# Corso di Progettazione di Applicazioni Web e Mobile

Mirko Calvaresi

# INTEROPERABILITY & DEPLOYMENT

# Let’s start with some definition

« W3C definition of a Web Service

*Software system designed to support interoperable machine-to-machine interaction over a network.*

»

# WEB SERVICE

- Web services are designed for machine-to-machine (or application-to-application) interaction
- Web services should Not platform dependent and technology dependent
- Web services should allow communication over a network using preferrably HTTP

http://www.springboottutorial.com/introduction-to-web-services-with-soap-and-rest

# WEB SERVICE AND INTEROPERABILITY

## SOAP

Simple Object Access Protocol (SOAP) is the basis on which web services are built. It is an XML-based protocol used to communicate and interoperate with web services

https://www.w3.org/TR/soap/

## REST

REST stands for REpresentational State Transfer.

Stateless client-server architecture in which the web services are viewed as resources and can be identified by their URLs

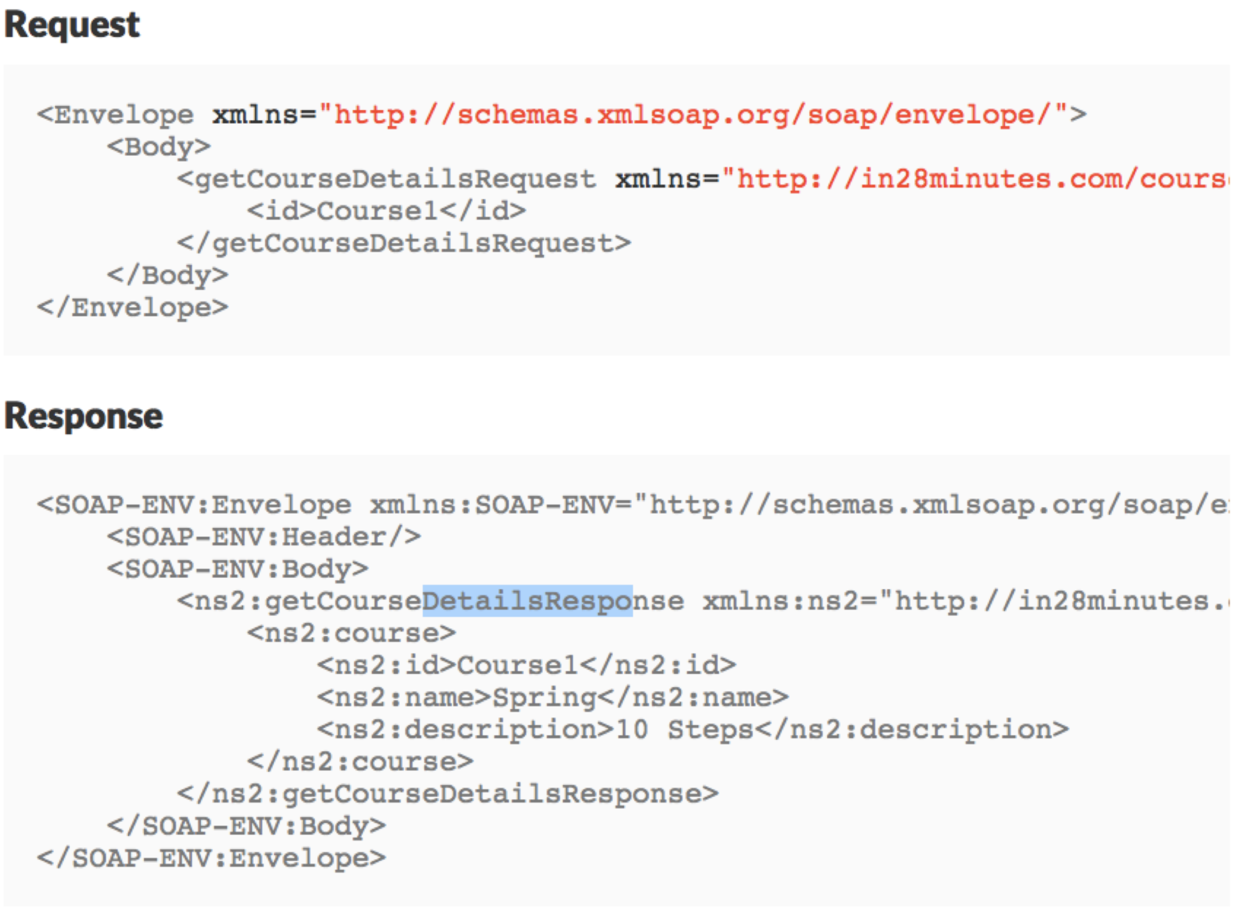
# WEB SERVICE AND INTEROPERABILITY

## SOAP

SOAP was earlier an abbreviation for Simple Object Access Protocol.

In SOAP, the request and response are in XML format. However, not all types of XML are valid SOAP Requests.
SOAP defines a standard XML format. We will use WSDL (Web Service Definition Language) to define the format of request xml and the response xml.

# SOAP ANATOMY



## Request

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <getCourseDetailsRequest xmlns="http://in28minutes.com/cours
            <id>Course1</id>
        </getCourseDetailsRequest>
    </Body>
</Envelope>
```

## Response

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/e
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns2:getCourseDetailsResponse xmlns:ns2="http://in28minutes.
            <ns2:course>
                <ns2:id>Course1</ns2:id>
                <ns2:name>Spring</ns2:name>
                <ns2:description>10 Steps</ns2:description>
            </ns2:course>
        </ns2:getCourseDetailsResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

# WEB SERVICE AND INTEROPERABILITY

REST builds on top of HTTP (Hypertext Transfer Protocol
Rests uses the HTTP methods

POST - Create a new resource
GET - Read a resource
PUT - Update an existing resource
DELETE - Delete a resource

And standard HTTP response codes:
200 - SUCESS
404 - RESOURCE NOT FOUND
400 - BAD REQUEST
201 - CREATED
401 - UNAUTHORIZED
415 - UNSUPPORTED TYPE - Representation not supported for the resource
500 - SERVER ERROR

# WEB SERVICE AND INTEROPERABILITY

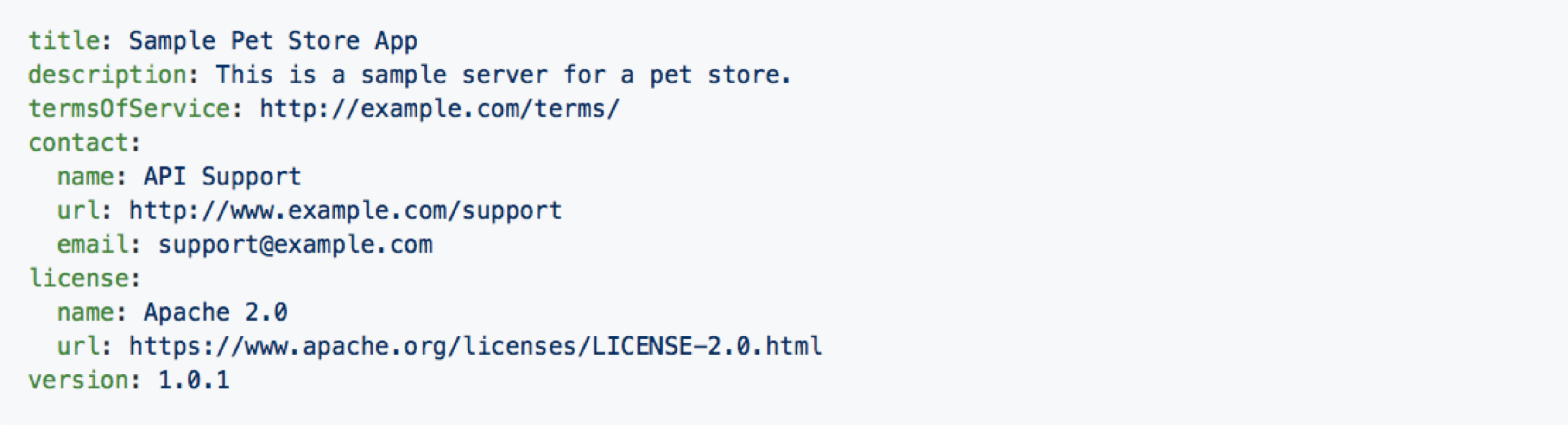
**Rest**
Currently the world is going REST, because it is much simpler to build and more effective in the result

To try a Rest Api and design it it can be used the following method.

https://editor.swagger.io

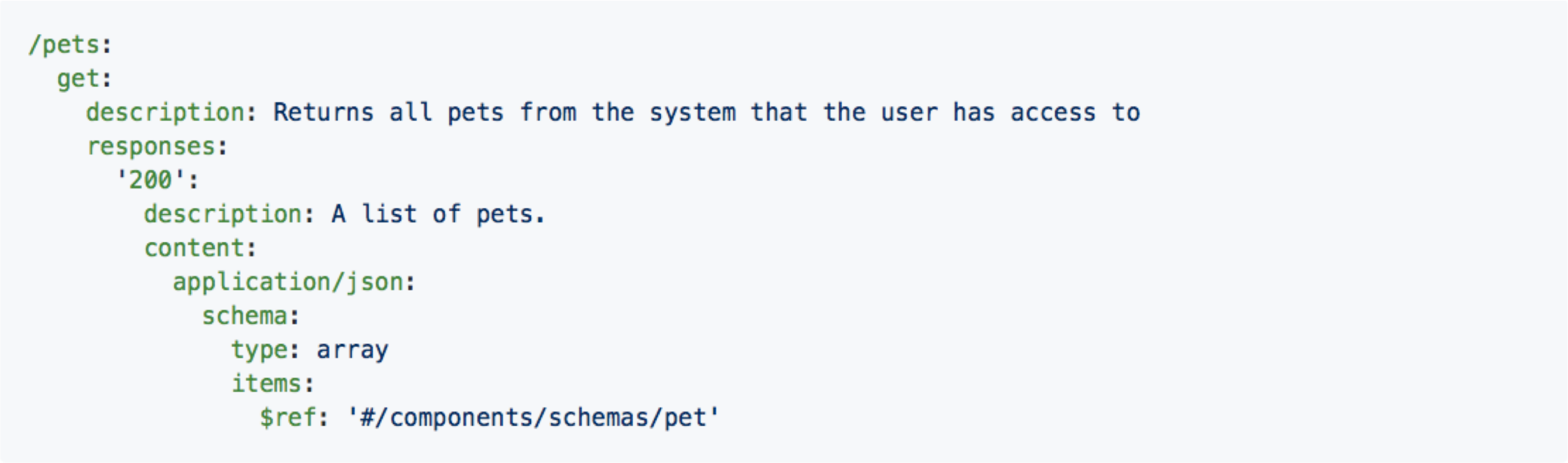
# OPEN API Specification

https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.1.md



```
title: Sample Pet Store App
description: This is a sample server for a pet store.
termsOfService: http://example.com/terms/
contact:
  name: API Support
  url: http://www.example.com/support
  email: support@example.com
license:
  name: Apache 2.0
  url: https://www.apache.org/licenses/LICENSE-2.0.html
version: 1.0.1
```

# OPEN API Specification: an example from the “Pet Store”



```yaml
/pets:
  get:
    description: Returns all pets from the system that the user has access to
    responses:
      '200':
        description: A list of pets.
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/pet'
```