WSDL 简介

* [WSDL 教程](http://www.w3school.com.cn/wsdl/index.asp)
* [WSDL 文档](http://www.w3school.com.cn/wsdl/wsdl_documents.asp)

**WSDL 是基于 XML 的用于描述 Web Services 以及如何访问 Web Services 的语言。**

**您应当具备的基础知识**

在继续学习之前，您需要对下面的知识有基本的了解：

* XML
* XML 命名空间
* XML Schema

如果您希望首先学习这些项目，请访问我们的 [XML 系列教程](http://www.w3school.com.cn/x.asp)。

**什么是 WSDL?**

* WSDL 指网络服务描述语言
* WSDL 使用 XML 编写
* WSDL 是一种 XML 文档
* WSDL 用于描述网络服务
* WSDL 也可用于定位网络服务
* WSDL 还不是 W3C 标准

**WSDL 可描述网络服务（Web Services）**

WSDL 指网络服务描述语言 (Web Services Description Language)。

WSDL 是一种使用 XML 编写的文档。这种文档可描述某个 Web service。它可规定服务的位置，以及此服务提供的操作（或方法）。

**在 W3C 的 WSDL 发展史**

在 2001 年 3 月，WSDL 1.1 被 IBM、微软作为一个 W3C 记录（W3C note）提交到有关 XML 协议的 W3C XML 活动，用于描述网络服务。

（W3C 记录仅供讨论。一项 W3C 记录的发布并不代表它已被 W3C 或 W3C 团队亦或任何 W3C 成员认可。）

在 2002 年 7 月，W3C 发布了第一个 WSDL 1.2 工作草案。

请在我们的 [W3C 教程](http://www.w3school.com.cn/w3c/w3c_wsdl.asp) 阅读更多有关规范的状态及时间线。

WSDL 文档

* [WSDL 简介](http://www.w3school.com.cn/wsdl/wsdl_intro.asp)
* [WSDL 端口](http://www.w3school.com.cn/wsdl/wsdl_ports.asp)

**WSDL 文档仅仅是一个简单的 XML 文档。**

**它包含一系列描述某个 web service 的定义。**

**WSDL 文档结构**

WSDL 文档是利用这些主要的元素来描述某个 web service 的：

|  |  |
| --- | --- |
| **元素** | **定义** |
| <portType> | web service 执行的操作 |
| <message> | web service 使用的消息 |
| <types> | web service 使用的数据类型 |
| <binding> | web service 使用的通信协议 |

一个 WSDL 文档的主要结构是类似这样的：

<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>

WSDL 文档可包含其它的元素，比如 extension 元素，以及一个 service 元素，此元素可把若干个 web services 的定义组合在一个单一的 WSDL 文档中。

如需完整的语法概述，请访问 [WSDL 语法](http://www.w3school.com.cn/wsdl/wsdl_syntax.asp) 这一节。

**WSDL 端口**

**<portType>** 元素是最重要的 WSDL 元素。

它可描述一个 web service、可被执行的操作，以及相关的消息。

可以把 <portType> 元素比作传统编程语言中的一个函数库（或一个模块、或一个类）。

**WSDL 消息**

**<message>** 元素定义一个操作的数据元素。

每个消息均由一个或多个部件组成。可以把这些部件比作传统编程语言中一个函数调用的参数。

**WSDL types**

**<types>** 元素定义 web service 使用的数据类型。

为了最大程度的平台中立性，WSDL 使用 XML Schema 语法来定义数据类型。

**WSDL Bindings**

**<binding>** 元素为每个端口定义消息格式和协议细节。

**WSDL 实例**

这是某个 WSDL 文档的简化的片段：

<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>

在这个例子中，**<portType>** 元素把 "glossaryTerms" 定义为某个**端口**的名称，把 "getTerm" 定义为某个**操作**的名称。

操作 "getTerm" 拥有一个名为 "getTermRequest" 的**输入消息**，以及一个名为 "getTermResponse" 的**输出消息**。

**<message>** 元素可定义每个消息的**部件**，以及相关联的数据类型。

对比传统的编程，glossaryTerms 是一个函数库，而 "getTerm" 是带有输入参数 "getTermRequest" 和返回参数 getTermResponse 的一个函数。

WSDL 端口

* [WSDL 文档](http://www.w3school.com.cn/wsdl/wsdl_documents.asp)
* [WSDL 绑定](http://www.w3school.com.cn/wsdl/wsdl_binding.asp)

**WSDL 端口可描述由某个 web service 提供的界面（合法操作）。**

**WSDL 端口**

**<portType>** 元素是最重要的 WSDL 元素。

它可描述一个 **web service**、可被执行的**操作**，以及相关的**消息**。

端口定义了指向某个 web service 的连接点。可以把该元素比作传统编程语言中的一个函数库（或一个模块、或一个类），而把每个操作比作传统编程语言中的一个函数。

**操作类型**

请求-响应是最普通的操作类型，不过 WSDL 定义了四种类型：

|  |  |
| --- | --- |
| **类型** | **定义** |
| One-way | 此操作可接受消息，但不会返回响应。 |
| Request-response | 此操作可接受一个请求并会返回一个响应 |
| Solicit-response | 此操作可发送一个请求，并会等待一个响应。 |
| Notification | 此操作可发送一条消息，但不会等待响应。 |

**One-Way 操作**

一个 one-way 操作的例子：

<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 >

在这个例子中，端口 "glossaryTerms" 定义了一个名为 "setTerm" 的 one-way 操作。

这个 "setTerm" 操作可接受新术语表项目消息的输入，这些消息使用一条名为 "newTermValues" 的消息，此消息带有输入参数 "term" 和 "value"。不过，没有为这个操作定义任何输出。

**Request-Response 操作**

一个 request-response 操作的例子：

<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>

在这个例子中，端口 "glossaryTerms" 定义了一个名为 "getTerm" 的 request-response 操作。

"getTerm" 操作会请求一个名为 "getTermRequest" 的输入消息，此消息带有一个名为 "term" 的参数，并将返回一个名为 "getTermResponse" 的输出消息，此消息带有一个名为 "value" 的参数。

WSDL 绑定

* [WSDL 端口](http://www.w3school.com.cn/wsdl/wsdl_ports.asp)
* [WSDL 和 UDDI](http://www.w3school.com.cn/wsdl/wsdl_uddi.asp)

**WSDL 绑定可为 web service 定义消息格式和协议细节。**

**绑定到 SOAP**

一个 **请求 - 响应** 操作的例子：

<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>

**binding** 元素有两个属性 - name 属性和 type 属性。

name 属性定义 binding 的名称，而 type 属性指向用于 binding 的端口，在这个例子中是 "glossaryTerms" 端口。

**soap:binding** 元素有两个属性 - style 属性和 transport 属性。

style 属性可取值 "rpc" 或 "document"。在这个例子中我们使用 document。transport 属性定义了要使用的 SOAP 协议。在这个例子中我们使用 HTTP。

**operation** 元素定义了每个端口提供的操作符。

对于每个操作，相应的 SOAP 行为都需要被定义。同时您必须如何对输入和输出进行编码。在这个例子中我们使用了 "literal"。

WSDL 和 UDDI

* [WSDL 绑定](http://www.w3school.com.cn/wsdl/wsdl_binding.asp)
* [WSDL 语法](http://www.w3school.com.cn/wsdl/wsdl_syntax.asp)

**UDDI 是一种目录服务，企业可以使用它对 Web services 进行注册和搜索。**

**UDDI，英文为 "Universal Description, Discovery and Integration"，可译为“通用描述、发现与集成服务”。**

**什么是 UDDI？**

UDDI 是一个独立于平台的框架，用于通过使用 Internet 来描述服务，发现企业，并对企业服务进行集成。

* UDDI 指的是通用描述、发现与集成服务
* UDDI 是一种用于存储有关 web services 的信息的目录。
* UDDI 是一种由 WSDL 描述的 web services 界面的目录。
* UDDI 经由 SOAP 进行通信
* UDDI 被构建入了微软的 .NET 平台

**UDDI 基于什么？**

UDDI 使用 W3C 和 IETF\* 的因特网标准，比如 XML、HTTP 和 DNS 协议。

UDDI 使用 WSDL 来描述到达 web services 的界面

此外，通过采用 SOAP，还可以实现跨平台的编程特性，大家知道，SOAP 是 XML 的协议通信规范，可在 W3C 的网站找到相关的信息。

**\*注释：**IETF - Internet Engineering Task Force

**UDDI 的好处**

任何规模的行业或企业都能得益于 UDDI。

在 UDDI 之前，还不存在一种 Internet 标准，可以供企业为它们的企业和伙伴提供有关其产品和服务的信息。也不存在一种方法，来集成到彼此的系统和进程中。

UDDI 规范帮助我们解决的问题：

* 使得在成百万当前在线的企业中发现正确的企业成为可能
* 定义一旦首选的企业被发现后如何启动商业
* 扩展新客户并增加对目前客户的访问
* 扩展销售并延伸市场范围
* 满足用户驱动的需要，为在全球 Internet 经济中快速合作的促进来清除障碍

**UDDI 如何被使用**

假如行业发布了一个用于航班比率检测和预订的 UDDI 标准，航空公司就可以把它们的服务注册到一个 UDDI 目录中。然后旅行社就能够搜索这个 UDDI 目录以找到航空公司预订界面。当此界面被找到后，旅行社就能够立即与此服务进行通信，这样由于它使用了一套定义良好的预订界面。

**谁在支持 UDDI？**

UDDI 是一个跨行业的研究项目，由所有主要的平台和软件提供商驱动，比如：Dell, Fujitsu, HP, Hitachi, IBM, Intel, Microsoft, Oracle, SAP, 以及 Sun, 它既是一个市场经营者的团体，也是一个电子商务的领导者。

已有数百家公司参与了这个 UDDI 团体。

完整的 WSDL 语法

* [WSDL 和 UDDI](http://www.w3school.com.cn/wsdl/wsdl_uddi.asp)
* [WSDL 总结](http://www.w3school.com.cn/wsdl/wsdl_summary.asp)

**描述于 W3C 工作草案的完整 WSDL 1.2 语法已列在下面：**

<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:output message="qname"> ?

<wsdl:documentation .... /> ?

</wsdl:output>

<wsdl:fault name="ncname" message="qname"> \*

<wsdl:documentation .... /> ?

</wsdl:fault>

</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:fault name="ncname"> \*

<wsdl:documentation .... /> ?

<-- binding details --> \*

</wsdl:fault>

</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>

# 您已经学习了 WSDL，下一步呢？

* [WSDL 语法](http://www.w3school.com.cn/wsdl/wsdl_syntax.asp)
* [WSDL 教程](http://www.w3school.com.cn/wsdl/index.asp)

## WSDL 概要

本教程已为您讲解了如何创建可描述 web 服务的 WSDL 文档。它也规定了服务的位置和服务所提供的操作（或方法）。

您已经学习到如何为 web 服务定义消息格式和协议细节。

您也学习了可通过 UDDI 来注册和搜索 web 服务。

## 您已经学习了 WSDL，下一步呢？

下一步应该学习 SOAP 和 Web Services。

### SOAP

SOAP 是一种基于 XML 的简易协议，允许应用程序通过 HTTP 来交换信息。

或者更简单地讲，SOAP 是用于访问 web 服务的协议。

如果您希望学习更多有关 SOAP 的知识，请访问我们的 [SOAP 教程](http://www.w3school.com.cn/soap/index.asp)。

### Web Services

Web services 可把您的应用程序转换为 web 应用程序。

通过使用 XML，可以在应用程序间传送消息。

如果您希望学习更多有关 Web services 的知识，请访问我们的 [Web services 教程](http://www.w3school.com.cn/webservices/index.asp)。