Curso POO PHP Servizos web: extensión SOAP de PHP

Sumario

Servizos web: extensión SOAP de PHP

Aínda que existen outras posibilidades para crear e utilizar servizos web, **a extensión SOAP** é a implementación de SOAP que se inclúe con PHP a partir da versión 5 da linguaxe. En versións anteriores tíñase que botar mán doutras opcións para traballar con SOAP. É unha extensión nativa (escrita en linguaxe C) e polo tanto máis rápida que as outras posibilidades.

É compatible coas versións SOAP 1.1 e SOAP 1.2, así como con WSDL 1.1, aínda que non permite a xeración automática do documento WSDL a partir do servizo web programado. Para poder usar a extensión, deberás comprobar se xa se encontra dispoñible (por exemplo, consultando a saída obtida pola función *phpinfo*).

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

As dúas clases principais da extensión son **SoapClient** e **SoapServer**. A primeira permitirache comunicarte cun servizo web, e coa segunda poderás crear os teus propios servizos.

Utilización dun servizo web

Para crear un cliente do servizo, deberás coñecer os detalles deste (como mínimo, os parámetros de entrada e saída que debes usar, e cal é a URL do servizo) e empregar no teu código a clase **SoapClient**. Para descubrir os detalles do servizo, podes consultar o **documento WSDL do servizo**. Con isto obterás información como:

- Os tipos de datos empregados no servizo web.
- Os nomes das funcións ás que é posible realizar chamadas.
- A URL para acceder ao servizo.

A clase SoapClient implementa dous métodos que mostran parte da información que contén o documento WSDL do servizo web; concretamente, os tipos de datos definidos polo servizo, e as funcións que ofrece. Para coñecer esta información, unha vez creado o obxecto, debes utilizar os métodos **__getTypes** e **__getFunctions** respectivamente.

Coa información anterior, para utilizar o servizo dende PHP é necesario crear (se aínda non o fixeches) un **novo obxecto da clase SoapClient**. Se o servizo ten un documento WSDL asociado, no construtor indícaslle onde se encontra.

```
$cliente = new SoapClient("http://localhost/servizo.php?WSDL");
```

E para realizar a chamada ao servizo, inclúes os parámetros nun array.

```
$parametros = array("parametro1" => "valor", "parametro2" => "valor", ...);
$resultado = $cliente->nome_da_funcion($parametros);
```

A chamada devolve un obxecto dunha clase predefinida en PHP chamada **StdClass**.

A extensión PHP5 SOAP tamén inclúe opcións de depuración moi útiles para descubrir que está a pasar cando a conexión ao servizo web non funciona como debería. Para habilitalas, cando fagas a chamada ao construtor da clase SoapClient, debes utilizar a opción **trace** no array de opcións do segundo parámetro.

```
$cliente = new SoapClient(
    "http://localhost/servizo.php?WSDL",
    array('trace'=>true)
);
```

Creación dun servizo web

Para crear un servizo web coa extensión SOAP, debes utilizar a **clase SoapServer**. Vexamos un exemplo sinxelo.

```
function suma($a,$b)  { return $a+$b; }
function resta($a,$b) { return $a-$b; }

$uri="http://localhost/";
$server = new SoapServer(null,array('uri'=>$uri));
$server->addFunction("suma");
$server->addFunction("resta");
$server->handle();
```

O código anterior crea un servizo web con dúas funcións: suma e resta. Cada función recibe dous parámetros e devolve un valor. Para **consumir este servizo**, necesitas escribir o seguinte código.

```
$url="http://localhost/servizo.php";
$uri="http://localhost/";
$cliente = new SoapClient(null, array('location'=>$url,'uri'=>$uri));

$suma  = $cliente->suma(2,3);
$resta = $cliente->resta(2,3);
echo 'A suma é ' . $suma . '<br />';
echo 'A resta é ' . $resta;
```

O servizo que creaches non inclúe un documento WSDL para describir as súas funcións. Sabes que existen os métodos suma e resta, e os parámetros que debes utilizar con eles, porque coñeces o código interno do servizo. Un usuario que non tivese esta información, non sabería como consumir o servizo.

Ao igual que sucedía con SoapClient ao programar un cliente, cando utilizas SoapServer podes crear un servizo sen documento WSDL asociado (como no caso anterior), ou indicar o documento WSDL correspondente ao servizo; pero antes deberás telo creado.

O primeiro parámetro do construtor indica a situación do WSDL correspondente. O segundo parámetro é unha colección de opcións de configuración do servizo. Se existe o primeiro parámetro, xa non fai falta máis información. A clase SoapServer utiliza a información do documento WSDL para executar o servizo. Se, como no exemplo, non existe WSDL, deberás indicar no segundo parámetro polo menos a opción **uri**, co espazo de nomes destino do servizo.

Creación dunha clase para o servizo web

En vez de utilizar funcións para a lóxica interna do servizo web, como a suma e a resta do exemplo anterior, é aconsellable definir unha clase que implementar os métodos que queiramos publicar no servizo.

```
class Calcula {
    public function suma($a, $b) { return $a+$b; }
    public function resta($a, $b){ return $a-$b; }
}
```

Ao facelo desta forma, en lugar de engadir unha a unha as funcións, podemos engadir a clase completa ao servidor utilizando o método **setClass** de SoapServer.

```
require_once('Calcula.php');

$server = new SoapServer(null, array('uri'=>''));
$server->setClass('Calcula');
$server->handle();
```

O arquivo anterior será o punto público de entrada ao noso servizo web.

Xerar o WSDL dun servizo web

Existen algúns mecanismos que nos permiten xerar o WSDL correspondente a un servizo, aínda que sempre é aconsellable revisar os resultados obtidos antes de publicalos. Unha das formas máis sinxelas é utilizar a **libraría WSDLDocument**.

Esta libraría revisa os comentarios que engadises ao código da clase que queiras a publicar (debe ser unha clase, non funcións illadas), e xera como saída o documento WSDL correspondente. Para que funcione correctamente, é necesario que os comentarios das clases sigan un formato específico: o mesmo que utiliza a ferramenta de documentación **PHPDocumentor**.

Os comentarios débense ir introducindo no código distribuídos en bloques, e utilizando certas marcas específicas como **@param** para indicar un parámetro e **@return** para indicar o valor devolto por unha función. Por exemplo, o código da clase Calcula comentada segundo estas normas quedaría da seguinte forma.

```
/**
 * Clase Calcula
 *
 * Exemplo: Documentación para xeración
 *          automática do documento WSDL
 * @author Víctor Lourido
*/

class Calcula {
    /**
     * Suma dous números e devolve o resultado
     *
     * @param float $a
     * @param float $b
     * @return float
     */
    public function suma($a, $b){
        return $a+$b;
    }

    /**
     * Resta dous números e devolve o resultado
     *
     * @param float $a
     * @param float $b
     * @return float
     */
    public function resta($a, $b){
        return $a-$b;
    }
}
```

Para xerar o documento WSDL a partir da clase Calcula anterior empregando a libraría WSDLDocument, debes crear un novo ficheiro co seguinte código.

```
require_once("Calcula.php");
// Ruta a WSDLDocument
require_once("WSDLDocument.php");

// O servizo publícase na URL http://localhost/servizo.php
$wsdl = new WSDLDocument(
    "Calcula",
    "http://localhost/servizo.php",
    "http://localhost/"
);
```

```
echo $wsdl->saveXml();
```

É dicir, crear un novo obxecto da clase WSDLDocument, e indicar como parámetros:

- O nome da clase que xestionará as peticións ao servizo.
- A URL en que se ofrece o servizo.
- O espazo de nomes destino.

O método **saveXML** obtén como saída o documento WSDL de descrición do servizo. Revísao, pois posiblemente teñas que realizar algúns cambios (por exemplo, pasar o formato de codificación a UTF-8, ou cambiar o nome dalgunha das clases que contén e do seu construtor respectivo).

Cando estea listo, publícao co teu servizo. Para iso, copia o ficheiro obtido nunha ruta accesible vía web (por exemplo, na mesma ruta na que se encontre a clase que xestiona o servizo), e indica a URL en que se encontra cando instancies a clase SoapServer no servizo web (arquivo *servizo.php*).

```
$server = new SoapServer("http://localhost/servizo.wsdl");
$server->setClass('Calcula');
$server->handle();
```

Se engades á URL do servizo o parámetro GET **wsdl**, verás o ficheiro de descrición do servizo. No noso caso a URL sería **http://localhost/servizo.php?wsdl**.

NOTA: Á hora de programar un servizo web é recomendable desactivar a caché WSDL no arquivo php.ini. De non facelo, os arquivos WSDL gárdanse nunha caché polo que os cambios que fagamos neles non terán efecto ata que expire a caché. Faise co parámetro **soap.wsdl_cache_enabled**:

```
soap.wsdl_cache_enabled = 0;
```

--Víctor Lourido 01:59 26 jul 2013 (CEST)