LẬP TRÌNH MẠNG

WEB SERVICE

hungdn@ptit.edu.vn

Nội dung

- Giới thiệu web service
  - SOAP
  - WSDL
  - UDDI
- Làm việc với web service trong Java
  - JAX-RPC
  - JAX-WS
  - JAX-RS

1. Web services

Khái niệm

- Ý tưởng ban đầu
  - Một ứng dụng web sử dụng các công nghệ Web để cung cấp chức năng tới người dùng cuối
  - Một dịch vụ web sử dụng các công nghệ Web để cung cấp chức năng cho các ứng dụng khác
- #
  - Web service: Web + Service

Khái niệm (Cont.)

- A Web service is a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols.
- Loosely coupled, reusable software components that semantically encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols.

Web Services technology

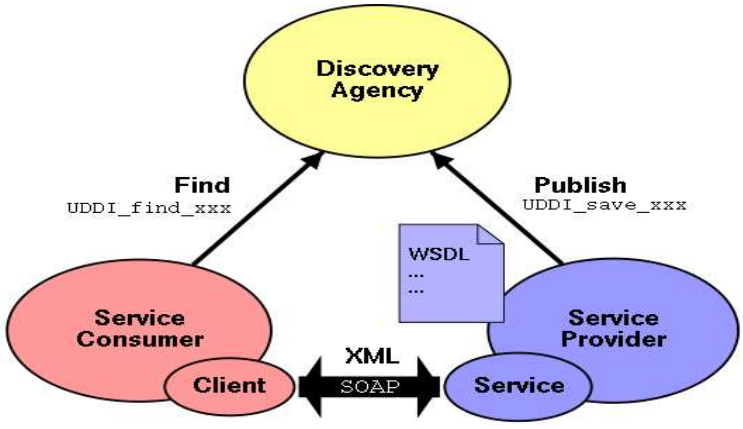
- 3 major Web services toolkits being used widely
  - .NET Web services: Developed by Microsoft and is an integral part of the complete .NET framework. Integrated and easy to use with Visual Studio .NET. services are hosted on IIS web servers.
  - Java Web services: Sun's Web service implementation for the Java community. Comes bundled in a complete Java Web services Development Pack (JWSDP Ver 1.3) including Tomcat web server.
  - Apache Axis: Initially developed by IBM and donated to the Apache group. One of the earliest and stable Web service implementation. Runs on Apache Web servers.

Advantages of Web service

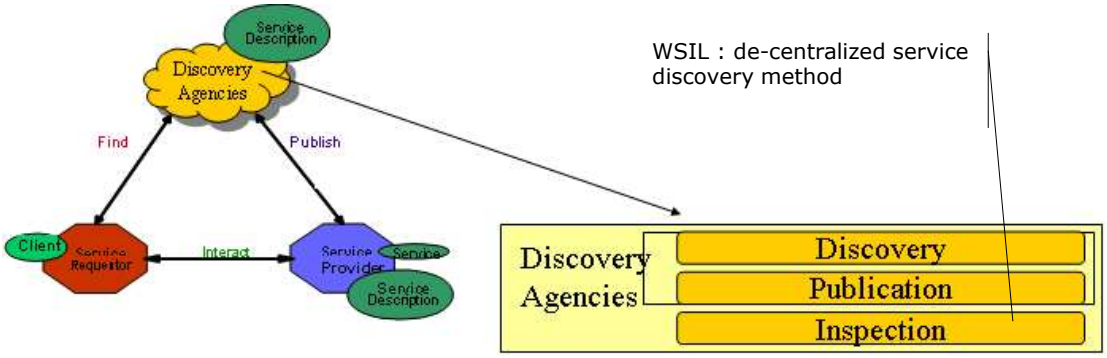
- Not based on a programming language:
  Java, .Net, C, C++, Python, Perl, …
- Not based on a programming data model:
  objects vs non-objects environments.
- Convergence of SOA (Service-Oriented Architecture) and Web.
- Based on web technologies
- Do not need huge framework of memory
- Basic usage is B2B ,remote controlled devices,internal external appl communications

Web service Architecture

- An architecture view based on SOAP, WSDL, and UDDI.

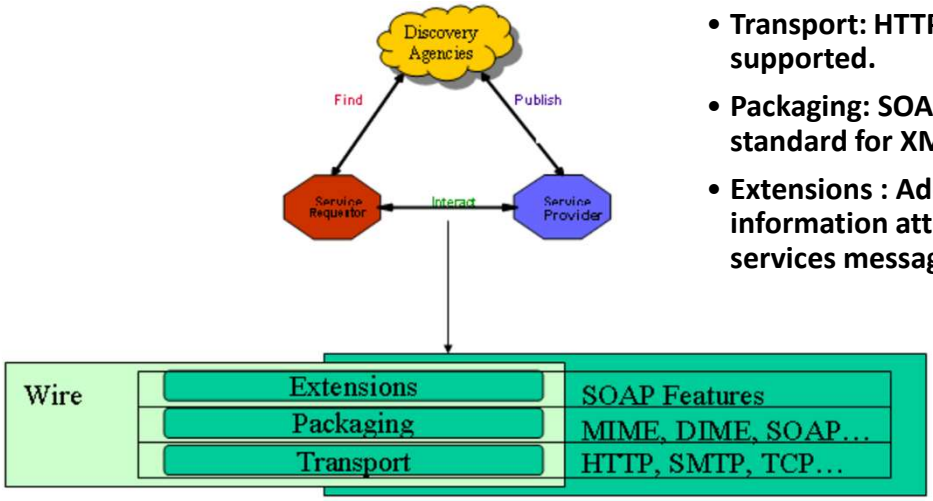


The Discovery Stack



- **Service can be published using a variety of mechanisms:**
  - Direct publish: description sent directly to requestor;
  - WSIL : HTTP GET retrieves descriptions from URL;
  - **U**niversal **D**escription, **D**iscovery and **I**ntegration (UDDI) registries: a Web-based distributed directory.

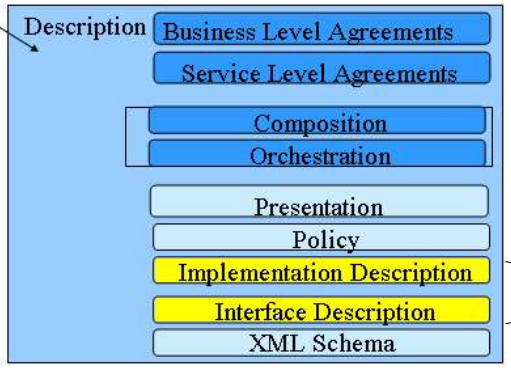
The Wire Stack



- **Transport: HTTP , other may be supported.**
- **Packaging: SOAP is the default standard for XML messaging.**
- **Extensions : Additional information attached to web services messages.**

The Description Stack

- **It is actually a stack of description documents defined using XML Schema.**



Satisfied by WSDL

The technology so far

- **The WS technology is completely based on XML. Therefore, both the data format and the interaction protocols are XML-based:**
  - customized XML -> data format
  - SOAP -> communication protocol
  - WSDL -> the Interface definition language
  - WSIL/UDDI -> standards for services discovery
- **The lowest-level layers (the transport layer) should exploit some existing Internet protocols, like HTTP or SMTP**

Web Service Description Language

- WSDL is a standard format to describe a Web Service (description stack)
- A WSDL document is composed by two sections:
  - An abstract interface section -> like in traditional IDL, it defines the signatures of procedures (RPC-style) or messages (document-style)
  - A deployment section -> it defines the service location and the supported transport protocols
- Fundamentally a client uses the WSDL to create the stub or to dynamically decode messages.

Web Service inspection Language

- WSIL and UDDI are the standard way to search Web Services. (Discovery stack)
  - WSIL is the decentralized approach.
- Fundamentally a WSIL document contains a directory of the Web Services deployed on a server.
- It is anologous to the index.html document for web pages.
- In the future, specific crawlers will browse the Internet looking for WSIL documents, like Google does today for web pages.

Ex: WSIL

```
<inspection>
  <abstract>Acme Industries Public Web Services</abstract>
  <service>
      <name>Store Finder Service</name>
      <abstract>
        A service to perform a geographical search of Acme stores.
      </abstract>
      <description
  location="http://example.org/services/storefinder.wsdl"/>
  </service>
  <link location="http://example.org/services/ecommerce.wsil"/>
</inspection>
```

Service name

Service location and description

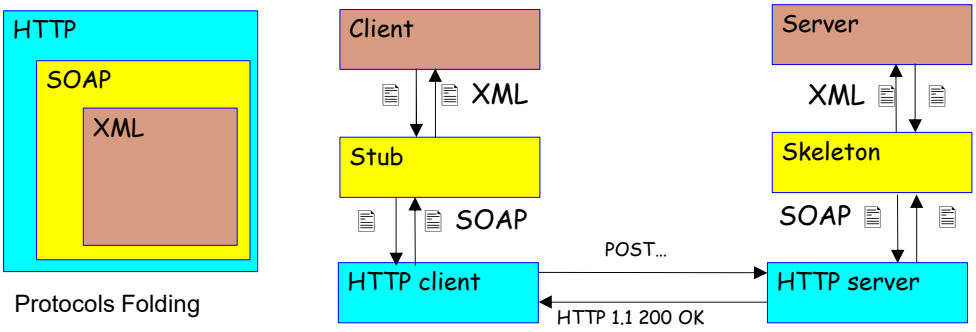
Link to an other WSIL page

UDDI

- UDDI is a complimentary approach for searching based on a centralized repository.
- The repository is an “electronic yellow pages” for firms that offer web services online. Besides the names of services and their WSDL descriptors, firms can add a description of their business, phone numbers, addresses...
- UDDI repositories are offered by many agencies - e.g. IBM, Microsoft and HP.

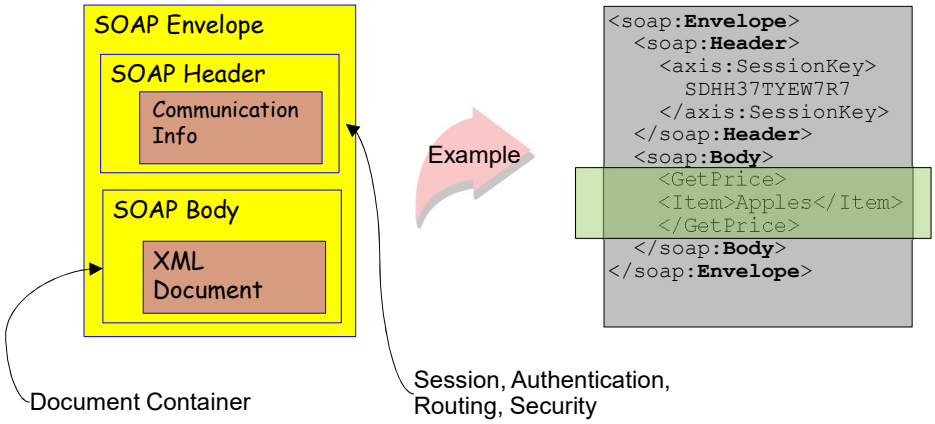
Simple Object Access Protocol

- SOAP is a technology to support the exchange of XML-coded messages over a transport protocol, such as HTTP and SMTP. (*wire stack*)



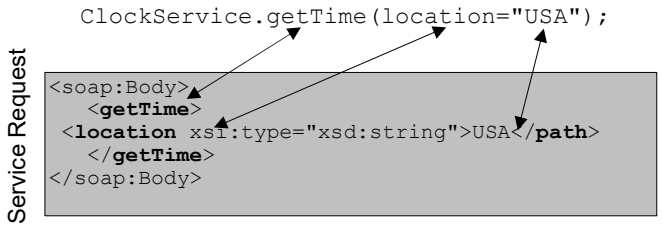
Simple Object Access Protocol

- A SOAP runtime engine basically adds a XML envelope to an existing XML document



SOAP Encoding

- The rules:
  - method name -> first level element in the SOAP Body
  - arguments identifiers -> second level elements
  - arguments values -> third level elements
  - arguments types -> attribute xsi:type



Web service Transport

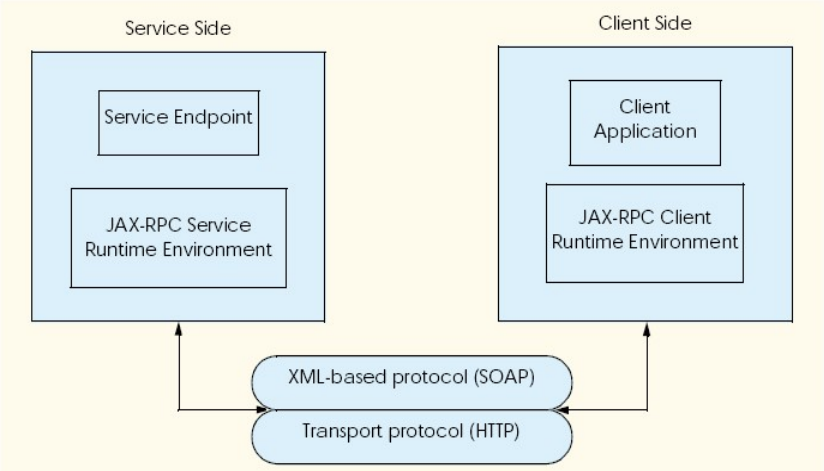
- **Data format**
  - XML (subset of XML 1.0), URL encoding.
  - Data format schema definition: XML Schema
- **Wire format**
  - XML Protocol (XML-RPC, SOAP), URI
  - Transfer protocol: HTTP, SMTP, JMS, BEEP, …
- **WS built on existing standards**
  - Extensible Markup Language (XML)
  - The HTTP (Hypertext Transfer Protocol) standard is allowing more systems to communicate with one another.
  - SOAP (Simple Object Access Protocol) (built on XML) standardizes the messaging capability on different systems.
  - UDDI (Universal Description, Discovery, and Integration) standardizes the publishing and finding of Web services.
  - WSDL (Web Services Description Language) standardizes the description of Web services so providers and requesters are speaking the same language

2. Làm việc với web service trong Java

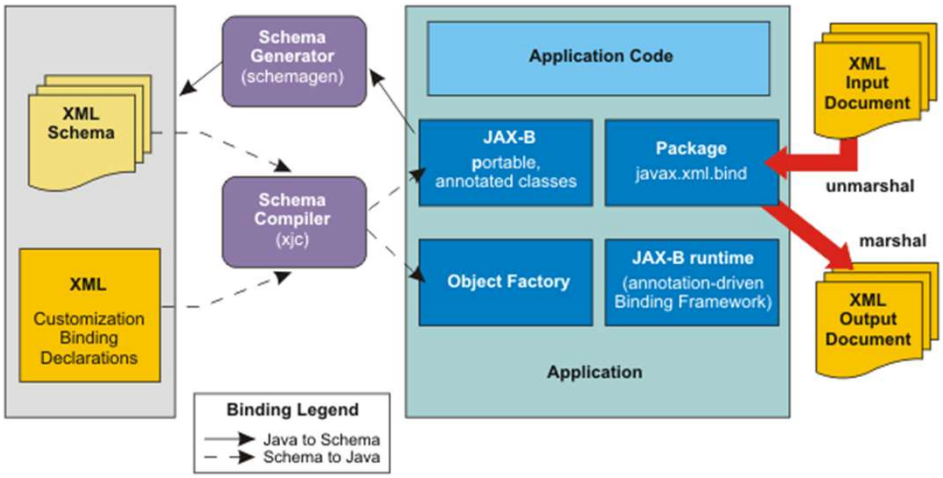
- JAX-RPC (J2EE 1.4)
- JAX-WS (J2EE 5.0)
- JAX-RS (J2EE 6 – J2EE 7.0)

JAX-RPC Architecture

- **To aid developers in building XML-based requests such as SOAP requests, the Java APIs for XML based RPC (JAX/RPC) is used.**
- **JAX/RPC is used for sending and receiving (including marshalling and unmarshalling) method calls using XML-based protocols such as SOAP, or others such as XMLP (XML Protocol).**
- **JAX/RPC isolates you from the specifics of these protocols, enabling rapid application development. There is no longer any need for developers to interact directly**

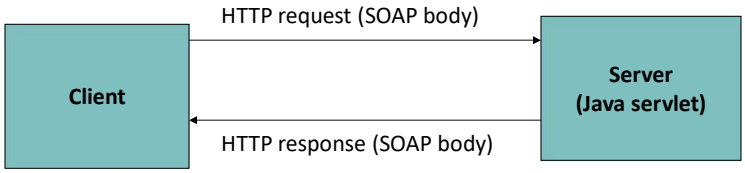


Java API for XML Web Services (JAX-WS)



Hoạt động web service

- Web services conceptually are just specialized web applications



- Java Web Services Developer Pack (JWSDP) `wscompile` tool can implement a Java API from a WSDL

Ex: Run server

- **Visit http://localhost:8080/converter/currency**

Ex: Client

- **Goal: write a JSP-based client**
  - Input: currency and value
  - Output: table of equivalent values

Demo

- **Web Service**
  - Java SE
  - Java EE
- **Web client**
  - Java SE
  - Java EE
- **Java EE**

WSDL Example

```
<?xml version="1.0" encoding="UTF-8"?>

<definitions name="HistoricCurrencyConverter"
  targetNamespace="http://tempuri.org/wsdl"
  xmlns:tns="http://tempuri.org/wsdl"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:ns2="http://tempuri.org/types"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
```

Namespaces specified in config files

WSDL namespaces

XML Schema NS

- Target namespace: namespace for names (*e.g.*, of operations) defined by the WSDL

WSDL Example

```
<types>
  <schema
    targetNamespace="http://tempuri.org/types"
    xmlns:tns="http://tempuri.org/types"
    xmlns:soap11-enc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns="http://www.w3.org/2001/XMLSchema">
    <import
      namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
    <complexType name="ExchangeValues">
      <sequence>
        <element name="dollars" type="double"/>
        <element name="euros" type="double"/>
        <element name="yen" type="double"/>
      </sequence>
    </complexType>
  </schema>
</types>
```

Namespace for data type definitions (ns2 in rest of document)

Defines struct using XML Schema

WSDL Example

Data type defined by XML Schema

```
<message name="CurCon_fromDollars">
  <part name="double_1" type="xsd:double"/>
</message>
<message name="CurCon_fromDollarsResponse">
  <part name="result" type="ns2:ExchangeValues"/>
</message>
<message name="CurCon_fromEuros">
  <part name="double_1" type="xsd:double"/>
</message>
<message name="CurCon_fromEurosResponse">
  <part name="result" type="ns2:ExchangeValues"/>
</message>
<message name="CurCon_fromYen">
  <part name="double_1" type="xsd:double"/>
</message>
<message name="CurCon_fromYenResponse">
  <part name="result" type="ns2:ExchangeValues"/>
</message>
```

Input messages (parameter lists)

Output messages (response data types)

WSDL Example

```
<portType name="CurCon">
  <operation name="fromDollars" parameterOrder="double_1">
    <input message="tns:CurCon_fromDollars"/>
    <output message="tns:CurCon_fromDollarsResponse"/>
  </operation>
  <operation name="fromEuros" parameterOrder="double_1">
    <input message="tns:CurCon_fromEuros"/>
    <output message="tns:CurCon_fromEurosResponse"/>
  </operation>
  <operation name="fromYen" parameterOrder="double_1">
    <input message="tns:CurCon_fromYen"/>
    <output message="tns:CurCon_fromYenResponse"/>
  </operation>
</portType>
```

WSDL Example

```
<binding name="CurConBinding" type="tns:CurCon">
  <operation name="fromDollars">
    <input>
      <soap:body
         encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
         use="encoded" namespace="http://tempuri.org/wsdl"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        use="encoded" namespace="http://tempuri.org/wsdl"/>
    </output>
    <soap:operation soapAction=""/>
  </operation>
    ...
  </operation>
  <soap:binding
    transport="http://schemas.xmlsoap.org/soap/http"
    style="rpc"/>
</binding>
```

Implement the operations using SOAP encoding of data structures and RPC (JWSDP defaults)

WSDL Example

```
<service name="HistoricCurrencyConverter">
  <port name="CurConPort" binding="tns:CurConBinding">
    <soap:address location="REPLACE_WITH_ACTUAL_URL"/>
  </port>
</service>

</definitions>
```

Replaced by server
when WSDL is visited

WSDL Example

- Summary:
  - `types` uses XML Schema to define data types
  - `message` elements define parameter lists and return types using `types` and XML Schema
  - portType defines abstract API for `operation`'s using `message`'s
  - `binding` specifies how `message`'s will be communicated and `operation`'s called
  - `service` associates URL with `binding`

XML Schema

- How do we send a Java `double` value to a web service using XML?
  - Is scientific notation allowed?
  - How large can the value be?
  - *Etc.*
- What if we want to send an object?
  - And what if the object contains references to other objects?

XML Schema

- XML Schema addresses such questions
  - Defines a number of simple data types, including
    - Range of allowed values
    - How values are represented as strings
  - Provides facilities for defining data structures in terms of simple types or other data structures
- Can also be used in place of XML DTD

XML Schema

- Built-in data types
  - Types corresponding to Java primitive types: `boolean`, `byte`,`int`, `double`, *etc.*

Built-in type

```
<part name="latitude" type="xsd:decimal" />
```

    - String representations much as Java
      - Exception: can use `0` for `false`, `1` for `true`
    - No `char`; use `string` instead
  - XML DTD types (`ID`, `CDATA`, *etc.*)

XML Schema

- Built-in data types
  - `integer` and `decimal` (arbitrary precision)
  - dates, times, and related subtypes
  - URLs
  - XML namespace qualified names
  - binary data
  - some restricted forms of the above, *e.g.*, `nonNegativeInteger`

XML Schema

- XML Schema namespace defining built-in types is called the document namespace
  - Standard prefix for this namespace is `xsd`

```
http://www.w3.org/2001/XMLSchema
```

XML Schema

TABLE 9.1: JAX-RPC mappings between supported Java classes and XML Schema built-in data types.

Java Class	XML Schema Type
String	string
java.math.BigDecimal	decimal
java.math.BigInteger	integer
java.util.Calendar	dateTime
java.util.Date	dateTime
java.xml.namespace.QName	QName
java.net.URI	anyURI

- Plus Java primitive types (`int`, *etc.*)

XML Schema

- Mapping from XML Schema data types to Java:
  - Primitives: one-for-one mapping
  - `date`, `time`, `dateTime`: map to `Calendar`
  - most others: map to `String`

XML Schema

- Elements in the document namespace can declare user-defined data types
- Two XML Schema data types:
  - Complex: requires markup to represent within an XML document
  - Simple: can be represented as character data

XML Schema

- User-defined data types are declared in the `types` element of a WSDL
  - Example: `ExchangeValue`
- In WSDL, user-defined types can be used
  - To define other data types within `types` element
  - To specify data types of parameters and return values in `message` elements

XML Schema

```
<types>
  <schema
    targetNamespace="http://tempuri.org/types"
    xmlns:tns="http://tempuri.org/types"
    xmlns:soap11-enc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns="http://www.w3.org/2001/XMLSchema">
    <import
      namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
    <complexType name="ExchangeValues">
      <sequence>
        <element name="dollars" type="double"/>
        <element name="euros" type="double"/>
        <element name="yen" type="double"/>
      </sequence>
    </complexType>
  </schema>
</types>
```

XML Schema

- An XML schema is markup that
  - Is written according to the XML Schema vocabulary
  - Defines an XML vocabulary
- A schema document is an XML document consisting entirely of an XML schema
- A document conforming with an XML schema vocabulary is call an instance of the schema

XML Schema

- Root element of the markup of an XML schema is `schema`
- Define data types with elements:
  - complexType
  - simpleType
- An XML schema can also define other vocabulary aspects (allowed elements, element content) that we won't cover

XML Schema

- One way to define simple types: restrict an existing simple base type

```
<simpleType name="memberType">
  <restriction base="string">
    <enumeration value="platinum" />
    <enumeration value="preferred" />
    <enumeration value="gold" />
    <enumeration value="member" />
  </restriction>
</simpleType>
```

Base type

XML Schema

- Built-in types all have facets, that is, aspects that can be restricted
  - `enumeration` is a facet that applies to all built-in types except `boolean`
  - `length`, `minLength`, `maxLength` apply to string-like types (*e.g.*, `string`, `QName`, `anyURI`)
  - `minInclusive`, `maxInclusive`, `minExclusive`, `maxExclusive` apply to numeric and time-oriented types
  - `totalDigits`, `fractionDigits` apply to numeric types

XML Schema

- Restricting multiple facets:

```
<simpleType name="priorityType">
  <restriction base="int">
    <minExclusive value="10" />
    <maxInclusive value="100" />
  </restriction>
</simpleType>
```

XML Schema

- `pattern` facet
  - applies to most types (except a few DTD)
  - specifies regular expression

```
<simpleType name="phoneNumType">
  <restriction base="string">
    <pattern value="\d{3}-\d{3}-\d{4}" />
  </restriction>
</simpleType>
```

XML Schema

- Other simple types
  - Union: combine two or more types

```
<simpleType name="oddType">
  <union memberTypes="memberType phoneNumType" />
</simpleType>
```

  - Lists of values of simple type

```
<simpleType name="intList">
  <list itemType="int" />
</simpleType>
```

XML Schema

- Complex type
  - Defined in an XML schema

```
<complexType name="ExchangeValues">
  <sequence>
    <element name="dollars" type="double"/>
    <element name="euros" type="double"/>
    <element name="yen" type="double"/>
  </sequence>
</complexType>
```

  - Used in an instance document

```
<anExchangeValue xsi:type="ExchangeValues">
  <dollars>1.0</dollars>
  <euros>0.746826</euros>
  <yen>102.56</yen>
</anExchangeValue>
```

XML Schema

- Complex type can be used in placed of XML DTD content specification
  - sequence element is equivalent to , operator in DTD

```
<!ELEMENT anExchangeValue (dollars, euros, yen)>
```

```
<complexType name="Arguments">
  <sequence>
    <element name="optArg" type="string"
             minOccurs="0" />
  </sequence>
</complexType>

<complexType name="ExchangeValues">
  <all>
    <element name="dollars" type="double"/>
    <element name="euros" type="double"/>
    <element name="yen" type="double"/>
  </all>
</complexType>
```

XML Schema

- Instance namespace

```
http://www.w3.org/2001/XMLSchema-instance
```

  - Normally associated with prefix xsi
- Used within instance documents to
  - define null-valued elements

```
<optArg xsi:nil="true"></optArg>
```

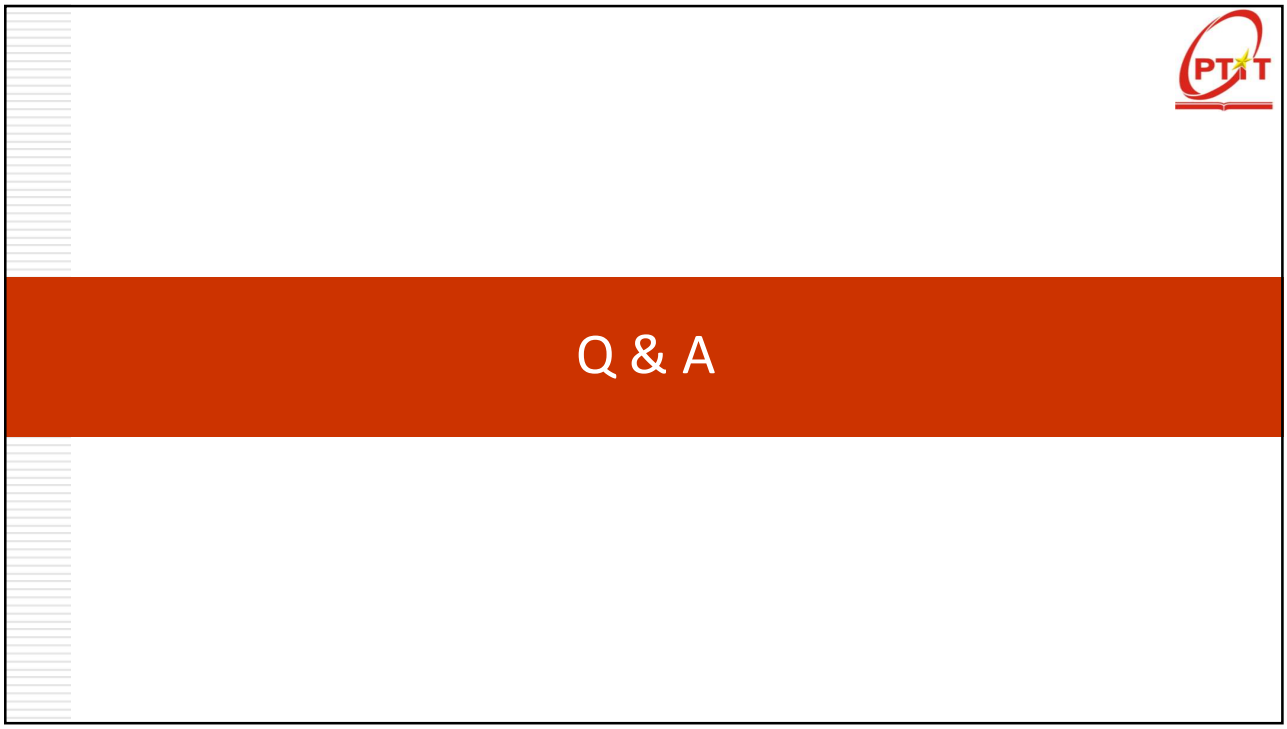
  - define data types

```
<latitude xsi:type="xsd:decimal">40.28</latitude>
<longitude xsi:type="xsd:decimal">-79.49</longitude>
```

Exercise

- Viết web service sử dụng JAX-WS cung cấp
  - Phương thức đảo ngược chuỗi
  - Phương thức kiểm tra thông tin người dùng (username, password)
- Client có thể sử dụng
  - Java SE
  - Java EE (web)
- **Web service**
  - Java SE
- **Web service**
  - Java EE 6
  - Java EE 7

Tổng kết


Q & A