WSDL - Web Services Description Language

- WSDL stands for Web Services Description Language.
- WSDL is written in XML.
- WSDL is an XML document.
- WSDL is used to describe Web services.
- WSDL is also used to locate Web services.
- WSDL is not yet a W3C standard.
- WSDL Describes Web Services.
- WSDL stands for Web Services Description Language.
- WSDL is a document written in XML. The document describes a Web service. It specifies the location of the service and the operations (or methods) the service exposes.

The WSDL Document Structure

A WSDL document describes a web service using these major elements:

<table>
<thead>
<tr>
<th>Element</th>
<th>Defines</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;portType&gt;</td>
<td>The operations performed by the web service</td>
</tr>
<tr>
<td>&lt;message&gt;</td>
<td>The messages used by the web service</td>
</tr>
<tr>
<td>&lt;types&gt;</td>
<td>The data types used by the web service</td>
</tr>
<tr>
<td>&lt;binding&gt;</td>
<td>The communication protocols used by the web service</td>
</tr>
</tbody>
</table>

```xml
<definitions>
  <types>
    definition of types........
  </types>

  <message>
    definition of a message....
  </message>

  <portType>
    definition of a port.......  
  </portType>

  <binding>
    definition of a binding....
  </binding>

</definitions>
```

A WSDL document can also contain other elements, like extension elements and a service element that makes it possible to group together the definitions of several web services in one single WSDL document.

- The `<portType>` element is the most important WSDL element.
- It describes a web service, the operations that can be performed, and the messages that are involved.
- The `<portType>` element can be compared to a function library (or a module, or a class) in a traditional programming language.
- The `<message>` element defines the data elements of an operation.
- Each message can consist of one or more parts. The parts can be compared to the parameters of a function call in a traditional programming language.
- The `<types>` element defines the data type that are used by the web service.
- For maximum platform neutrality, WSDL uses XML Schema syntax to define data types.
- The `<binding>` element defines the message format and protocol details for each port.
This is a simplified fraction of a WSDL document:

```xml
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>
```

In this example the `<portType>` element defines "glossaryTerms" as the name of a port, and "getTerm" as the name of an operation.

The "getTerm" operation has an input message called "getTermRequest" and an output message called "getTermResponse".

The `<message>` elements define the parts of each message and the associated data types.

Compared to traditional programming, glossaryTerms is a function library, "getTerm" is a function with "getTermRequest" as the input parameter and getTermResponse as the return parameter.

**WSDL Ports**

A WSDL port describes the interfaces (legal operations) exposed by a web service.

- The `<portType>` element is the most important WSDL element.
- It defines a web service, the operations that can be performed, and the messages that are involved.
- The port defines the connection point to a web service. It can be compared to a function library (or a module, or a class) in a traditional programming language. Each operation can be compared to a function in a traditional programming language.

**Operation Types**

The request-response type is the most common operation type, but WSDL defines four types:

<table>
<thead>
<tr>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-way</td>
<td>The operation can receive a message but will not return a response</td>
</tr>
<tr>
<td>Request-response</td>
<td>The operation can receive a request and will return a response</td>
</tr>
<tr>
<td>Solicit-response</td>
<td>The operation can send a request and will wait for a response</td>
</tr>
<tr>
<td>Notification</td>
<td>The operation can send a message but will not wait for a response</td>
</tr>
</tbody>
</table>

**One-Way Operation**

A one-way operation example:

```xml
<message name="newTermValues">
  <part name="term" type="xs:string"/>
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="setTerm">
    <input name="newTerm" message="newTermValues"/>
  </operation>
</portType>
```

In this example the port "glossaryTerms" defines a one-way operation called "setTerm".
The "setTerm" operation allows input of new glossary terms messages using a "newTermValues" message with the input parameters "term" and "value". However, no output is defined for the operation.

Request-Response Operation

A request-response operation example:

```xml
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>
```

In this example the port "glossaryTerms" defines a request-response operation called "getTerm".

The "getTerm" operation requires an input message called "getTermRequest" with a parameter called "term", and will return an output message called "getTermResponse" with a parameter called "value".

WSDL Bindings

WSDL bindings defines the message format and protocol details for a web service.

Binding to SOAP

A request-response operation example:

```xml
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>

<binding type="glossaryTerms" name="b1">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http" />
  <operation>
    <soap:operation
      soapAction="http://example.com/getTerm"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
```

The **binding** element has two attributes - the name attribute and the type attribute.

The name attribute (you can use any name you want) defines the name of the binding, and the type attribute points to the port for the binding, in this case the "glossaryTerms" port.
The **soap:binding** element has two attributes - the style attribute and the transport attribute.

The style attribute can be "rpc" or "document". In this case we use document. The transport attribute defines the SOAP protocol to use. In this case we use HTTP.

The **operation** element defines each operation that the port exposes.

For each operation the corresponding SOAP action has to be defined. You must also specify how the input and output are encoded. In this case we use "literal".

**WSDL and UDDI**

Universal Description, Discovery and Integration (UDDI) is a directory service where businesses can register and search for Web services.

**What is UDDI**

UDDI is a platform-independent framework for describing services, discovering businesses, and integrating business services by using the Internet.

- UDDI stands for Universal Description, Discovery and Integration
- UDDI is a directory for storing information about web services
- UDDI is a directory of web service interfaces described by WSDL
- UDDI communicates via SOAP
- UDDI is built into the Microsoft .NET platform

**What is UDDI Based On?**

UDDI uses World Wide Web Consortium (W3C) and Internet Engineering Task Force (IETF) Internet standards such as XML, HTTP, and DNS protocols.

UDDI uses WSDL to describe interfaces to web services

Additionally, cross platform programming features are addressed by adopting SOAP, known as XML Protocol messaging specifications found at the W3C Web site.

**UDDI Benefits**

Any industry or businesses of all sizes can benefit from UDDI

Before UDDI, there was no Internet standard for businesses to reach their customers and partners with information about their products and services. Nor was there a method of how to integrate into each other's systems and processes.

Problems the UDDI specification can help to solve:

- Making it possible to discover the right business from the millions currently online
- Defining how to enable commerce once the preferred business is discovered
- Reaching new customers and increasing access to current customers
- Expanding offerings and extending market reach
- Solving customer-driven need to remove barriers to allow for rapid participation in the global Internet economy
- Describing services and business processes programmatically in a single, open, and secure environment

**How can UDDI be Used**

If the industry published an UDDI standard for flight rate checking and reservation, airlines could register their services into an UDDI directory. Travel agencies could then search the UDDI directory to find the airline's reservation interface. When the interface is found, the travel agency can communicate with the service immediately because it uses a well-defined reservation interface.

**Who is Supporting UDDI?**

UDDI is a cross-industry effort driven by all major platform and software providers like Dell, Fujitsu, HP, Hitachi, IBM, Intel, Microsoft, Oracle, SAP, and Sun, as well as a large community of marketplace operators, and e-business leaders.
The full WSDL 1.2 syntax as described in the W3C Working Draft is listed below.

```xml
<wsdl:definitions name="nmtoken" targetNamespace="uri">
   <import namespace="uri" location="uri"/> *
   <wsdl:documentation .... /> ?
   <wsdl:types> ?
      <wsdl:documentation .... /> ?
      <xsd:schema .... /> *
   </wsdl:types>
   <wsdl:message name="ncname"> *
      <wsdl:documentation .... /> ?
      <part name="ncname" element="qname"? type="qname"/>? *
   </wsdl:message>
   <wsdl:portType name="ncname"> *
      <wsdl:documentation .... /> ?
      <wsdl:operation name="ncname"> *
         <wsdl:documentation .... /> ?
         <wsdl:input message="qname"> ?
            <wsdl:documentation .... /> ?
            <wsdl:input>
            </wsdl:input>
         </wsdl:input>
         <wsdl:output message="qname"> ?
            <wsdl:documentation .... /> ?
            <wsdl:output>
            </wsdl:output>
         <wsdl:output>
            <wsdl:fault name="ncname" message="qname"> *
               <wsdl:documentation .... /> ?
               <wsdl:fault>
            </wsdl:fault>
         </wsdl:output>
      </wsdl:operation>
   </wsdl:portType>
   <wsdl:serviceType name="ncname"> *
      <wsdl:portType name="qname"/> +
   </wsdl:serviceType>
   <wsdl:binding name="ncname" type="qname"> *
      <wsdl:documentation .... /> ?
      <!-- binding details --> *
      <wsdl:operation name="ncname"> *
         <wsdl:documentation .... /> ?
         <-- binding details --> *
         <wsdl:input> ?
            <wsdl:documentation .... /> ?
            <-- binding details -->
         </wsdl:input>
         <wsdl:output> ?
            <wsdl:documentation .... /> ?
            <-- binding details --> *
         </wsdl:output>
         <wsdl:output>
            <wsdl:fault name="ncname"> *
               <wsdl:documentation .... /> ?
               <-- binding details --> *
            </wsdl:fault>
         </wsdl:output>
      </wsdl:operation>
   </wsdl:binding>
   <wsdl:service name="ncname" serviceType="qname"> *
      <wsdl:documentation .... /> ?
      <wsdl:port name="ncname" binding="qname"> *
         <wsdl:documentation .... /> ?
         <-- address details -->
      </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```