Criando um Web Service PHP com NuSoap e acessando-o com NCLua Soap

by Johnny Moreira Gomes

Neste tutorial, apresentarei passo-a-passo o processo de criação de um Web Service PHP utilizando a biblioteca NuSoap (você pode baixá-la através da seção de links ao final deste arquivo) de modo que o mesmo possa ser consumido por uma aplicação de tv digital em Ncl/Lua.

Primeiramente criaremos um Web Service de exemplo, para que posteriormente montemos o seu cliente em NCL/Lua, utilizando o módulo NCLua SOAP, criado por Manoel Campos. Este último pode ser baixado em nossa seção de links ou através da página do Manoel Campos referente a este módulo, pelo link http://ncluasoap.manoelcampos.com .

Parte I – Criação do Web Service

Vamos criar um Web Service em PHP baseado em SOAP contendo uma simples função que retorna a classificação de um triângulo de acordo com os tamanhos de seus lados, passados como parâmetros. Para isso, utilizaremos a biblioteca PHP NuSoap, também encontrada na seção de links deste tutorial. Esta biblioteca é uma alternativa ao módulo *SOAP Extension*, nativo do PHP, que foi escolhida para esta situação pelo fato de oferecer suporte à criação automática do WSDL.

Para que o módulo NuSoap funcione em sua versão atual, é necessária a utilização do PHP versão dentre 5.0 até 5.25. E para o teste de nosso cliente NCLua em uma máquina local, é necessário configurar o Apache para o modo online. Para facilitar o nosso trabalho por aqui, foi utilizado o WampServer 1.7.4 (também disponível para download ao final deste documento) para Windows, que é um software que reúne o Apache, o PHP 5.25 e o MySql 5.0.45, automaticamente configurados após a instalação para o uso em máquina local.

O nosso Web Service estará contido em uma pasta de nome “mywebservice” no diretório raiz de nosso servidor Apache. Dentro desta pasta estará a pasta “lib”, referente ao módulo NuSoap, e o arquivo “server.php”, que consiste no arquivo principal de nosso Web Service. O arquivo é iniciado da seguinte maneira:

```
1  <?php
2      require_once("lib/nusoap.php");
3
4      $server = new soap_server;
5
6      $server->configureWSDL('server.triangle', 'urn:server.triangle');
7      $server->wsdl->schemaTargetNamespace = 'urn:server.triangle';
```

Nesse início, nós simplesmente incluímos o arquivo principal do módulo NuSoap, instanciamos a variável $server como um objeto soap_server e inicializamos o suporte ao WSDL, batizando o nosso serviço com o namespace “server.triangle” nas linhas 6 e 7. A diretiva “urn:” indica que estamos utilizando um identificador de recurso ao invés de um caminho, como ocorre em URL's ou URI's.

Na listagem abaixo, registramos a função de nosso Web Service utilizando o método “register”,

cujos parâmetros são explicados nos comentários.

```
    $server->register(  'triangle',                         //Nome da Função
                        array('a' => 'xsd:string',         //Parâmetros da Função
                              'b' => 'xsd:string',
                              'c' => 'xsd:string'),
                        array('return' => 'xsd:string'),    //Valores de Retorno
                        'urn:server.triangle',              //Namespace
                        'urn:server.triangle#triangle',     //SOAPAction
                        'rpc',                              //Style
                        'encoded',                          //Use
                        'Classifica um triângulo com base nas medidas dos
lados.' //Descrição do Serviço
                      );
```

Agora, escreveremos a função de nosso Web Service, ou seja, implementaremos o método registrado acima, recebendo os parâmetros indicados e retornando o valor esperado.

```
    function triangle($a, $b, $c){

        if ($a<=0 || $b<=0 || $c <=0)
            return 'Not a triangle.';
        if (($a>=$b + $c)||($b>=$a + $c)||($c>=$a + $b))
            return 'Not a triangle.';
        if (($a<=abs($b - $c))||($b<=abs($a - $c))||($c<=abs($a - $b)))
            return 'Not a triangle.';

        //É um triângulo
        //Colocando os lados em ordem crescente
        $lados = array($a, $b, $c);
        sort($lados, SORT_NUMERIC);
        $a = $lados[2]; $b = $lados[1]; $c = $lados[0];

        //Variável de resposta
        $ans = '';

        //É não equilátero?
        if ($a>$c) {
            //Não é assim que se verifica a igualdade de 2 números reais em
            //um computador, mas vai assim mesmo...
            if ($a==$b || $b==$c)
                $ans = 'Isosceles, ';
            else
                $ans = 'Scalene, ';


            if ($a*$a > $b*$b + $c*$c)
                $ans .= 'obtuse triangle.';
            else if ($a*$a < $b*$b + $c*$c)
                $ans .= 'acute triangle.';
            else
                $ans .= 'rectangle triangle.';

        } else {
            $ans = 'Equilateral triangle.';
        }

        return $ans;

    }
```

Finalmente, incluiremos as linhas referentes ao processamento da requisição enviada pelo cliente

ao nosso Web Service, conforme mostrado na listagem abaixo:

```
64      $HTTP_RAW_POST_DATA = isset($HTTP_RAW_POST_DATA) ? $HTTP_RAW_POST_DATA : ''
    ;
65      $server->service($HTTP_RAW_POST_DATA);
66
67  ?>
```

Antes de iniciarmos nossos trabalhos com o NCL/Lua, podemos testar o funcionamento do nosso Web Service através de um cliente PHP. Para tanto, basta executar o script PHP listado abaixo, lembrando que "localhost" se refere à máquina local e pode ser substituído por um ip ou um domínio, conforme o ambiente onde está hospedado o nosso serviço.

```
 1  <?php
 2      require_once("lib/nusoap.php");
 3
 4      $wsdl = 'http://localhost/mywebservice/server.php?wsdl';
 5      $client = new soapclient($wsdl, true);
 6
 7      $erro = $client->getError();
 8      if ($erro) {
 9          echo 'Erro no construtor <pre>'. $erro .'</pre>';
10      }
11
12      $result = $client->call('triangle', array('3', '4', '5'));
13
14      if ($result->fault)
15          echo 'Falha: <pre>'. $result->fault .'</pre>';
16      else {
17          $erro = $client->getError();
18          if ($erro) {
19              echo 'Erro: <pre>'. $erro .'</pre>';
20          } else {
21              echo $result;
22          }
23      }
24  ?>
```

Caso haja interesse em acessar o WSDL do nosso Web Service, basta acessar a url do arquivo principal em adição do parâmetro "?wsdl". Por exemplo, se a aplicação estiver rodando na máquina local, conseguiremos visualizar o WSDL no navegador através do endereço "http://localhost/mywebservice/server.php?wsdl".

Observações e Cuidados a serem tomados:

- O arquivo principal do Web Service e os demais arquivos incluídos no mesmo não devem conter absolutamente nada no exterior dos delimitadores de código PHP, incluindo espaços e quebras de linha. Isso porque há métodos do módulo NuSoap que enviam *HTTP Headers*, exigindo que nenhuma informação prévia tenha sido enviada via HTTP.

- O código de implementação do método de retorno não pode conter comandos "echo" ou "print", pois toda a informação de saída do Web Service será transformada em conteúdo XML. Isso implica que a presença desses comandos pode vir a comprometer a codificação do arquivo a ser gerado.

Parte II – Criação do cliente NCLua

Como foi dito anteriormente, utilizaremos o módulo NCLua Soap para a comunicação de nossa aplicação de tv digital com o nosso recém-criado Web Service PHP. O módulo em questão utiliza as bibliotecas Lua TCP, LuaXML, HTTP e base64, além de alguns métodos de propósito geral presentes no arquivo "util.lua".

O código fonte da aplicação que criaremos aqui está presente na nossa seção de links e contém todos os arquivos conforme a especificação acima. Caso haja interesse em entender com mais profundidade as extensões utilizadas, sugiro que acesse a página do Manoel Campos para a obtenção de maiores informações (link presente no início deste tutorial ou na seção de fontes).

A nossa aplicação NCLua conterá, além dos arquivos essenciais para o funcionamento do NCLua SOAP, um arquivo NCL com apresentação em branco. Optei por essa estratégia para que o nosso código lua fique mais "limpo", contendo apenas comandos de saída "print", ou seja, gerando saída apenas para o cliente de SSH que se comunicará com a máquina virtual Ginga-NCL. Além disso, haverá o arquivo "main.lua", que será executado assim que a aplicação NCL se iniciar.

Começaremos o nosso "main.lua" com a inicialização das variáveis contendo os parâmetros a serem passados para o Web Service e o ip de onde o mesmo está hospedado. Caso o mesmo esteja em sua máquina local, o servidor Apache deve estar configurado para o modo online e a diretiva "localhost" não irá servir, pois agora o escopo da aplicação é uma máquina virtual. Portando, basta informar o ip atual de sua conexão com a internet. Abaixo está uma imagem mostrando como ativar o modo online do Apache através do WampServer.

Listagem contendo as primeiras linhas de “main.lua”:

```
1  --Inclusão do módulo criado por Manoel Campos para consumir
2  --web services baseados em soap
3  require "ncluasoap"
4
5  local sa = "3"
6  local sb = "4"
7  local sc = "5"
8  local ip = "192.168.0.100"
```

Agora, criaremos uma Table Lua contendo as informações necessárias para a criação da requisição ao Web Service. Tais informações são: Endereço do arquivo principal do Web Service; o namespace do serviço, anteriormente escolhido (sem o “urn:”); o nome do método e os parâmetros do mesmo.

```
10 local msgTable = {
11   --Endereco do Web Service
12   address = "http://".. ip .."/mywebservice/server.php",
13   --Namespace, encontrado no arquivo WSDL referente ao web service
14   namespace = "server.triangle",
15   --Nome do metodo a ser acessado
16   operationName = "triangle",
17   --Abaixo, os parametros do metodo a ser acessado
18   params = {
19     a = sa,
20     b = sb,
21     c = sc
22   }
23 }
```

Julgo importante ressaltar um fato interessante com relação aos parâmetros em nossa Table Lua. Um Web Service PHP baseado em NuSoap **não considera o nome dos parâmetros, somente a ordem em que eles aparecem no xml de requisição**. No nosso exemplo, com a Table Lua considerada, o módulo NCLua soap irá gerar um xml de requisição contendo o seguinte fragmento, referente aos parâmetros a serem passados para a função do serviço:

```
18 ...
19 <a>3</a>
20 <b>4</b>
21 <c>5</c>
22 ...
```

Apesar de nosso Web Service ter sido registrado com os parâmetros de nomes “a”, “b” e “c”, o mesmo comportamento será evidenciado se o fragmento acima for substituído por:

```
18 ...
19 <ladoa>3</ladoa>
20 <ladob>4</ladob>
21 <ladoc>5</ladoc>
22 ...
```

Isso ocorre porque, conforme dito antes, o NuSoap simplesmente joga o valor da primeira tag de parâmetros no primeiro argumento do método de serviço, e assim por diante. O módulo NCLua SOAP, por sua vez, ao gerar o xml de requisição, escreve as tags de parâmetros em ordem arbitrária. Poderia acontecer, por exemplo, de o fragmento xml mostrado acima ter sido gerado da seguinte forma:

```
18  ...
19  <a>4</a>
20  <b>5</b>
21  <c>3</c>
22  ...
```

Quando o NuSoap fosse executar o trabalho de interpretação da requisição, a nossa função seria chamada da seguinte maneira:

triangle(“4”, “5”, “3”);

Ou seja, se no nosso caso a ordem em que os lados do triângulo aparecessem na função importasse , ocorreria um resultado inesperado (para o nosso Web Service a ordem não importará, pois dentro do método *triangle* os lados são postos em ordem crescente de medida). Portando, para evitar maiores problemas, aconselho sempre a se conhecer a ordem em que aparecem os argumentos da função de serviço e, ao se gerar a Table Lua de requisição, especificá-los respectivamente da seguinte forma:

```
params = {
    {a = "3"},
    {b = "4"},
    {c = "5"}
}
```

Como cada elemento da tabela *params* não possui identificador explícito, ou seja, os seus elementos são sub-tabelas sem nome, o mesmo passa a ter o formato de um *array*. Deste modo, seus elementos são associados biunivocamente com índices numéricos, garantindo assim que a ordem de declaração dos parâmetros seja respeitada no momento de montagem do xml de requisição por parte do módulo NCLua SOAP.

Voltando ao nosso “main.lua”, após especificarmos o Table Lua de requisição, implementaremos a função que irá receber e tratar a resposta obtida do Web Service. Esta função obrigatoriamente deve receber um parâmetro referente à resposta a ser obtida do Web Service.

```
27  -- A funcao de retorno enviada para o metodo call deve ter um parametro
28  -- referente à resposta obtida do web service.
29  -- Se o resultado for composto, result será uma table lua, caso contrário
30  -- será uma string.
31  local function resposta(result)
32      print("\n\n********* Saida da Aplicacao **********\n")
33      print("Lados: " .. sa .., "..sb..", " .. sc)
34      --Mostra o conteúdo recebido
35      print("---------------------- "..result)
36      print("\n********** Saida da Aplicacao **********\n\n")
37  end
```

Finalmente, chamamos o método “call” do módulo NCLua SOAP, passando como argumentos o Table Lua de requisição, o nome da função de retorno que acabamos de criar e a versão do SOAP utilizada pelo Web Service, que no caso do NuSoap será 1.1.

```
38  --Método que envia a requisição ao web service e obtém o retorno
39  --              Tabela Req.    Funcao retorno    Versão do Soap
40  ncluasoap.call(msgTable,      resposta,         '1.1')
```

Conclusão

Por aqui, terminamos mais um tutorial sobre aplicações em Tv Digital, desta vez buscando mostrar um pouco sobre o funcionamento da comunicação entre um Web Service PHP e uma aplicação NCL/Lua. Espero que, de modo geral, o que foi exposto aqui tenha sido proveitoso, uma vez que Web Services se mostram bastante adequados para fornecer informações às mais variadas plataformas, podendo constituir uma ferramenta de interatividade poderosíssima nas mais diversas aplicações de Tv Digital. E esse recurso, quando somado às vantagens oferecidas pela tecnologia PHP, torna-se um instrumento gratuito e de fácil manuseio, que atende desde as aplicações mais simples até as mais engenhosas.

Links

- Módulo NuSoap PHP:
  http://sourceforge.net/projects/nusoap/
- WampServer 1.7.4:
  http://www.soft-go.com/view/WAMP5_20010.html
- Código Fonte Web Service PHP (Inclui a NuSoap 0.9.5)
  http://www.ufjf.br/lapic/files/2010/05/mywebservice.zip
- Código Fonte Cliente NCLua SOAP (Inclui módulo NCLua SOAP 0.5.6 e dependências)
  http://www.ufjf.br/lapic/files/2010/06/client-triangle.zip

Fontes

http://www.scottnichol.com/nusoapintro.htm
http://www.scottnichol.com/nusoapprogwsdl.htm
http://ncluasoap.manoelcampos.com

Outros Projetos Interessantes em TV Digital (por Manoel Campos)

http://enquetetvd.manoelcampos.com
http://ncluatweet.manoelcampos.com
http://ncluarss.manoelcampos.com

Johnny Moreira Gomes

Aluno do curso de Ciências Exatas da Universidade Federal de Juiz de Fora (UFJF) e Bolsista do Grupo de Tv Digital do Laboratório de Aplicações e Inovação em Computação (LApIC) da UFJF