PHINMA EDUCATION
MAKING LIVES BETTER THROUGH EDUCATION

ITE 306: Integrative Programming and Technologies
Module #18

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

Lesson title: Web service Architecture

Lesson Objectives:

At the end of this module, I should be able to:

1.) Know how Web Services Work
2.) Differentiate the type of web services
3.) Create a simple RESTful Web Services With Python Flask

Materials:

Activity Sheets, computer, internet connection

References:

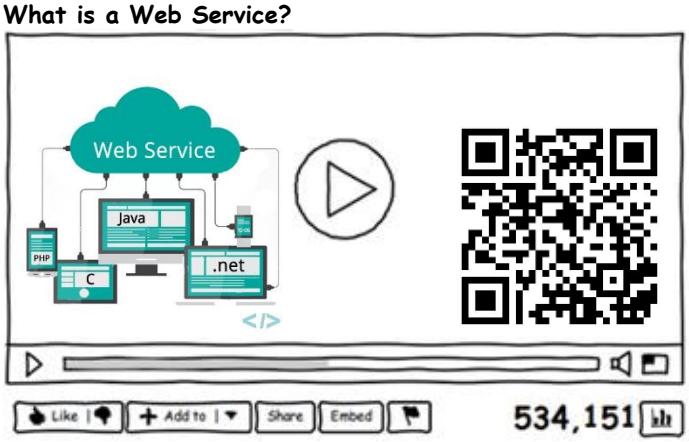
- guru99.com
- https://dzone.com/articles/restful-web-services-with-python-flask
- https://form.jotform.me/90421326958460

Productivity Tip:

"Question, rather than affirm, your goals. Research shows that if you ask yourself whether you will accomplish your tasks or not, rather than simply saying, "I will accomplish this task," leads to greater productivity."

A. LESSON PREVIEW/REVIEW

1) Introduction



In this video we will see:

1. What is a web service
2. Basic concept behind web services
3. Why we use it

What is a Web Service?

Web Service
- service available over the web
- enables communication between applications over the web
- provides a standard protocol/format for communication

Why we use it?

- platform independent communication
- using web services two different applications can talk to each other and exchange data or information.

Scan the QR code or Open this link:
https://www.youtube.com/watch?v=oTzNRv6X51o

2) Activity 1: What I Know Chart, part 1

Try answering the questions below by writing your ideas under the first column *What I Know*. It's okay if you write key words or phrases that you think are related to the questions. Leave the third column blank for Activity 4.

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

What I Know	Questions:	What I Learned (Activity 4)
	1. What is your idea about web services?	
	2.What are the types of web services that you know?	

B.MAIN LESSON

1) Activity 2: Content Notes

What is Web Service?

Web service is a standardized medium to propagate communication between the client and server applications on the World Wide Web. A web service is a software module that is designed to perform a certain set of tasks.

- The web services can be searched for over the network and can also be invoked accordingly.
- When invoked, the web service would be able to provide the functionality to the client, which invokes that web service.

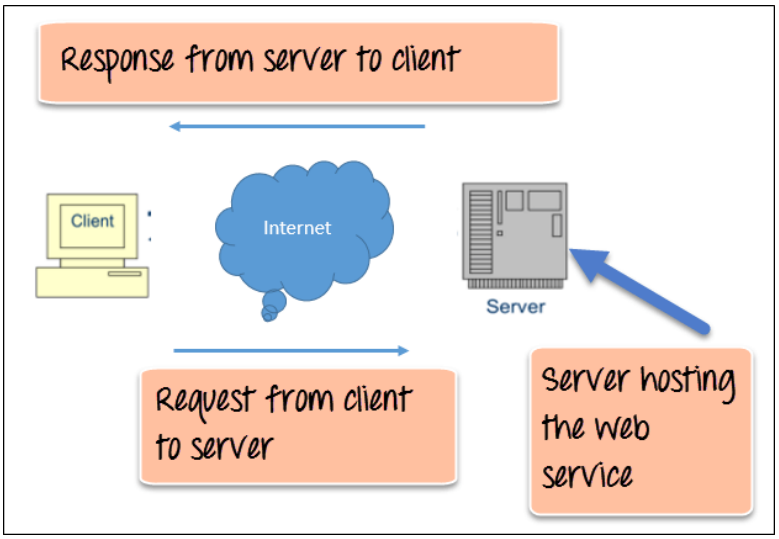
How Web Services Work?

The diagram shows a very simplistic view of how a web service would actually work. The client would invoke a series of web service calls via requests to a server which would host the actual web service.

These requests are made through what is known as remote procedure calls. **Remote Procedure Calls (RPC)** are calls made to methods which are hosted by the relevant web service.

As an example, Amazon provides a web service that provides prices for products sold online via amazon.com. The

Web Service Architecture Diagram



Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

front end or presentation layer can be in .Net or Java but either programming language would have the ability to communicate with the web service.

The main component of a web service is the data which is transferred between the client and the server, and that is XML. XML (Extensible markup language) is a counterpart to HTML and easy to understand the intermediate language that is understood by many programming languages.

So when applications talk to each other, they actually talk in XML. This provides a common platform for application developed in various programming languages to talk to each other.

Web services use something known as **SOAP (Simple Object Access Protocol)** for sending the XML data between applications. The data is sent over normal HTTP. The data which is sent from the web service to the application is called a SOAP message. The SOAP message is nothing but an XML document. Since the document is written in XML, the client application calling the web service can be written in any programming language.

Why do you need a Web Service?

Modern day business applications use variety of programming platforms to develop web-based applications. Some applications may be developed in Java, others in .Net, while some other in Angular JS, Node.js, etc.

Most often than not, these heterogeneous applications need some sort of communication to happen between them. Since they are built using different development languages, it becomes really difficult to ensure accurate communication between applications.

Here is where web services come in. Web services provide a common platform that allows multiple applications built on various programming languages to have the ability to communicate with each other.

Type of Web Service

There are mainly two types of web services.

1. SOAP web services.
2. RESTful web services.

In order for a web service to be fully functional, there are certain components that need to be in place. These components need to be present irrespective of whatever development language is used for programming the web service.

SOAP (Simple Object Access Protocol)

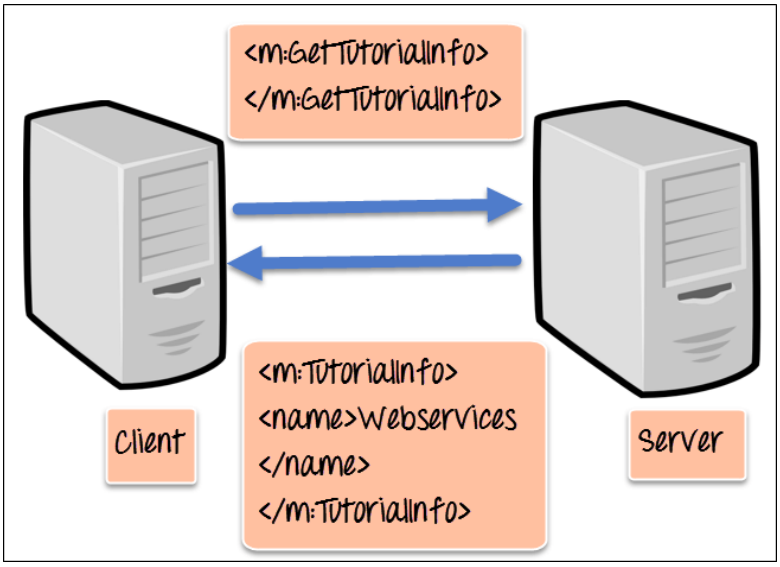
SOAP is known as a transport-independent messaging protocol. SOAP is based on transferring XML data as SOAP Messages. Each message has something which is known as an XML document. Only the structure of the XML document follows a specific pattern, but not the content. The best part of Web services and SOAP is that its all sent via HTTP, which is the standard web protocol.

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

Here is what a SOAP message consists of:

- Each SOAP document needs to have a root element known as the <Envelope> element. The root element is the first element in an XML document.
- The "envelope" is in turn divided into 2 parts. The first is the header, and the next is the body.
- The header contains the routing data which is basically the information which tells the XML document to which client it needs to be sent to.
- The body will contain the actual message.



The diagram shows a simple example of the communication via SOAP.

SOAP – SOAP is a protocol which was designed before REST came into the picture. The main idea behind creating SOAP was to ensure that programs built on different platforms and programming languages could securely exchange data.

REST – This was designed specifically for working with components such as media components, files, or even objects on a particular hardware device. Any web service which is defined on the principles of REST can be called a RESTful web service. REST uses the normal HTTP verbs of GET, POST, PUT and DELETE for working with the required components.

WSDL (Web Services Description Language)

A web service cannot be used if it cannot be found. The client invoking the web service should know where the web service actually resides.

Secondly, the client application needs to know what the web service actually does, so that it can invoke the right web service. This is done with the help of the **WSDL**, known as the Web services description language. The WSDL file is again an XML-based file which basically tells the client application what the web service does. By using the WSDL document, the client application would be able to understand where the web service is located and how it can be utilized.

Universal Description, Discovery, and Integration (UDDI)

UDDI is a standard for describing, publishing, and discovering the web services that are provided by a particular service provider. It provides a specification which helps in hosting the information on web services.

How can a client application locate a WSDL file to understand the various operations offered by a web service? So UDDI is the answer to this and provides a repository on which WSDL files can be

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

hosted. The client application will have complete access to the UDDI, which acts as a database containing all the WSDL files.

Just as a telephone directory has the name, address and telephone number of a particular person, the same way the UDDI registry will have the relevant information for the web service. So that a client application knows, where it can be found.

Web Services Advantages

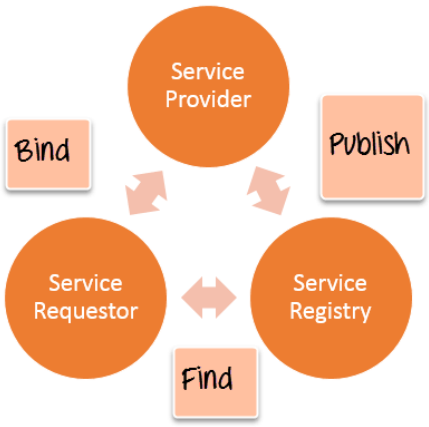
We already understand why web services came about in the first place, which was to provide a platform which could allow different applications to talk to each other. Let's look at some other advantages of why it is important to use web services.

1. **Exposing Business Functionality on the network** - A web service is a unit of managed code that provides some sort of functionality to client applications or end users. This functionality can be invoked over the HTTP protocol which means that it can also be invoked over the internet. Nowadays all applications are on the internet which makes the purpose of Web services more useful. That means the web service can be anywhere on the internet and provide the necessary functionality as required.
2. **Interoperability amongst applications** - Web services allow various applications to talk to each other and share data and services among themselves. All types of applications can talk to each other. So instead of writing specific code which can only be understood by specific applications, you can now write generic code that can be understood by all applications
3. **A Standardized Protocol which everybody understands** - Web services use standardized industry protocol for the communication. All the four layers (Service Transport, XML Messaging, Service Description, and Service Discovery layers) uses well-defined protocols in the web services protocol stack.
4. **Reduction in cost of communication** - Web services use SOAP over HTTP protocol, so you can use your existing low-cost internet for implementing web services.

Web service Architecture

Every framework needs some sort of architecture to make sure the entire framework works as desired. Similarly, in web services, there is an architecture which consists of three distinct roles as given below

1. **Provider** - The provider creates the web service and makes it available to client application who want to use it.
2. **Requestor** - A requestor is nothing but the client application that needs to contact a web service. The client application can be a .Net, Java, or any other language based application which looks for some sort of functionality via a web service.
3. **Broker** - The broker is nothing but the application which provides access to the UDDI. The UDDI which enables the client application to locate the web service.



Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

The diagram showcases how the Service provider, the Service requestor and Service registry interact with each other.

- **Publish** - A provider informs the broker (service registry) about the existence of the web service by using the broker's publish interface to make the service accessible to clients
- **Find** - The requestor consults the broker to locate a published web service
- **Bind** - With the information it gained from the broker (service registry) about the web service, the requestor is able to bind, or invoke, the web service.

2) Activity 3: Skill-building Activities (with answer key)

A. Warm-up Exercise. Encircle the letter of the correct answer.

1. Which of the following is true about Web service?
   a. It is available over the Internet or private (intranet) networks.
   b. It uses a standardized XML messaging system.
   c. It is not tied to any one operating system or programming language.
   d. All of the above.

2. What UDDI stands for?
   a. Uniform Description, Discovery, and Integration
   b. Universal Description, Discovery, and Integration
   c. Uniform Discovery, Description, and Integration
   d. Uniform Discovery, Delivery, and Integration

3. A web services takes the help of XML to tag the data, format the data.
   a. True
   b. False
   c. Maybe
   d. Not sure

4. The service provider implements the service and makes it available on the Internet.
   a. True
   b. False
   c. Maybe
   d. Not sure

5. Which of the following is not the type of web services?
   a. SOAP
   b. REST
   c. WSDL
   d. None of the above

B. Coding Time. Let's create a simple RESTful Web Services with Python Flask.

RESTful Web Service is an architectural style, where the data or the structural components of a system is described in the form of **URI (Uniform Resource Identifier)** and the behaviors are described in terms of methods. The resources can be manipulated using **CRUD (Create, Read,**

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

Update and Delete) operations. The communication protocol for REST is HTTP since it suits the architecture requirement of being a stateless communication across the Client and Server.
There are many frameworks available in Python for web development, but Django (pronounced as yango) and Flask stands out of the crowd being a full stack framework.

STEP 1: Installation of Flask

To install flask framework. If you have pip installed in your Python environment, please follow this step.

```
pip install Flask
```

If you don't have pip, please download the flask from: http://pypi.python.org/packages/source/F/Flask/Flask-0.10.1.tar.gz and execute the setup.py

Step 2: Create a Hello World - Web Server

First, we create a web server, create a dictionary to hold a JSON objects for a couple of employee records and then we add RESTful APIs for each supported operation. Please look at the below program, which creates a web server. Save the below program into **hello.py** and execute it.

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def hello():
    return "Hello World!"
if __name__ == "__main__":
    app.run()
```

The below line of the code creates an app object from Flask.

```
app = Flask(__name__)
```

app.run() starts the web server and ready to handle the request. But at this moment, it can handle only one request. It is defined in the below line of code.

```
@app.route("/")
def hello():
    return "Hello World !"
```

Execute the above program and you will see that you web server is ready to service you.

```
* Running on http://localhost:5000/
```

Now you can open your web browser and check your web server. The server is available in the URL **http://localhost:5000/.**

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

STEP 3: Develop the RESTful Services

To develop the restful services for the planned objective, let's create an in-memory database in python using the dictionary data type. Please find the code snippet below: We can continue to use the ***hello.py*** and type the below code, just after the Flask app creation statement ***app = Flask(__name__)***.

```
empDB=[
 {
 'id':'101',
 'name':'Dorry Britz',
 'title':'Technical Leader'
 },
 {
 'id':'102',
 'name':'Barbie Gurl',
 'title':'Software Engineer'
 }
 ]
```

STEP 3.1: GET

In the previous section, we have created two employees in the dictionary. Now let's write a code to retrieve them using web services. As per our plan, we need two implementations one is to retrieve all the employees and another one to retrieve the specific employee with the given id.

STEP 3.1.1: GET All

```
@app.route('/empdb/employee/',methods=['GET'])
def getAllEmp():
    return jsonify({'emp':empDB})
```

In the above code, we have created a URI named **'/empdb/employee'** and also we defined the method as **"GET"**. To service the GET call for the URI, Flask will call the function **getAllEmp()**. It will in turn simply call the **"jsonify"** method with ***employeeDB*** as the argument. The **"jsonify"** is a flask method, will set the data with the given JSON object which is passed as a Python dictionary and set the headers appropriately, in this case **"Content-type: application/json"**.

STEP 3.1.2: Get Specific

Now we develop the rest service to get an employee with a given id.

```
@app.route('/empdb/employee/<empId>',methods=['GET'])
def getEmp(empId):
    usr = [ emp for emp in empDB if (emp['id'] == empId) ]
    return jsonify({'emp':usr})
```

The above code will find the employee object with the given id and send the JSON object in the data.

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

STEP 4: Double check the complete code, RUN **hello.py** and open your web browser on ***http:///localhost:5000/*** the basic web service that you have developed.

Hello.py

```
from flask import Flask
from flask import jsonify
from flask import request
app = Flask(__name__)
empDB=[
 {
 'id':'101',
 'name':'Dorry Britz',
 'title':'Technical Leader'
 },
 {
 'id':'102',
 'name':'Barbie Gurl',
 'title':'Software Engineer'
 }
 ]

@app.route("/")
def hello():
    return "Hello World!"

@app.route('/empdb/employee/',methods=['GET'])
def getAllEmp():
    return jsonify({'emp':empDB})

@app.route('/empdb/employee/<empId>',methods=['GET'])
def getEmp(empId):
    usr = [ emp for emp in empDB if (emp['id'] == empId) ]
    return jsonify({'emp':usr})

if __name__ == '__main__':
       app.run()
```

ANSWER THE FOLLOWING QUESTIONS:

Q1: What is the output if you open this URL while hello.py is running: ***http://127.0.0.1:5000/*** or ***http:///localhost:5000/***

Answer:

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

Q2: What is the output if you open this URL while hello.py is running:
http://localhost:5000/empdb/employee/

Answer:

Q3: What is the output if you open this URL while hello.py is running:
http://localhost:5000/empdb/employee/101

Answer:

Q4: What is the output if you open this URL while hello.py is running:
http://localhost:5000/empdb/employee/103

Answer:

Q5: On your command prompt press ***"ctrl +c"*** to end the process of hello.py. What is the output if you open this URL if hello.py is not running: ***http://localhost:5000/empdb/employee/***

Answer:

3) Activity 4: What I Know Chart, part 2

It's time to answer the questions in the *What I Know* chart in Activity 1. Log in your answers in the third column of the table in Activity 1.

4) Activity 5: Check for Understanding

Give the meaning of the following acronyms.

1.) What is RPC?
Answer: _____

2.) What is WSDL?
Answer: _____

3.) What UDDI means?
Answer: _____

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

4.) What REST means?
Answer: _____

5.) What is the meaning of SOAP?
Answer: _____

C. LESSON WRAP-UP

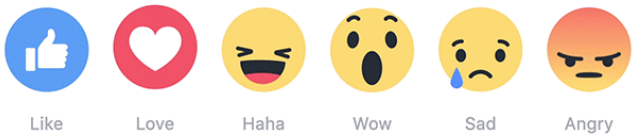
Activity 6: Thinking about Learning

Congratulations for finishing this module! **Shade** the number of the module that you finished to track how much work you have accomplished and how much work there is left to do.

You are done with the session! Let's track your progress.

Period 1											Period 2											Period 3								
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Rate the session for today by encircling the emoji that best captures your experience and write your reason for choosing that emoji.



Reason/s: _____

FAQs

1. What is Web Service?

Web service is a standardized medium to propagate communication between the client and server applications on the World Wide Web. Web services provide a common platform that allows multiple applications built on various programming languages to have the ability to communicate with each other.

2. What are the popular web services protocols?

- ***SOAP- Simple Object Access Protocol***
  SOAP was developed as an intermediate language so that applications built on various programming languages could talk quickly to each other and avoid the extreme development effort.
- ***WSDL - Web Services Description Language***
  WSDL is an XML-based file which tells the client application what the web service does and gives all the information required to connect to the web service.
- ***REST - REpresentational State Transfer***
  REST is used to build Web services that are lightweight, maintainable, and scalable.

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

KEY TO CORRECTIONS

Answers to Skill-Building Exercises:

A. Warm-up Exercise. Encircle the letter of the correct answer.

1. Which of the following is true about Web service?
   a. It is available over the Internet or private (intranet) networks.
   b. It uses a standardized XML messaging system.
   c. It is not tied to any one operating system or programming language.
   d. All of the above.

2. What UDDI stands for?
   a. Uniform Description, Discovery, and Integration
   b. Universal Description, Discovery, and Integration
   c. Uniform Discovery, Description, and Integration
   d. Uniform Discovery, Delivery, and Integration

3. A web services takes the help of XML to tag the data, format the data.
   a. True
   b. False
   c. Maybe
   d. Not sure

4. The service provider implements the service and makes it available on the Internet.
   a. True
   b. False
   c. Maybe
   d. Not sure

5. Which of the following is not the type of web services?
   a. SOAP
   b. REST
   c. WSDL
   d. None of the above

B. Coding Time

Q1: What is the output if you open this URL while hello.py is running: ***http://127.0.0.1:5000/*** or ***http://localhost:5000/***

Answer: `Hello World!`

Q2: What is the output if you open this URL while hello.py is running: ***http://localhost:5000/empdb/employee/***

Name: _____ Class number: _____

Section: _____ Schedule: _____ Date: _____

Answer:
{"emp":[{"id":"101","name":"Dorry Britz","title":"Technical Leader"},{"id":"102","name":"Barbie Gurl","title":"Software Engineer"}]}

Q3: What is the output if you open this URL while hello.py is running:
http:///localhost:5000/empdb/employee/101

Answer:
{"emp":[{"id":"101","name":"Dorry Britz","title":"Technical Leader"}]}

Q4: What is the output if you open this URL while hello.py is running:
http:///localhost:5000/empdb/employee/103

Answer:
{"emp":[]}

Q5: On your command prompt press ***"ctrl +c"*** to end the process of hello.py. What is the output if you open this URL if hello.py is not running: ***http:///localhost:5000/empdb/employee/***

Answer: This site can't be reached / ERR_CONNECTION_REFUSED

Answers to Checking for Understanding:

1.) What is RPC?
Answer: Remote Procedure Calls
2.) What is WSDL?
Answer: Web Services Description Language
3.) What UDDI means?
Answer: Universal Description Discovery and Integration
4.) What REST means?
Answer: Representational State Transfer
5.) What is the meaning of SOAP?
Answer: Simple Object Access Protocol