Introduction to Web Services

Software Testing

Agenda

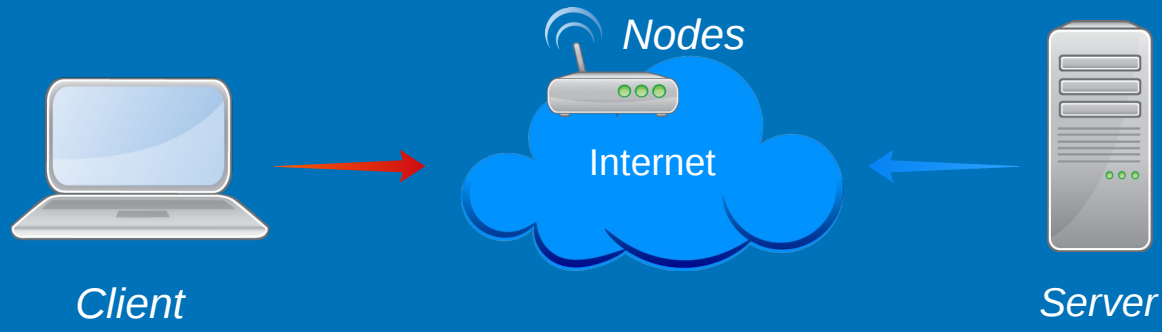
- Brief introduction to computer networks
- How does the Internet work?
- Understanding HTTP/HTTPS protocol
- Transferring data using JSON & XML
- Web services and APIs
- Types of Web Services
- Testing RESTful web services with Postman

Brief introduction to computer networks

- A computer network is a group of computers that „talk“ to each other via communication protocols, for the purpose of sharing resources/information.
- Depending on their sizes and purpose, there are different types of networks (e.g. LAN – Local Area Network, WAN – Wide Area Network, VPN – Virtual Private Network, etc.)
- **The Internet** is the global system of interconnected computer - a **network of networks** that consists of private, public, academic, business, and government networks, linked by a broad array of electronic, wireless, and optical networking technologies.
- The **World Wide Web** (WWW), is an information system where documents (web pages) and other web resources are identified by **URLs – Unform resource locators**, or Internet addresses (*e.g. www.example.com*), which are accessible over the Internet.

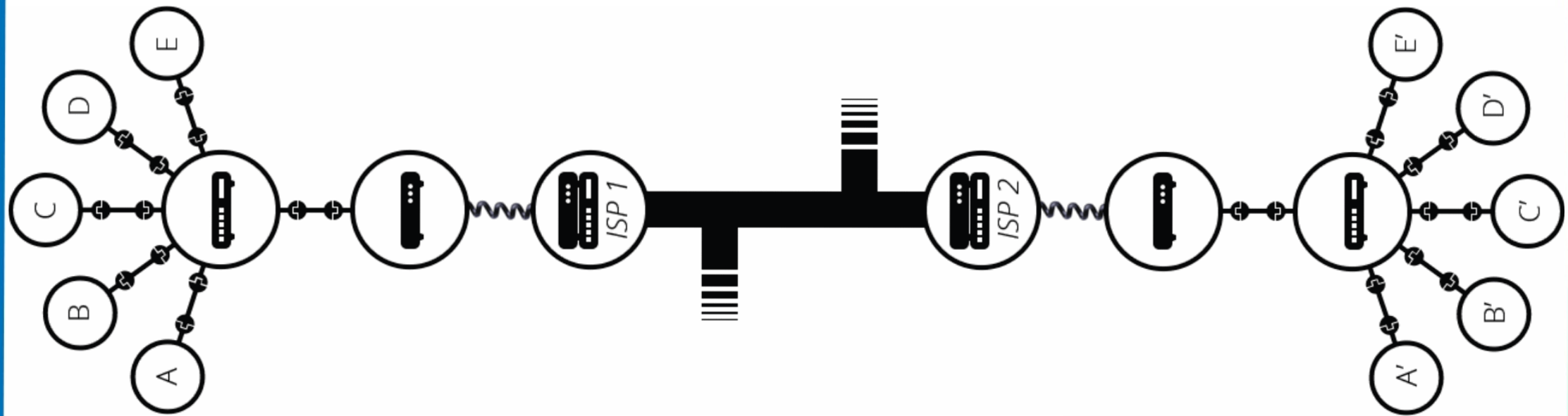
How does the Internet work?

- To understand the Internet, we may look at it as a system of two main components - **hardware** (servers, routers, cables, satellites, etc.) and **protocols** (a set of software rules that machines follow to complete tasks)
- From **hardware perspective**, devices connected on the Internet can be classified as: **clients** (consuming information); **servers** (storing & providing information); **nodes** (connection points to allow the traffic between clients and servers)



How does the Internet work?

- How are all these devices connected in such a complex worldwide network?
- In smaller networks (e.g. home network), each computer is connected to a special tiny computer called a **router**. To send a message to computer B, computer A must send the message to the router, which in turn forwards the message to computer B and makes sure the message is not delivered to computer C.
- In larger networks (e.g. LAN), computers are connected to routers, which in turn are connected to other routers, making it possible to scale connections.
- When we want connect to the Internet, we also need an **Internet Service Provider (ISP)**. An ISP is a company that manages special routers that are all linked together and can also access other ISPs' routers. So the message from our network is carried through the network of ISP networks to the destination network. The Internet consists of this whole infrastructure of networks.


E
D
C
B
A
ISP 1
ISP 2
E'
D'
C'
B'
A'

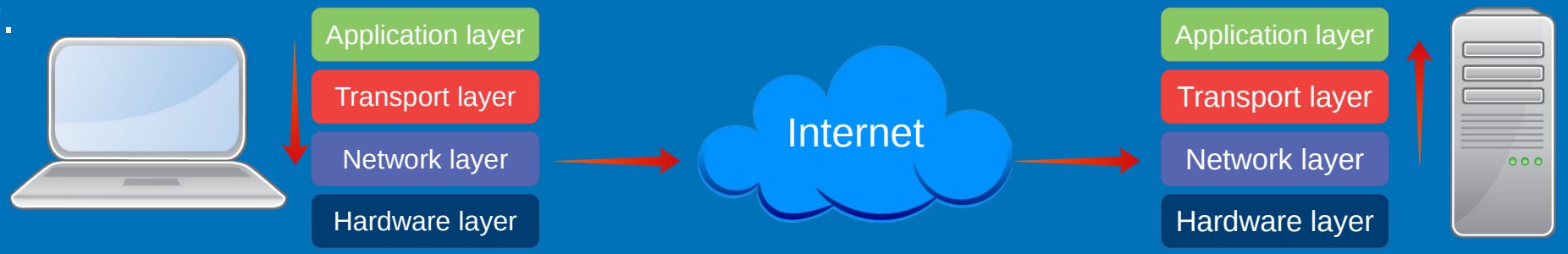
How does the Internet work?

- All of these hardware connections wouldn't create a network without the second component of the Internet: the protocols.
- **Protocols** are sets of rules that machines follow to complete tasks. They provide both the method and a common language for machines to use to transmit data.
- Communication on the Internet is achieved via the **Internet protocol suite**, also known as **TCP/IP** protocol suite (named after two of the most important protocols in it - the Transmission Control protocol (TCP) and the Internet protocol (IP).
- The TCP/IP protocol suite is actually a stack of protocols, working together at different layers.

Layer	Protocols			Purpose
Application layer	HTTP	FTP	SMTP	Protocols specific to applications
Transport layer	TCP	UDP		TCP ensures packets are reliably sent to another device
Network layer	IP			IP directs packets to a specific computer using an IP address
Hardware layer	Ethernet			Converts binary packet data to network signals and back

How does the Internet work?

- When Computer A wants to send message to Computer B, a call is initiated via the Application layer **downward**.
- In the Transport layer, **TCP protocol** splits the message into small chunks, called **packets**. Packets have headers and footers that tell computers what's in the packet and how the information fits with other packets to create an entire file.
- At this stage, packet are also assigned a **port number**. Since many programs are sending messages via TCP/IP stack, we know **which program on the destination computer needs to receive the message**, because it is listening on a **specific port**.

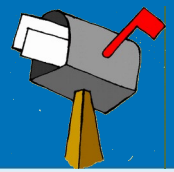


How does the Internet work?

- After going through the TCP layer, the packets proceed to the Network (IP protocol) layer. Here each packet receives a **destination address**, called **IP**. IP addresses identify both **the network** and the destination computer (**host**), thus the capability of establishing a path (connection) to that host.
- You can think of IP addresses and ports like appartment houses with different tenants. Each computer (house) in the network has many programs (tenants) running on it. If you want to send a message, you need to know not only the IP (house address), but also the port (appartment number), in order for the message to reach the desired program (tenant).

Physical address: London, 11 Highway Str.; ap.8

IP address: 96.112.48.21:80



How does the Internet work?

- Each device connected to the Internet has an IP address. This is how one machine can find another through the massive network.
- The version of IP most of us use today is **IPv4**, which is based on a 32-bit address system.
- However, as IPv4 addresses became insufficient, a newer version was also developed in the 1990s - **IPv6**, a 128-bit address system.

SOFT ACADEMY

IPv6

128-bit address

340 undecillion
possible addresses

Example:
2002:db8::8a3f:362:7897

IPv4

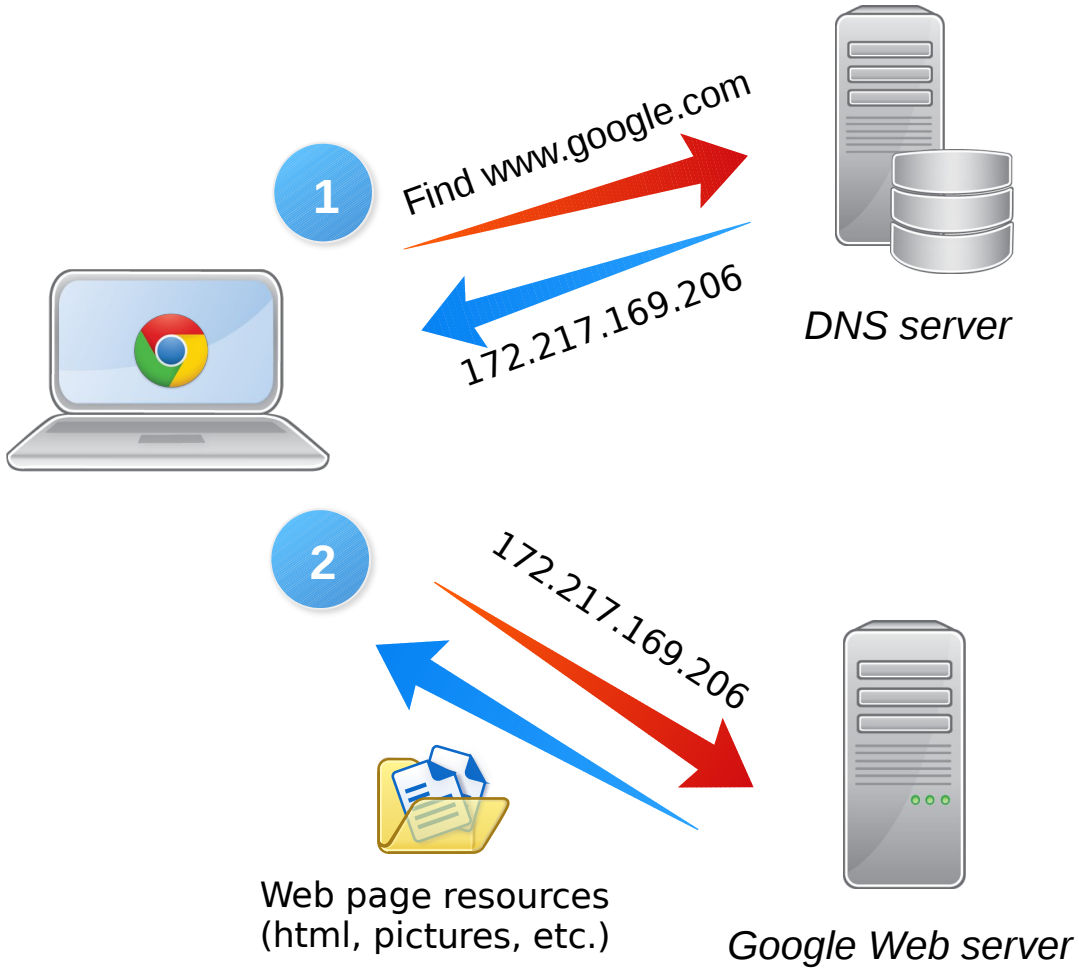
32-bit address

4.3 billion
possible addresses

Example:
192.0.1.246

How does the Internet work?

- But what if we don't know the IP address of the computer we want to connect to? And how can one remember all those numbers?
- This is why IP addresses are presented with a human readable name, called a **domain name.** This way, when we want to access a website, we write its domain name in the browser (e.g. www.google.com), instead of its IP address (172.217.169.206).
- But what actually happens when you type www.google.com in the browser?
- To answer this question comes the **Domain Name Service** - **DNS**. The DNS is a distributed database which keeps track of computer's names and their corresponding IP addresses on the Internet.Many computers connected to the Internet host part of the DNS database and the software that allows others to access it. These computers are known as **DNS servers**.



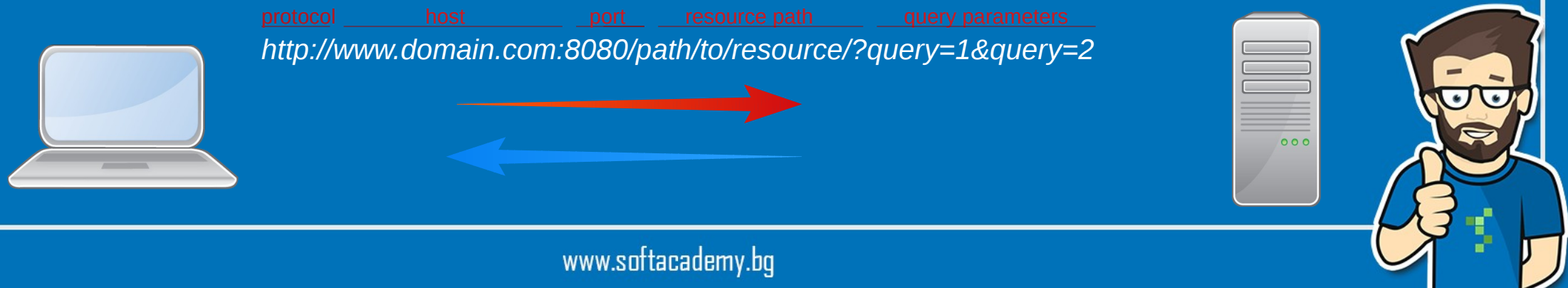
Simplified model of DNS

Understanding HTTP Protocol

- Now that we know how messages travel through the Internet with the help of the TCP/IP protocol, its time to understand the most important part in terms of the software – how web applications work and communicate.
- Web applications use the application level protocols of the TCP/IP protocol suite to communicate over the Internet. These include FPT (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol) and many others. The most important and heavily used one, however, is the **Hypertext Transfer Protocol (HTTP).**
- HTTP is the application protocol used by web browsers and services to transfer all kinds of information (web pages, data, videos, images, etc.)

Understanding HTTP protocol

- HTTP communication is based on request/response pairs. The client (e.g. web browser) initiates a connection to the host (server) and sends an **HTTP request** (e.g. to open a web page); the host returns an **HTTP response** (e.g. an HTML file of the requested web page).
- The request message is sent via **Uniform Resource Locator (URL)**. The URL is nothing more than the address for a given resource (e.g. a web page, or image) on the web.
- The **default HTTP port is 80** (no need to specify it in the URL), but one can be set explicitly, as illustrated in the below image



Understanding HTTP protocol

- Another important part of each HTTP request is the **verb**. HTTP protocol defines several verbs, which instruct the host what action to perform with the request.
- The most used verbs are:

  GET: *fetch an existing resource. The URL contains all the necessary information the server needs to locate and return the resource*

  POST: *create a new resource. POST requests usually carry a payload (additional instruction) that specifies the data for the new resource*

  PUT: *update an existing resource*

  DELETE: delete an existing resource
- HTTP requests also have request headers – meta data about request content

Understanding HTTP protocol

- With URLs and verbs, the client can initiate requests to the server. In return, the server sends HTTP response, containing **status code, request headers** and **message payload**.
- The status code is important and tells the client how to interpret the server response. Status codes are predefined by the HTTP protocol and separated into several groups (e.g. 200 OK; 404 Page Not Found; 500 Internal server error; etc.)
- Response payload contains the actual information requested from the server (e.g. web page html, raw data, etc.) or specific response message
- Response headers provide meta data on the response content

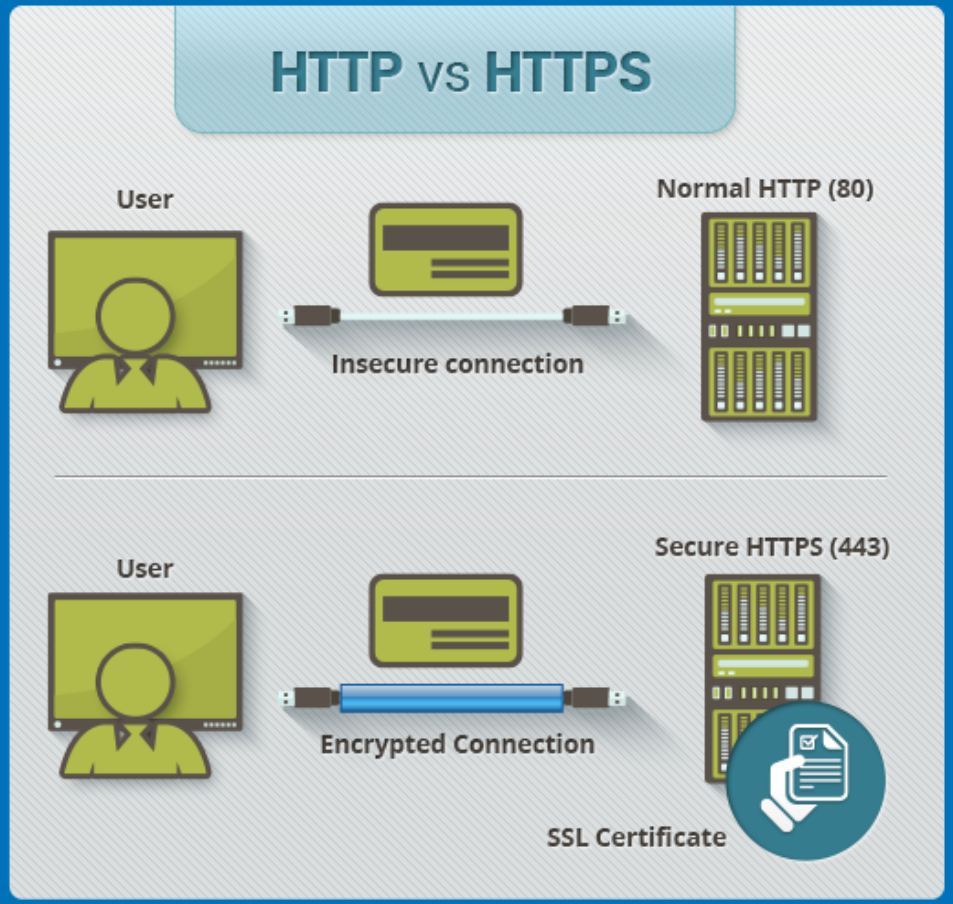
Request		Response
GET /page.html HTTP/1.1	***Start line***	HTTP/1.1 **200 OK**
Accept: text/* Accept-Language: en-US	***Headers***	Content-type: text/html Access-Control-Allow-Origin: *
	Body (payload)	<html> <header>The page</header> <body>My webpage</body> </html>

HTTPS protocol

- HTTPS is a secure version of HyperText Transfer Protocol. The ‘S’ stands for Secure. It is a method for ensuring secure communication between the client and the server
- With HTTPS client and server use special encryption protocols, such as Secure Sockets Layer (**SSL**) or Transport Layer Security (**TLS**) to establish secure connection and exchange encrypted data
- This prevents a man-in-the-middle attack, when a hacker can trace the HTTP traffic and read the request/response data

HTTP vs HTTPS

- HTTPS is the de-facto standard of the web today, with almost every site using it, insted of standard HTTP
- Modern browsers even have special indications/icons to show when connection to a web site is not secure (i.e. web server hosting the site uses HTTP instead of HTTPS)
- HTTPS default port is 443



Transfering data – XML & JSON

- Apart from web pages, images, etc, client and server can also exchange data
- To be able to understand web services, first we need to know something about data formats
- Two of the most common formats used to exchange data over the Internet are XML & JSON

XML

- XML stands for e**X**tensible **M**arkup **L**anguage
- XML is a markup language much like HTML
- XML, however, is designed to carry data, not to display data (i.e. a meta language)
- XML tags are not predefined
- XML is designed to be self-descriptive

XML example:

```
<employees>
	<employee>
		<firstName>John</firstName>
		<lastName>Doe</lastName>
	</employee>
	<employee>
		 <firstName>Anna<firstName>
		<lastName>Smith</lastName>
	</employee>
	<employee>
		<firstName>Peter<firstName>
		<lastName>Jones</lastName>
	</employee>
</employees>
```

JSON

- **JSON** stands for **JavaScript Object Notation**
- JSON is a syntax for storing and exchanging data
- JSON is is human readable, but very easy for machines to parse and generate

- Compared to XML, it is much lighter, flexible and easy to read/write
- It also supports wider variety of data types (String, number, **array**, JSON Object, boolean, null)

JSON Example:

```
{
"employees":
[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]
}
```

What is a Web Service?

- Now that we already know how Internet works, how applications (such as a web browser and a web server) communicate with each other over HTTP protocol, and how data can be transfered in different formats, it is time to dig in the concept of web services
- A web service is simply a piece of software (application), accessible via the Internet (over HTTP protocol), which allows other software applications to get or manipulate data (remember GET, POST, PUT, DELETE operations?)
- Web services differ from web sites, as they are designed to provide information consumable by other software applications (through JSON or XML format), rather than humans
- Web services are not dependent on technology, platform or program language, thus providing universal ways for diffrent applications to exchange data over the Internet

SOFT ACADEMY

Web services and APIs

- Web services are often referred to as APIs
- **API** stands for **Application Program Interface**
- An API is a software interface (like user interface, but for software programs) that allows applications to talk to each other and exchange/manipulate data
- Sounds familiar? This is because in practice all web services are APIs (since they expose application's data and/or functionality)
- Not all APIs, however, are web services. This is because API is a much more general term. For example, an API that is not a **web API** may not require network (as all web services do), or specific protocols to work with it
- Web APIs (or web services) are all around us. They make the web what it is today

Types of web services

- The two main types of web services are SOAP and REST
- **SOAP** (Simple Object Access Protocol) is an XML-based Web service protocol that allows exchange of data and documents over HTTP or SMTP. SOAP web services work only with XML
- **REST** (Representational State Transfer) is actually a software architectural style that defines a set of constraints to be used for creating Web services. Web services that conform to the REST architectural style are called **RESTful Web services**.
- RESTful web services (or REST APIs) support more data formats (including JSON and XML). They are lightweight, scalable, and overall preferred architecture for cloud-based applications than SOAP

RESTful web services

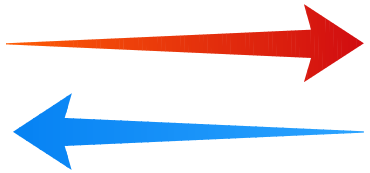
- REST messages contain following components:

  Resource Path
  HTTP Verb
  Header
  Body

- Sounds familiar? These are actually the same as HTTP request/response message components
- Let us see how a REST API request and response look like

Client

Request
POST https://weather.com/api/weather
Content-type: application/json
{“city”: “Rome”, “units”: “C”}



Response
Status: 200 OK
Content-type: application/json
{“low”: 16, “high”: “26”, “humidity”: “86”}

Server hosting
weather.com
web service

Testing RESTful web services with Postman

- As a web service is something that has no user interface, we can not interact with it without special tools
- Such tools allow composing requests and receiving responses from the web service through a user interface, and are often referred to as REST clients
- There are many REST clients available, but the most popular and widely used by both developers and testers is Postman
- Postman provides intuitive user interface, useful testing features (including writing and running automated test scripts) and lots of additional features to help build and test REST APIs

References

- Working with Postman
- JSON Tutorial
- JSON Path Helper

SOFT ACADEMY