the tutorials typically don’t talk about
WHY OPENAPI / SWAGGER

- RESTful API specification and tooling
- Solves several problems when faced with building services
PYTHON SUPPORT

- bravado
- swaggerpy
- pyramid_swagger
- connexion
- django-rest-swagger
swagger: "2.0"
info:
  version: "1.0.0"
  title: "User service"
host: "user-service.com"
basePath: "/api"
schemes:
  - "http"
consumes:
  - "application/json"
produces:
  - "application/json"
paths:
  /users:
    get:
      summary: List users by IDs
      operationId: list_users
      tags:
        - user
      parameters:
        - name: "user_ids"
          in: "query"
          description: "IDs for which to return user objects"
          required: true
          type: "array"
          items:
            type: "integer"
responses:
  "200":
    description: "A list of users"
    schema:
      type: "array"
      items:
        $ref: "#/definitions/User"
    default:
      description: unexpected error
      schema:
        $ref: "#/definitions/Error"
definitions:
  User:
    type: "object"
    required:
      - "id"
      - "username"
    properties:
      id:
        type: "integer"
      username:
        type: "string"
      business_id:
        type: "integer"
# this is an example of the Uber API

```yaml
- # as a demonstration of an API spec in YAML
- swagger: '2.0'
  info:
    title: Uber API
    description: Move your app forward with the Uber API
    version: "1.0.0"
    host: api.uber.com
  schemes:
    - https
  basePath: /v1
  produces:
    - application/json
  paths:
    /products:
      get:
        summary: Product Types
        description: The Products endpoint returns information about the *Uber* products offered at a given location. The response includes the display name and other details about each product, and lists the products in the proper display order.
        parameters:
          - name: latitude
            in: query
            description: Latitude component of location.
            required: true
            type: number
            format: double
          - name: longitude
            in: query
            description: Longitude component of location.
            required: true
            type: number
            format: double
```
## Business Owner App backend API

### account: Management of the biz user's account

**GET**  `/business/{business_id}/ad_stats/delivery/v1`  
( getAdStatsDelivery ) Returns delivery information for the business' ad campaigns

**GET**  `/business/{business_id}/ad_stats/historical_metrics/v1`  
( getAdStatsHistoricalMetrics ) Returns historical metrics for the business' ad campaigns

**GET**  `/business/{business_id}/ad_stats/historical_metrics/v2`  
( getAdStatsHistoricalMetricsV2 ) Returns historical metrics for the business' ad campaigns

**Response Class (Status 200)**
Returns historical metrics for the business' ad campaigns

### Model Example Value

<table>
<thead>
<tr>
<th>AdStatsHistoricalMetricsV2Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>metrics (Array[HistoricalMetric]), available_months (Array[integer]): The month timestamps for which metrics are available, time_period (string): Term that defines for which period data is returned = ['30_DAYS', '12_MONTHS', 'MONTH'], month_timestamp (object, optional): Denotes the month and year for which the metrics are returned. Only returned if time_period is MONTH, currency_code (string): ISO currency code for amount based metrics</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HistoricalMetric</th>
</tr>
</thead>
<tbody>
<tr>
<td>data (Array[HistoricalMetricData]): List of metric data points (timestamp - value pairs), display_name (string): Localized name of the metric</td>
</tr>
</tbody>
</table>

[swagger.io/swagger-ui/](swagger.io/swagger-ui/)
from bravado.client import SwaggerClient
from bravado.fido_client import FidoClient

user_client = SwaggerClient.from_url(
    'http://service_host:port/swagger.yaml',
    http_client=FidoClient(),
)

user_future = client.user.list_users(user_ids=[1])
business_future = client.business.list_bizs(business_ids=[1])

user = user_future.result(timeout=DEFAULT_TIMEOUT)
business = business_future.result(timeout=DEFAULT_TIMEOUT)
<table>
<thead>
<tr>
<th>URL</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>/business/72bgxw/cxsvjx71bmsrqr/mtb/e6rt5hged7dlwya-ypnphgv1</td>
<td>446.713ms</td>
</tr>
<tr>
<td>/token/normal/obiozqthfjkjibsvobu650yzvw</td>
<td>8.000ms</td>
</tr>
<tr>
<td>/biz_user/565654/business_ids</td>
<td>63.000ms</td>
</tr>
<tr>
<td>/message_to_business/v1/4271192</td>
<td>37.791ms</td>
</tr>
<tr>
<td>/message_to_business/v1/condensed</td>
<td>24.000ms</td>
</tr>
<tr>
<td>/user/v1/user_stats_by_ids</td>
<td>63.000ms</td>
</tr>
<tr>
<td>/user/v1/permissions</td>
<td>30.718ms</td>
</tr>
<tr>
<td>/message_to_business/v1/4271192/permissions</td>
<td>86.183ms</td>
</tr>
<tr>
<td>/private_message/v1/contents</td>
<td>29.000ms</td>
</tr>
<tr>
<td>/flag/is_</td>
<td>47.366ms</td>
</tr>
</tbody>
</table>
"WAR STORIES"
1. DEALING WITH NETWORK ISSUES

```python
@retry(tries=3, exceptions=[fido.exceptions.HTTPTimeoutError])
def retry_result(future):
    return future.result(timeout=2)

future = client.user.list_users(user_ids=[1])
# redo the request in case of network failure... right?
result = retry_result(future)
```
2. **NULL VALUES FOR OPTIONAL FIELDS**

```json
{
    "id": 1,
    "username": "john",
    "business_id": null
}
```

```python
client = SwaggerClient.from_url('http://service_host:port/swagger.yaml',
    config={'validate Responses': False},
)
```
3. CREATING THE CLIENT MAY KILL PERFORMANCE

```python
import time
from bravado.client import SwaggerClient

time_start = time.time()
print(round(time.time() - time_start, 2))

>>> 1.60
```
get_yelp_sorted 50/95/99
4. ISSUES WITH DEPLOYMENT AT SCALE
4.1 Adding a Non-Optional Field to the Response

1. Add it as optional to the spec, ship implementation.
2. Change the spec to mark it as required.
4.2 REMOVING A REQUIRED FIELD FROM THE RESPONSE

1. Remove it from the spec
2. Ship implementation
   (Don't do that)
4.2 REMOVING A REQUIRED FIELD FROM THE RESPONSE

Summary
A change was made to an internalapi endpoint which caused many bizapp errors until the bizapp service was restarted. Even though attempted to push it in a forwards-compatible way, he failed to do so because bizapp didn’t restart in between the 2 pushes.
4.3 ADDING A REFERENCE TO A NEW SPEC FILE

1. Add the file
2. Add the reference to it

Or let `pyramid_swagger` combine the spec for you
5. CHANGING THE TAG OF AN ENDPOINT

```
future = client.user.list_users(user_ids=[1])
tag --^       ^-- operationId
```

Summary

The tag parameter in some of yelp-main's internal API swagger files were changed, causing some services using those swagger files to throw errors, as changing the tags also changes the method name to call the internal API endpoint.
CONCLUSION

1. When in doubt: version it
2. Deal with the network
   • "Graceful Degradation when Services Fail" by Daniel Riti
     @PyCon 2016
3. Rolling forward and backward is not instantaneous
4. Be mindful of the differences between swaggerpy and bravado
5. Don't do services if you don't have to
OTHER TALKS BY YELPERS

"Asynchronous network requests in a web application"
by Lauris Jullien; Thursday, 10:30, A1

Watch the video for "Protect your users with circuit
breakers"
by Scott Triglia; Tuesday, 14:00, A2
QUESTIONS?

sjaensch@yelp.com / @sJaensch