

[Web Services 简介 ➔](#)

# Web Services 教程

通过使用 Web Services，您的应用程序可以向全世界发布信息，或提供某项功能。

Web Services 脚本平台需支持 XML + HTTP。

**现在开始学习Web Services!**

## 内容目录

[Web Services 简介](#)

对 Web Services 的简要介绍。

[Why Web Services?](#)

为什么及如何使用 Web Services ？

[Web Services 平台](#)

Web Services 平台之后的组成元素。

[Web Services 实例](#)

一个 ASP.NET 的 Web Services 实例。

[Web Services 总结](#)

本节包括在本教程所学内容的一个总结，以及我们向你推荐的下一步应该学习的内容。

[Web Services 简介 ➔](#)

 [点我分享笔记](#)

[← Web Services 教程](#)[为什么使用 Web Services? →](#)

# Web Services 简介

Web Services 可使您的应用程序成为 Web 应用程序。

Web Services 通过 Web 进行发布、查找和使用。

## 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：

- HTML
- XML

如果您希望首先学习这些项目，请在我们的[首页](#)访问这些教程。

## 什么是Web Services ？

- Web Services 是应用程序组件
- Web Services 使用开放协议进行通信
- Web Services 是独立的（ self-contained ）并可自我描述
- Web Services 可通过使用UDDI来发现
- Web Services 可被其他应用程序使用
- XML 是 Web Services 的基础

## 它如何工作 ？

基础的 Web Services 平台是 XML + HTTP。

HTTP 协议是最常用的因特网协议。

XML 提供了一种可用于不同的平台和编程语言之间的语言。

### Web services 平台的元素：

- SOAP (简易对象访问协议)
- UDDI (通用描述、发现及整合)
- WSDL (Web services 描述语言)

我们会在本教程后面章节讲解这些主题。

[← Web Services 教程](#)[为什么使用 Web Services? →](#)

 点我分享笔记

## 为什么使用 Web Services?

几年前，Web services 的速度还没有快到让人们产生兴趣的程度。

### 最重要的事情是协同工作

由于所有主要的平台均可通过 Web 浏览器来访问 Web，不同的平台可以借此进行交互。为了让这些平台协同工作，Web 应用程序被开发了出来。

Web 应用程序是运行在 Web 上的简易应用程序。它们围绕 Web 浏览器标准被进行构建，几乎可被任何平台之上的任何浏览器来使用。

### Web services 把 Web 应用程序提升到了另外一个层面

通过使用 Web services，您的应用程序可向全世界发布功能或消息。

Web services 使用 XML 来编解码数据，并使用 SOAP 借由开放的协议来传输数据。

通过 Web services，您的会计部门的 Win 2k 服务器可与 IT 供应商的 UNIX 服务器进行连接。

### Web services 有两种类型的应用

#### 可重复使用的应用程序组件

有一些功能是不同的应用程序常常会用到的。那么为什么要周而复始地开发它们呢？

Web services 可以把应用程序组件作为服务来提供，比如汇率转换、天气预报或者甚至是语言翻译等等。

比较理想的情况是，每种应用程序组件只有一个最优秀的版本，这样任何人都可以在其应用程序中使用它。

#### 连接现有的软件

通过为不同的应用程序提供一种链接其数据的途径，Web services 有助于解决协同工作的问题。

通过使用 Web services，您可以在不同的应用程序与平台之间来交换数据。

# Web Services 平台元素

Web Services 拥有三种基本的元素:SOAP、WSDL 以及 UDDI。

## 什么是 SOAP ?

基本的 Web services 平台是 XML + HTTP。

- SOAP 指简易对象访问协议
- SOAP 是一种通信协议
- SOAP 用于应用程序之间的通信
- SOAP 是一种用于发送消息的格式
- SOAP 被设计用来通过因特网进行通信
- SOAP 独立于平台
- SOAP 独立于语言
- SOAP 基于 XML
- SOAP 很简单并可扩展
- SOAP 允许您绕过防火墙
- SOAP 将作为 W3C 标准来发展

如需更多有关 SOAP 的知识，请访问我们的 [《SOAP 教程》](#)

## 什么是 WSDL?

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

- 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

如需更多有关 WSDL 的知识，请访问我们的 [《WSDL 教程》](#)

## 什么是UDDI ?

UDDI 是一种目录服务，通过它，企业可注册并搜索 Web services。

- UDDI 指通用的描述、发现以及整合 ( Universal Description, Discovery and Integration ) 。
- UDDI 是一种用于存储有关 web services 的信息的目录。
- UDDI 是一种由 WSDL 描述的网络服务接口目录。
- UDDI 经由 SOAP 进行通讯。
- UDDI 被构建于 Microsoft .NET 平台之中。

[← 为什么使用 Web Services?](#)[Web Service 实例 →](#)[📝 点我分享笔记](#)

# Web Service 实例

任何应用程序都可拥有 Web Service 组件。  
Web Service 的创建与编程语言的种类无关。  
本章节我们将为大家介绍使用 PHP 的 SOAP 扩展来创建 Web Service。  
SOAP有两种操作方式，NO-WSDL 与 WSDL。

- **NO-WSDL模式**：使用参数来传递要使用的信息。
- **WSDL模式**：使用WSDL文件名作为参数，并从WSDL中提取服务所需的信息。

## 一个实例：PHP Web Service

在开始实例前，我们需要确定PHP是否安装了 SOAP 扩展。查看 phpinfo，出现以下信息表明已经安装了 SOAP 扩展：

soap		
Soap Client	enabled	
Soap Server	enabled	
Directive	Local Value	Master Value
soap.wsdl_cache	1	1
soap.wsdl_cache_dir	/tmp	/tmp
soap.wsdl_cache_enabled	1	1
soap.wsdl_cache_limit	5	5
soap.wsdl_cache_ttl	86400	86400

在这个例子中，我们会使用 PHP SOAP 来创建一个简单的 Web Service。

服务端 **Server.php** 文件代码如下：

```
<?php
// SiteInfo 类用于处理请求
Class SiteInfo
{
    /**
     * 返回网站名称
     * @return string
     */
    public function getName(){
        return "菜鸟教程";
    }

    public function getUrl(){
        return "www.runoob.com";
    }
}
```

```
}

// 创建 SoapServer 对象
$s = new SoapServer(null,array("location"=>"http://localhost/soap/Server.php","uri"=>"Server.php"));

// 导出 SiteInfo 类中的全部函数
$s->setClass("SiteInfo");
// 处理一个SOAP请求，调用必要的功能，并发送回一个响应。
$s->handle();
?>
```

客户端 **Client.php** 文件代码如下：

```
<?php
try{
    // non-wsdl方式调用web service
    // 创建 SoapClient 对象
    $soap = new SoapClient(null,array('location'=>"http://localhost/soap/Server.php",'uri'=>'Server.php'
));
    // 调用函数
    $result1 = $soap->getName();
    $result2 = $soap->__soapCall("getUrl",array());
    echo $result1."<br/>";
    echo $result2;
} catch(SoapFault $e){
    echo $e->getMessage();
} catch(Exception $e){
    echo $e->getMessage();
}
```

这时我们访问 <http://localhost/soap/Client.php>，输出结果如下所示：



localhost/soap/Client.php

菜鸟教程  
www.runoob.com



[← Web Services 平台元素](#)

Web Services 总结 [→](#)

[点我分享笔记](#)

[← Web Service 实例](#)

## 您已经学习了 Web Services, 下一步学习什么内容呢？

### Web Services 概要

本教程已经向您讲解了如何把应用程序转换为网络应用程序。

您已经学习了如何使用 XML 在应用程序间发送消息。

您也学习了如何从应用程序导出某项功能（创建一个 web service ）。

### 您已经学习了 Web Services, 下一步呢？

下一步您应当学习 WSDL 和 SOAP。

#### WSDL

WSDL 是基于 XML 的用来描述 Web services 以及如何访问它们的一种语言。

WSDL 可描述 web service ，连同用于 web service 的消息格式和协议的细节。

如果您希望学习更多有关 WSDL 的知识，请访问我们的《[WSDL 教程](#)》。

#### SOAP

SOAP 是一种使应用程序有能力通过 HTTP 交换信息的基于 XML 的简易协议。

或者可以更简单地说：SOAP 是一种用于访问 web service 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](#)》。

[← Web Service 实例](#)[✎ 点我分享笔记](#)