Белгородский Государственный Технологический Университет им. В. Г. Шухова

Кафедра программного обеспечения вычислительной техники  
и автоматизированных систем

## Лабораторная работа №7 по теме: «Разработка веб-служб»

**Выполнил:**  
студент группы ПВ-41  
Адаменко И. И.

**Проверил:**  
старший преподаватель  
Гарибов А. И.

Белгород  
2015

**Цель работы:** научиться работать с удалённым вызовом процедур и такими технологиями как XML-RPC, SOAP и пр.

# Задание

Разработать службу на PHP или ASP.NET, которая имеет 2-3 веб-метода с параметрами. Также разработать приложения на PHP и ASP.NET для демонстрации универсальности этой службы.

# Теоретическая часть

## XML-RPC

XML-RPC — стандарт/протокол вызова удалённых процедур, использующий XML для кодирования своих сообщений и HTTP в качестве транспортного механизма. Является прародителем SOAP, отличается исключительной простотой в применении. XML-RPC, как и любой другой интерфейс Remote Procedure Call (RPC), определяет набор стандартных типов данных и команд, которые программист может использовать для доступа к функциональности другой программы, находящейся на другом компьютере в сети.

Протокол XML-RPC был изначально разработан Дэйвом Винером из компании «UserLand Software» в сотрудничестве с Майкрософт, в 1998 году. Однако корпорация Майкрософт вскоре сочла этот протокол слишком упрощённым, и начала расширять его функциональность. После нескольких циклов по расширению функциональности, появилась система, ныне известная как SOAP. Позднее Майкрософт начала широко рекламировать и внедрять SOAP, а изначальный XML-RPC был отвергнут. Но, несмотря на отвержение его Майкрософтом, стандарт XML-RPC очаровал многих программистов своей необычайной простотой и, за счёт этого, существует по сей день и даже постепенно набирает популярность.

## SOAP

SOAP — протокол обмена структурированными сообщениями в распределённой вычислительной среде. Первоначально SOAP предназначался в основном для реализации удалённого вызова процедур (RPC). Сейчас протокол используется для обмена произвольными сообщениями в формате XML, а не только для вызова процедур. SOAP является расширением протокола XML-RPC.

SOAP может использоваться с любым протоколом прикладного уровня: SMTP, FTP, HTTP, HTTPS и др. Однако его взаимодействие с каждым из этих протоколов имеет свои особенности, которые должны быть определены отдельно. Чаще всего SOAP используется поверх HTTP.

## WSDL

WSDL — язык описания веб-сервисов и доступа к ним, основанный на языке XML.

Каждый документ можно разбить на следующие логические части:

1. Определение типов данных (types) — определение вида отправляемых и получаемых сервисом XML-сообщений.
2. Элементы данных (message) — сообщения, используемые веб-сервисом.
3. Абстрактные операции (portType) — список операций, которые могут быть выполнены с сообщениями.
4. Связывание сервисов (binding) — способ, которым сообщение будет доставлено.

WSDL широко используется в связке с технологиями XML-RPC и SOAP для описания методов, запросов и в целом сервисов и их взаимодействия.

## WCF

Windows Communication Foundation — программный фреймворк, используемый для обмена данными между приложениями, входящий в состав .NET Framework.

WCF делает возможным построение безопасных и надёжных транзакционных систем через упрощённую унифицированную программную модель межплатформенного взаимодействия. Комбинируя функциональность существующих технологий .NET по разработке распределённых приложений, WCF предоставляет единую инфраструктуру разработки, при умелом применении повышающую производительность и снижающую затраты на создание безопасных, надёжных и транзакционных веб-служб нового поколения. Заложенные в неё принципы интероперабельности позволяют организовать работу с другими платформами, для чего используются технологии взаимодействия платформ, например WSIT, разрабатываемые на базе открытого исходного кода.

# Практическая часть

В этой лабораторной работе я разработал веб-службу с помощью технологии XML-RPC на языке PHP, которая повторяет функционал третьей лабораторной работы: вычисление корней квадратного уравнения по коэффициентам и площади треугольника по длинам сторон.

Тестовые приложения написаны на PHP с помощью XML-RPC, а также на ASP.NET.

Для реализации клиента и сервера с XML-RPC в PHP я использовал открытый пакет библиотек PHPXMLRPC, поскольку его использование не предполагает изменения настроек сервера и конфигурирования интерпретатора, а значит программа будет работать на любом сервере с поддержкой PHP.

Для реализации клиента с XML-RPC в ASP.NET я использовал библиотеку XML-RPC.NET, которая, так же, позволяет использовать службы XML-RPC даже в среде с .NET версии 1.0. Тем самым программа практически не зависит от используемой версии .NET.

## Сервер

Как уже было сказано выше, сервер, т. е., сама веб-служба, разработана на PHP. Если просто перейти на её адрес в интернете, то откроется документация по всем существующим методам (см. скриншоты). Сами же методы, которые предоставляет служба, используют для вычисления те же функции, которые были реализованы в третьей лабораторной работе.

Исходный код веб-службы:

1. <?
2. **include**("lib/xmlrpc.inc");
3. **include**("lib/xmlrpcs.inc");
4. **include**("lib/xmlrpc\_wrappers.inc");
5. **include**("lib/docxmlrpcs.inc");
7. **function** getContent($url) {
8. $ch = curl\_init();
9. curl\_setopt($ch, CURLOPT\_HEADER, 0);
10. curl\_setopt($ch, CURLOPT\_VERBOSE, 0);
11. curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);
12. curl\_setopt($ch, CURLOPT\_URL, $url);
14. $result = curl\_exec($ch);
16. curl\_close($ch);
18. **return** $result;
19. }
21. // methods
22. $calcTriangle\_Sig = **array**(**array**(
23. $xmlrpcString,
24. $xmlrpcDouble,
25. $xmlrpcDouble,
26. $xmlrpcDouble
27. ));
28. $calcTriangle\_Doc = 'Returns area of triangle by length of sides';
29. **function** calcTriangle($params) {
30. $data = **array**(
31. 'a' => $params->getParam(0)->scalarval(),
32. 'b' => $params->getParam(1)->scalarval(),
33. 'c' => $params->getParam(2)->scalarval()
34. );
36. $url = 'http://igoradamenko.com/bstu/pv/solvers/square.php?' . http\_build\_query($data);
38. **return** **new** xmlrpcresp(**new** xmlrpcval(getContent($url)));
39. }
41. $calcEquals\_Sig = **array**(**array**(
42. $xmlrpcString,
43. $xmlrpcDouble,
44. $xmlrpcDouble,
45. $xmlrpcDouble
46. ));
47. $calcEquals\_Doc = 'Returns the roots of quadratic equation';
48. **function** calcEquals($params) {
49. $data = **array**(
50. 'a' => $params->getParam(0)->scalarval(),
51. 'b' => $params->getParam(1)->scalarval(),
52. 'c' => $params->getParam(2)->scalarval()
53. );

56. $url = 'http://igoradamenko.com/bstu/pv/solvers/equals.php?' . http\_build\_query($data);
58. **return** **new** xmlrpcresp(**new** xmlrpcval(getContent($url)));
59. }
61. // all to one obj
62. $methods = **array**(
63. 'lab.calcTriangle' => **array**(
64. 'function' => 'calcTriangle',
65. 'signature' => $calcTriangle\_Sig,
66. 'docstring' => $calcTriangle\_Doc
67. ),
68. 'lab.calcEquals' => **array**(
69. 'function' => 'calcEquals',
70. 'signature' => $calcEquals\_Sig,
71. 'docstring' => $calcEquals\_Doc
72. )
73. );
75. **if** ($\_SERVER['REQUEST\_METHOD'] != 'POST' || $\_SERVER['CONTENT\_TYPE'] == 'application/x-www-form-urlencoded') {
76. $server = **new** documenting\_xmlrpc\_server($methods, false);
77. } **else** {
78. $server = **new** xmlrpc\_server($methods, false);
79. }
81. $server->compress\_response = true;
83. $server->service();
84. ?>

## Демонстрационные приложения

Все демо-приложения состоят из формы и кода за ней. Форма отправляет POST-запрос на страницу, на которой она находится. Код проверяет, если запрос пришёл, то отправляет данные из него в XML-RPC метод, получает ответ и выводит его вместо формы.

### PHP

Исходный код одного из демонстрационных приложений (без разметки страницы) выглядит следующим образом:

1. <?
2. **if** (isset($\_POST['submit'])) {
3. **include**("lib/xmlrpc.inc");
5. $msg = **new** xmlrpcmsg('lab.calcEquals', **array**(
6. **new** xmlrpcval($\_POST['a'], "double"),
7. **new** xmlrpcval($\_POST['b'], "double"),
8. **new** xmlrpcval($\_POST['c'], "double")
9. ));
11. $client = **new** xmlrpc\_client("http://igoradamenko.com/bstu/pv/xmlrpc/server.php");
13. $response = &$client->send($msg);
15. **if** (!$response->faultCode()) {
16. $result = $response->value()->scalarval();
17. } **else** {
18. $result = "Ошибка:\n"
19. . "<pre>Код: " . htmlspecialchars($response->faultCode())
20. . " Причина: '" . htmlspecialchars($response->faultString())  
     . "'</pre><br/>";
21. }
22. }
23. ?>

### ASP.NET

Для работы XML-RPC-клиента реализован интерфейс, который описывает используемые методы.

Исходный код одного из демонстрационных приложений (без разметки страницы) выглядит следующим образом:

1. // interface in Global.asax.cs
2. [XmlRpcUrl("http://igoradamenko.com/bstu/pv/xmlrpc/server.php")]
3. **public** **interface** ILab : IXmlRpcProxy
4. {
5. [XmlRpcMethod("lab.calcEquals")]
6. **string** calcEquals(**double** a, **double** b, **double** c);
8. [XmlRpcMethod("lab.calcTriangle")]
9. **string** calcTriangle(**double** a, **double** b, **double** c);
10. }
12. // methods in the code behind one of the demo pages
13. **protected** **string** response = **null**;
14. **protected** **void** Page\_Load(**object** sender, EventArgs e)
15. {
16. var \_post = Request.Form;
18. **if** (\_post[«submit»] != **null**)
19. {
20. ILab proxy = XmlRpcProxyGen.Create<Ilab>();
21. proxy.NonStandard = XmlRpcNonStandard.All;
22. proxy.UseStringTag = **true**;
23. proxy.UseEmptyParamsTag = **true**;
24. response = proxy.calcEquals(Convert.ToDouble(\_post[“a”]),
25. Convert.ToDouble(\_post[«b»]),
26. Convert.ToDouble(\_post[«c»]));
27. }
28. }

# Скриншоты

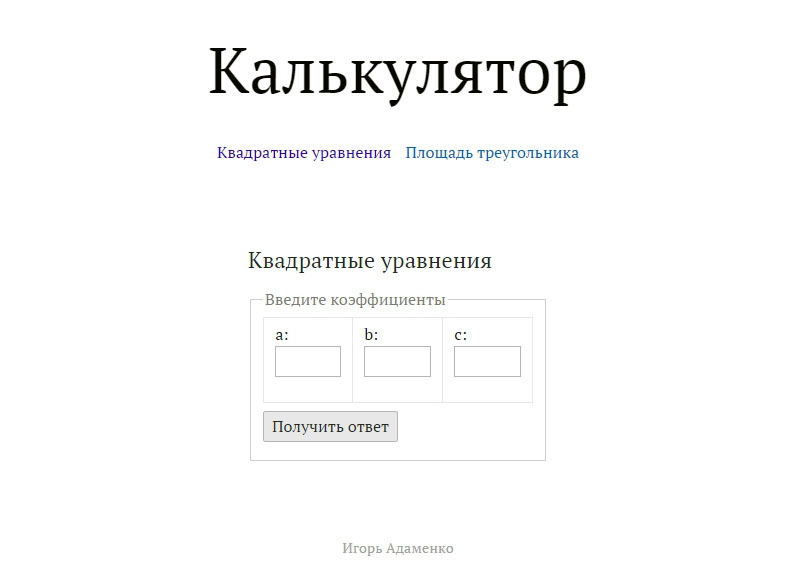


Рисунок 1. Демонстрационное веб-приложение

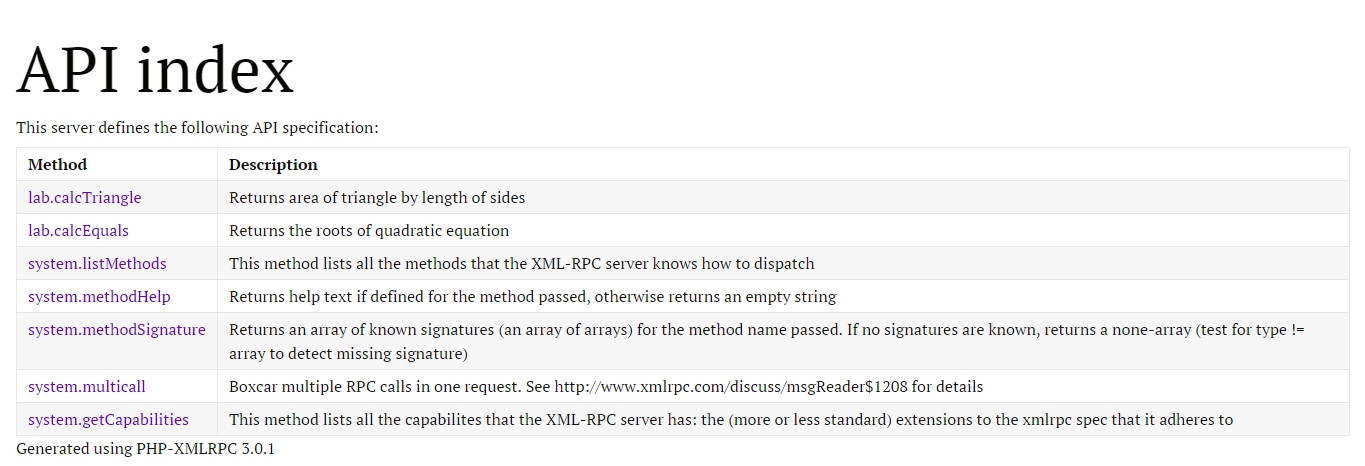


Рисунок 2. Автоматически сгенерированное описание методов веб-службы