GetGeneratorVersion()

The GetGeneratorVersion() CRUD method allows you get the current version of the generator. While this function does not seem very useful at first sight, it can potentially be used to detect if POG has been upgraded.

GetGeneratorVersion() returns the POG version in a base 64 encoded string

CODE EXAMPLES:

Using NuSOAP:

```
$client = new soapclient ($wsdl, true);
$params = array();
$object = base64_decode($client->call('GetGeneratorVersion', $params));
```

Using PHP SOAP extension (PHP 5+):

```
$client = new SoapClient($wsdl);
$object = base64_decode($client -> GetGeneratorVersion());
```

GenerateObject()

The GenerateObject() SOAP method allows you generate a PHP object by specifying some variables. When all the necessary variables are passed correctly, this will produce the same result obtained when using the main POG interface. This is probably the 2nd most useful SOAP function after "GeneratePackage". (see below)

GenerateObject() takes 6 parameters:

1. $objectName (string)
2. $attributeList (array of strings)
3. $typeList (array of strings)
4. $language (string)
5. $wrapper (string)
6. $pdoDriver (string)

GenerateObject() returns a string representation of the entire object.

CODE EXAMPLES:

Using NuSOAP:

```
$client = new soapclient ($wsdl, true);

$params = array(
        'objectName'    => $objectName,       //e.g. "User"
        'attributeList' => $attributeList,    //e.g. array ("firstName", "lastName")
        'typeList'      => $typeList,         //e.g. array ("VARCHAR(255)", "TEXT")
        'language'      => $language,         //e.g. "php4"
        'wrapper'       => $wrapper,          //e.g. "POG"
        'pdoDriver'     => $pdoDriver         //e.g. "mysql"
        );

$object = base64_decode($client->call('GenerateObject', $params));
```

Using PHP SOAP extension (PHP 5+):

```
$client = new SoapClient($wsdl);
$object = base64_decode($client -> GenerateObject(
                                        $objectName,
                                        $attributeList,
                                        $typeList,
                                        $language,
                                        $wrapper,
                                        $pdoDriver
                                        )
                                );
```

GenerateConfiguration()

The GenerateConfiguration() SOAP method allows you to generate the configuration file for the POG objects you already have. Generally speaking, this SOAP function is not needed except in very special cases.

GenerateConfiguration() returns the string representation of the contents of configuration.php

CODE EXAMPLES:

Using NuSOAP:

```
$client = new soapclient ($wsdl, true);
$params = array('pdoDriver' => $pdoDriver);
$object = base64_decode($client->call('GenerateConfiguration', $params));
```

Using PHP SOAP extension (PHP 5+):

```
$client = new SoapClient($wsdl);
$object = base64_decode($client -> GenerateConfiguration($pdoDriver));
```

GeneratePackage()

*The GeneratePackage() SOAP method is probably the most useful SOAP function and allows you to generate the entire POG package. This SOAP function gives the same result as the zip file provided to POG users who use the main interface. This function is extremely useful as it returns a string representation of the said zip package you can extend with your own content, zip it, and present it to the user of **your** web-application.*

GeneratePackage() takes 6 parameters:

1. $objectName (string)
2. $attributeList (array of strings)
3. $typeList (array of strings)
4. $language (string)
5. $wrapper (string)
6. $pdoDriver (string)

GeneratePackage() returns a string representation of the POG package.

CODE EXAMPLES

Using NuSOAP:

```
$client = new soapclient ($wsdl, true);
$params = array(
        'objectName'    => $objectName,       //e.g. 'User'
        'attributeList' => $attributeList,    //e.g. array ("firstName", "lastName")
        'typeList'      => $typeList,         //e.g. array ("VARCHAR(255)", "TEXT")
        'language'      => $language,         //e.g. 'php4'
        'wrapper'       => $wrapper,          //e.g. 'POG'
        'pdoDriver'     => $pdoDriver         //e.g. 'mysql'
        );
$object = base64_decode($client->call('GeneratePackage', $params));
```

Using PHP SOAP extension (PHP 5+):

```
$client = new SoapClient($wsdl);
$object = base64_decode($client -> GeneratePackage
                                        $objectName,
                                        $attributeList,
                                        $typeList,
                                        $language,
                                        $wrapper,
                                        $pdoDriver
                                        )
                                );
```

GenerateObjectFromLink()

The GenerateObjectFromLink() SOAP method allows you to generate an object from the @link representation of the POG object. This returns the same thing as "GenerateObject()". The only difference is that GenerateObjectFromLink only requires 1 parameter ($link) whereas GenerateObject requires many. Since $link is a self-contained representation of a POG object, you can see why both GenerateObjectFromLink() and GenerateObject() return the same result.

Object() returns true if the deletion was completed successfully, false otherwise

CODE EXAMPLES:

Using NuSOAP:

```
$client = new soapclient ($wsdl, true);
$params = array('link' => $link);
$object = base64_decode($client->call('GenerateObjectFromLink', $params));
```

Using PHP SOAP extension (PHP 5+):

```
$client = new SoapClient($wsdl);
$object = base64_decode($client -> GenerateObjectFromLink($link));
```

PHP OBJECT GENERATOR

SOAP

The purpose of this document is to introduce the POG API to advanced developers who wish to build applications on top of POG. Using the POG API, developers can generate objects without having to go through the POG web interface (http://www.phpobjectgenerator.com). Instead, using SOAP, developers can generate PHP Objects remotely for use in their own web applications

Note:

In all the code samples below you need to set $wsdl. $wsdl is the URL to the WSDL file that describes the POG webservice. Use the url:

http://www.phpobjectgenerator.com/services/soap.php?wsdl

GenerateMapping()

The GenerateMapping() SOAP method allows you generate a PHP Mapping object which is required when you have 2 objects with many-many relations.

GenerateMapping() takes 6 parameters:

1. $objectName1 (string)
2. $objectName2 (string)
3. $language (string)
4. $wrapper (string)
5. $pdoDriver (string)

GenerateMapping() returns a string representation of the entire object.

CODE EXAMPLES:

Using NuSOAP:

```
$client = new soapclient ($wsdl, true);

$params = array(
        'objectName1'  => $objectName,      //e.g. "Author"
        'objectName2' => $$objectName2,     //e.g. "Book"
        'language'     => $language,        //e.g. "php4"
        'wrapper'      => $wrapper,         //e.g. "POG"
        'pdoDriver'    => $pdoDriver        //e.g. "mysql"
        );

$object = base64_decode($client->call('GenerateMapping', $params));
```

Using PHP SOAP extension (PHP 5+):

```
$client = new SoapClient($wsdl);
$object = base64_decode($client -> GenerateMapping(
        $objectName1,
        $objectName2,
        $language,
        $wrapper,
        $pdoDriver
        )
);.
```

USEFUL TIPS

WE LOVE FEEDBACK

Don't be shy, come and talk to us on the POG Google Group:
http://groups.google.com/group/Php-Object-Generator

Subscribe to our RSS feed to obtain the latest news:
http://www.phpobjectgenerator.com/plog/rss/

Send us a hello through email:
pogguys@phpobjectgenerator.com

GenerateFromLink()

The GenerateFromLink() SOAP method allows you to generate a POG package from the @link representation of the POG object. This returns the same thing as "GenerateFromLink()". The only difference is that GenerateFromLink only requires 1 parameter (link) whereas GenerateFromLink requires many. Since $link is a self-contained representation of a POG object, you can see why both GenerateFromLink() and GenerateFromLink() return the same result.

CODE EXAMPLES:

Using NuSOAP:

```
$client = new soapclient ($wsdl, true);

$params = array('link' => $link);

$object = base64_decode($client->call('GenerateFromLink', $params));
```

Using PHP SOAP extension (PHP 5+):

```
$client = new SoapClient($wsdl);

$object = base64_decode($client -> GenerateFromLink($link));
```