

## *Web Services*

Web services are (at least according to me) the main point of .NET. They are the tool for enabling a program on computer A to call programs on computer B just as if they were running on computer A. The vision is that the entire Internet is (or will eventually be) just one big computer.

Some of the difficulties involved are similar to the difficulties that ActiveX was invented to solve: how can programs written in different languages call each other? We learned that IDL (interface definition language) was the solution to that problem. Similarly, there is WSDL (Web services definition language). This language is described in a document submitted to the World Wide Web Consortium by (employees of) Ariba, IBM, and Microsoft. The full document is available at

[http://www.w3.org/TR/wsdl#\\_types](http://www.w3.org/TR/wsdl#_types)

You might want to go there and read the introduction. There is no need to read the whole document for CS130. Here is an excerpt:

As communications protocols and message formats are standardized in the web community, it becomes increasingly possible and important to be able to describe the communications in some structured way. WSDL addresses this need by defining an XML grammar for describing network services as collections of communication endpoints capable of exchanging messages. WSDL service definitions provide documentation for distributed systems and serve as a recipe for automating the details involved in applications communication.

A WSDL document defines **services** as collections of network endpoints, or **ports**. In WSDL, the abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings. This allows the reuse of abstract definitions: **messages**, which are abstract descriptions of the data being exchanged, and **port types** which are abstract collections of **operations**. The concrete protocol and data format specifications for a particular port type constitutes a reusable **binding**. A port is defined by associating a network address with a reusable binding, and a collection of ports define a service.

Don't be surprised if that is difficult to understand. It's not your fault—it's just a sketch. To understand WSDL you would have to study it hard. From

this introduction you're just supposed to get a vague idea that WSDL defines a language in which web services can be described. Here's the

***Bottom Line:* Every web service is defined in a .wsdl file.**

For example, the Google Search web service that you will use in your programming assignment is defined in the file GoogleSearch.wsdl. That file is found at

<http://api.google.com/GoogleSearch.wsdl>

That file is quite readable. Near the bottom it defines the services offered by the GoogleSearch service, in terms of messages. The functions you can call are defined by message names beginning with *do*. Their arguments are indicated as “parts” of the message.

*Question:* IDL (Interface Definition Language) is used to specify interfaces for COM components. Why do we need WSDL when we already have IDL? Isn't IDL enough to describe a web service by telling its properties, methods, and the events it can fire? *Answer:* While properties, methods, and events can be considered as *messages* in the sense of WSDL, IDL only defines some fixed types; WSDL doesn't contain a type definition language at all, but allows you to use IDL for that purpose if you like or some other type definition language. WSDL also offers XML-based ways to define data formats.

*Question:* What do I do with a WSDL file? *Answer:* You want to use a tool that reads the WSDL file for you and creates a corresponding class in your program, with methods corresponding to the interface defined in the WSDL file. Each of these methods, when called, will create a network connection to the web service, prepare (“marshall”) the arguments to the method in the XML format specified in the WSDL file, send the “message” to the web service, receive the answer in XML form, and return it to the method's caller in your program. In Visual Studio, this class and its methods are created for you when you choose *Add Web Reference* and point to a WSDL file.

## *An Example Program*

In the lab, you will create a program using a Web service. Here you will study the conceptual background needed for that lab. To do that, you will start creating the project that you will finish in the lab, but we will only go a couple of steps into it here. The Web service we will use is the Google search service. This web service provides an API (Application Programming Interface) to the Google search engine.

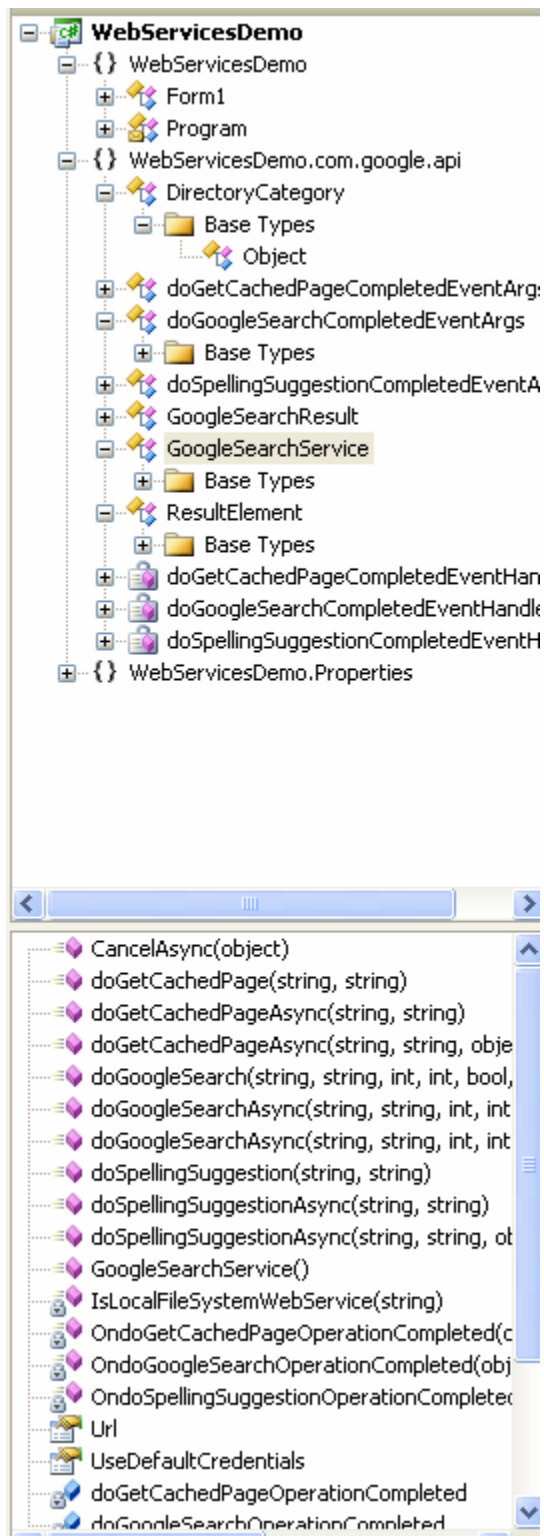
1. Create a new project called *WebServicesDemo*, Make it a C# “Windows Application”.

2. In Solution Explorer, right-click *WebServicesDemo* and choose *Add Web Reference*.

Alternately, you can use the menus: *Project / Add Web Reference*.

3. Navigate to <http://api.google.com/GoogleSearch.wsdl> . When you see the Google Search Service in the *Services Available at this URL* window, click *Add Reference*. The WSDL file provides a formal description of the API of this web service.

4. Go to Class View and look under *WebServicesDemo / com*. Observe that you have several classes, including *GoogleSearchService*, and in that class, many methods, all available to use in your code. Observe that all these methods are in the namespace *WebServicesDemo.com.google.api*. Here is a screen shot to show what you will see:



When you have gotten this far, you are prepared for the lab. The point is to understand that the many functions written by Google programmers, and used in programming the Google search engine, are *available to you for use*

*in your own programs!* Look, there they are in your Visual Studio window, ready to be called.