Antes de hacer nada la carpeta de la tarea debe estar en la ruta **Xampp → htdocs → DWES06**, o deberás cambiar todas las urls de todos los archivos para que funcione el ejercicio.

Archivos de la Tarea.

Carpeta DWES06

Funciones.php → contiene la clase Servicio_F con las funciones getPVP, getStock, getFamilias y getProductosFamilia. Además cuenta con una función adicional para conectar con la base de datos y ejecutar las consultas.

Servicio.php → hace de "servidor".

Cliente.php → Donde se llama a las funciones de la clase Servicio_F y muestra el resultado.

Carpeta WSDL

servicios.php → archivo con la clase comentada para sacar el wsdl, además de la clase servidor de SOAP.

generar_wsdl.php → archivos para sacar el archivo .wsdl a partir de una clase.

cliente.php → Donde se llama a las funciones de la clase Servicio_F y muestra el resultado, pero esta vez la clase se ha creado a través del wsdl.

servicio.wsdl → resultado de generar_wsdl.php

Carpeta src

WSDLDocument.php → necesario para crear el archivo .wsdl

Carpeta wsdl2php

composer.json,composer.lock, composer.url y carpeta vendor → parte del composer para poder generar la clase de php a partir del wsdl.

generar_php.php → archivo que crea el php a través del composer con un .wsdl.

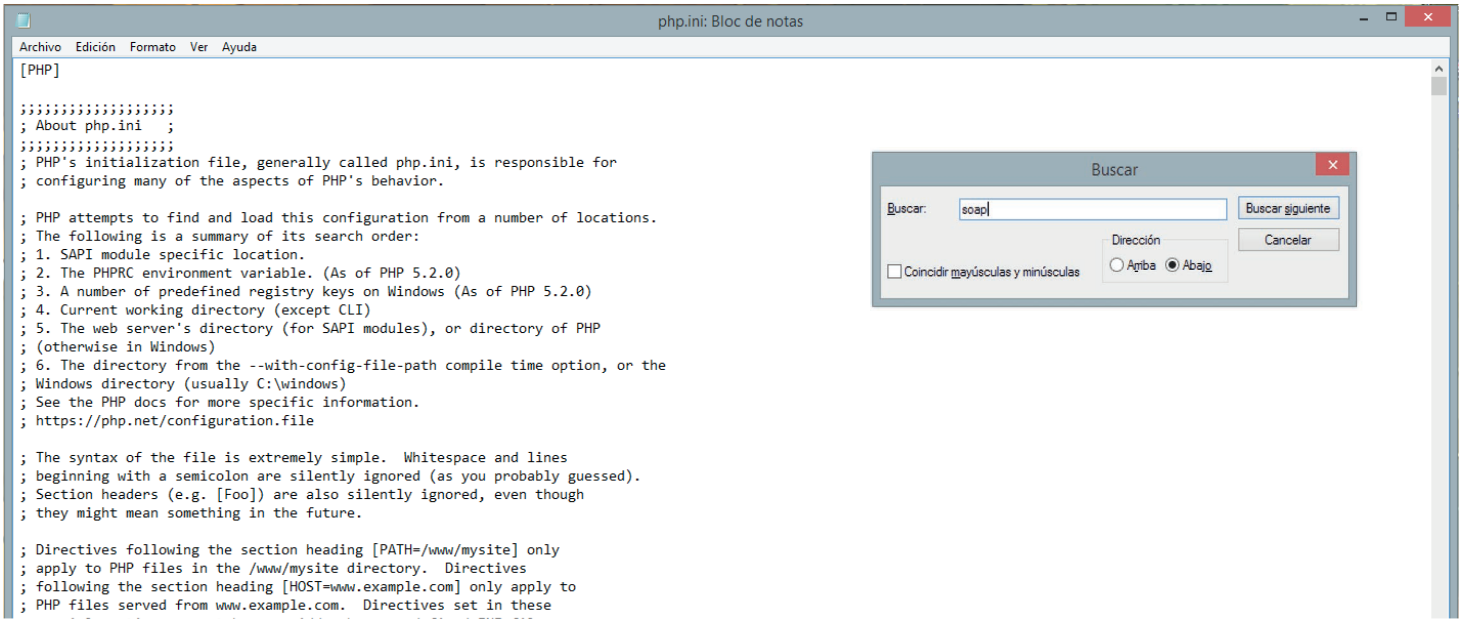
Carpeta src del wsdl2php → resultados de generar_php.php

Servicio_F.php → clase php resultante del wsdl.

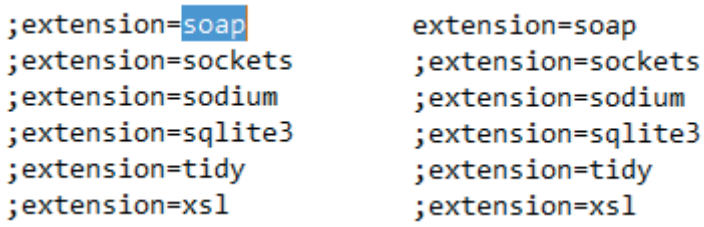
Comencemos con el desarrollo de la Tarea.

ACTIVAR SOAP EN PHP

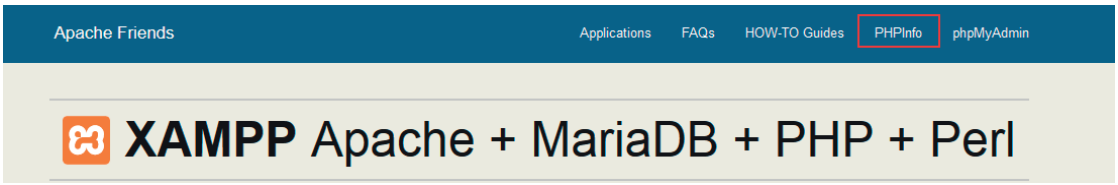
En la carpeta Xampp → php → php.ini. Lo abrimos y con ctrl + B, nos aparece la opción de buscar, escribimos soap y le damos a Buscar siguiente.



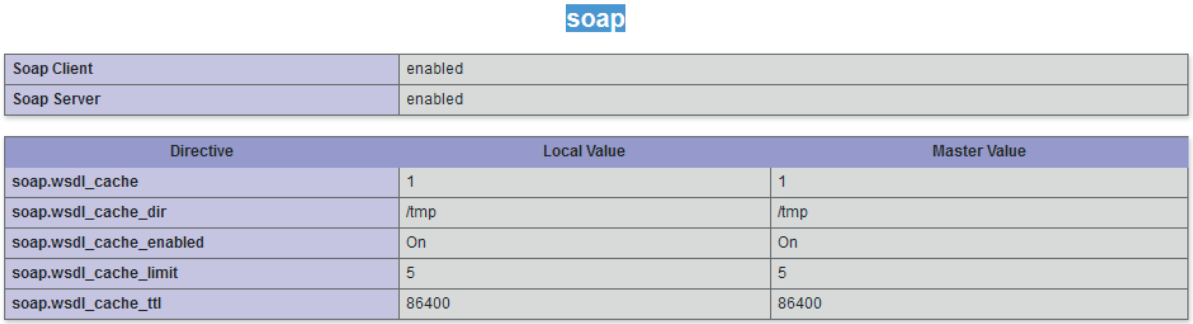
Nos llevará hasta donde debemos descomentar la extensión soap de php. Quitamos el ; . También debemos modificar soap.wsdl_cache_enabled y ponerlo a 0, para poder crear el documento wsdl sin problemas. Guardamos los cambios.



Activamos el servidor local y vamos a la página de inicio del localhost y clickamos en PHPinfo.



Se nos abrirá una nueva ventana, aquí con F3, buscamos soap. Podemos ver que está activado. Pasamos al código de la Tarea.



Pasamos al código de la Tarea.

PHP SOAP

Funciones.php

Creamos el archivo Funciones.php y escribimos las funciones que se nos piden en el enunciado, quedando así.

```
<?php
/*------------------ CLASE SERVICIOS ------------------*/
class Servicio_F{
/*------------------ CONEXION A LA BBDD ------------------*/
    /**
     * Ejecutar consultas sql
     * @param string $sql
     * @return string[] $resultado
     */
    public static function ejecutaConsulta($sql) {
        $db  = new PDO('mysql:host=localhost;dbname=dwes','dwes','abc123.');
        $resultado = null;
        if (isset($db)) $resultado = $db->query($sql);
        return $resultado;
    }//funcion ejecutar consulta
/*------------------ FIN CONEXION A LA BBDD ------------------*/
/*------------------ GETPVP sacar el pvp por el código de producto ------------------*/
    /**
    *  GET PVP Obtiene el PVP con el código de producto
    * @param string $codigo
    * @return float $pvp
    */
    public function getPVP($codigo){
        $pvp="";
        $sql = "SELECT cod, pvp from producto where cod = '".$codigo."' ";
        $result = self::ejecutaConsulta($sql);
        if(isset($result)) {
            $row = $result->fetch();
            $pvp = $row['pvp'];
        }
        return $pvp;
    }//funcion getPVP
/*------------------ FIN GETPVP ------------------*/
/*------------------ GETSTOCK ------------------*/
   /**
    *  GET STOCK  Obtiene el numero de unidades por producto en una tienda determinada
    * @param string $codigo_producto
    * @param int $codigo_tienda
    * @return int $stock
    *
    */
    public function getStock($codigo_producto,$codigo_tienda){
        $stock="";
        $sql = "SELECT producto, tienda, unidades from stock where producto = '".$codigo_pro-
ducto."'AND tienda = '".$codigo_tienda."' ";
        $result = self::ejecutaConsulta($sql);
        if(isset($result)) {
            $row = $result->fetch();
            $stock = $row['unidades'];
        }
        return $stock;
    }//function getStock
/*------------------ FIN GETSTOCK ------------------*/
/*------------------ GETFAMILIAS ------------------*/
   /**
    * GET FAMILIAS  Obtiene la lista de los códigos de las Familias de productos
    * @return string[] $familias
    */
    public function getFamilias(){
        $familias = [];
```

```
        $sql = “SELECT cod from familia”;
        $result = self::ejecutaConsulta($sql);
        $row = $result->fetch();
       while($row != null){
            array_push($familias,$row[‘cod’]);
            $row = $result->fetch();
        }
        return $familias;
    }//function getFamilias
/*------------------ FIN GETFAMILIAS ------------------*/
/*------------------ GETPRODUCTOSFAMILIA ------------------*/
    /**
    * GET PRODUCTOS FAMILIA  Obtiene una lista de codigos de prpducto según el codigo de fami-
lia
    * @param string $codigo_familia
    * @return string[] $productos_familia
    *
    */
    public function getProductosFamilia($codigo_familia){
        $lista_productos=[];
        $sql=”SELECT cod, familia FROM producto WHERE familia =’”.$codigo_familia.”’ “;
        $result=self::ejecutaConsulta($sql);
        $row = $result->fetch();
        while($row != null){
            array_push($lista_productos,$row[‘cod’]);
            $row = $result->fetch();
         }
         return $lista_productos;
    }//function getProductosFamilia
/*------------------ FIN GETPRODUCTOSFAMILIA ------------------*/
}//clase SERVICIOS
?>
```

Servicio.php

En este archivo llamado a Funciones.php para poder crear la clase Servicios_F en el SOAP.

Además de crear un server que hará las comunicaciones necesarias entre las páginas.

```
<?php
    /* ----------- CLASE FUNMCIONES ----------- */
    require_once(‘Funciones.php’);
    /* ----------- URL ----------- */
    $uri=”http://localhost/DWES06/”;
    /* ----------- SERVIDOR SOAP ----------- */
    //creamos una clase server desde la que podremos comunicar las páginas
    $server = new SoapServer(null,array(‘uri’=>$uri));
    /* ----------- CLASES SOAP ----------- */
    //creamos la clase Servicios
    $server->setClass(‘Servicio_F’);
    $server->handle();
?>
```

Cliente.php

Aquí creamos las variables que usamos para sacar los datos de la base de datos. También creamos un cliente para el SOAP y llamamos a las funciones pasando las variables. Con un HTML mostramos los resultados por pantalla.

```
<?php
/* ----------- URL necesarias para el funcionamiento del SOAP ----------- */
$url=”http://localhost/DWES06/Servicio.php”;
$uri=”http://localhost/DWES06/”;
```

```
//VARIABLES QUE VAMOS A UTILIZAR COMO EJEMPLO
$codigo=”3DSNG”;
$codigo_tienda=”1”;
$codigo_familia= “MEMFLA”;
/* ----------- Cliente SOAP ----------- */
$cliente = new SoapClient(null,array(‘location’=>$url,’uri’=>$uri));
/* ----------- LLAMADAS A LAS FUNCIONES ----------- */
//getPVP
$pvp = $cliente->getPVP($codigo);
//getStock
$unidades = $cliente->getStock($codigo, $codigo_tienda);
//getFamilias
$familias = $cliente->getFamilias();
//getProductosFamlia
$productos=$cliente->getProductosFamilia($codigo_familia);
?>

<!DOCTYPE html>
<html lang=”en”>
<head>
    <meta charset=”UTF-8”>
    <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>
    <title>Cliente</title>
</head>
<body>
    <!--Muestra los resultados de GETPVP  -->
    <p>El precio del producto <?php echo $codigo;?> es: <?php echo $pvp;?> </p>
    <hr>
    <!--Muestra los resultados de GETSTOCK  -->
    <p>En la tienda <?php echo $codigo_tienda;?> hay: <?php echo $unidades;?> del producto
<?php echo $codigo;?> </p>
    <hr>
    <!--Muestra los resultados de GETFAMILIAS  -->
    <p>La lsita de familias es : <?php var_dump($familias); ?></p>
    <hr>
    <!--Muestra los resultados de GETPRODUCTOSFAMILIAS  -->
    <p>La lsita de los codigo de producto de la famlia <?php echo $codigo_familia ?>  es :
<?php var_dump($productos); ?></p>
    <hr>

</body>
</html>
```

El precio del producto 3DSNG es: 270.00

En la tienda 1 hay: 2 del producto 3DSNG

La lsita de familias es : array(15) { [0]=> string(6) "CAMARA" [1]=> string(6) "CONSOL" [2]=> string(5) "EBOOK" [3]=> string(6) "IMPRES" [4]=> string(6) "MEMFLA" [5]=> string(3) "MP3" [6]=> string(6) "MULTIF" [7]=> string(6) "NETBOK" [8]=> string(6) "ORDENA" [9]=> string(6) "PORTAT" [10]=> string(6) "ROUTER" [11]=> string(3) "SAI" [12]=> string(6) "SOFTWA" [13]=> string(2) "TV" [14]=> string(6) "VIDEOC" }

La lsita de los codigo de producto de la famlia MEMFLA es : array(4) { [0]=> string(10) "KSTDT101G2" [1]=> string(12) "KSTDTG332GBR" [2]=> string(11) "KSTMSDHC8GB" [3]=> string(12) "TSSD16GBC10J" }

Una vez comprobado que esto funciona, pasamos a la siguiente parte.

WSDL

Lo primero de todo es comentar la clase Funciones para poder pasarla a .wsdl. Con esto creamos serviciow. Le añadimos también lo de crear el SOAP server y la clase de este.

serviciow.php

```
<?php
    /* ----------- URL ----------- */
    //$uri=”http://localhost/DWES06/WSDL/”;
    $uri_xml=”http://localhost/DWES06/WSDL/serviciow.wsdl”;
class Servicio_Fw{
      /*------------------ CONEXION A LA BBDD ------------------*/
          /**
           * Ejecutar consultas sql
           * @param string $sql
```

```
           * @return string[] $resultado
           */
          public static function ejecutaConsulta($sql) {
              $db  = new PDO('mysql:host=localhost;dbname=dwes','dwes','abc123.');
              $resultado = null;
              if (isset($db)) $resultado = $db->query($sql);
              return $resultado;
          }//funcion ejecutar consulta
      /*------------------ FIN CONEXION A LA BBDD ------------------*/
      /*------------------ GETPVP sacar el pvp por el código de producto ------------------*/
          /**
          *  GET PVP Obtiene el PVP con el código de producto
          * @param string $codigo
          * @return float $pvp
          */
          public function getPVP($codigo){
              $pvp="";
              $sql = "SELECT cod, pvp from producto where cod = '".$codigo."' ";
              $result = self::ejecutaConsulta($sql);
              if(isset($result)) {
                  $row = $result->fetch();
                  $pvp = $row['pvp'];
              }
              return $pvp;
          }//funcion getPVP
      /*------------------ FIN GETPVP ------------------*/
      /*------------------ GETSTOCK ------------------*/
         /**
          *  GET STOCK  Obtiene el numero de unidades por producto en una tienda determinada
          * @param string $codigo_producto
          * @param int $codigo_tienda
          * @return int $stock
          *
          */
          public function getStock($codigo_producto,$codigo_tienda){
              $stock="";
              $sql = "SELECT producto, tienda, unidades from stock where producto = '".$codi-
go_producto."'AND tienda = '".$codigo_tienda."' ";
              $result = self::ejecutaConsulta($sql);
              if(isset($result)) {
                  $row = $result->fetch();
                  $stock = $row['unidades'];
              }
              return $stock;
          }//function getStock
      /*------------------ FIN GETSTOCK ------------------*/
      /*------------------ GETFAMILIAS ------------------*/
         /**
          * GET FAMILIAS  Obtiene la lista de los códigos de las Familias de productos
          * @return string[] $familias
          */
          public function getFamilias(){
              $familias = [];
              $sql = "SELECT cod from familia";
              $result = self::ejecutaConsulta($sql);
              $row = $result->fetch();
             while($row != null){
                  array_push($familias,$row['cod']);
                  $row = $result->fetch();
              }
              return $familias;
          }//function getFamilias
      /*------------------ FIN GETFAMILIAS ------------------*/
      /*------------------ GETPRODUCTOSFAMILIA ------------------*/
          /**
          * GET PRODUCTOS FAMILIA  Obtiene una lista de codigos de prpducto según el codigo de
familia
          * @param string $codigo_familia
          * @return string[] $productos_familia
```

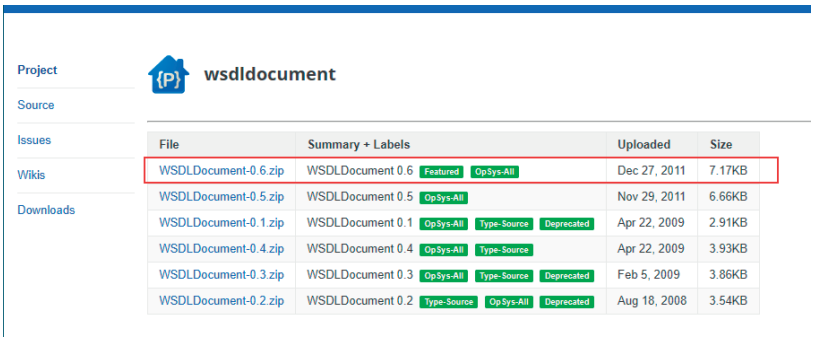
```
         *
         */
         public function getProductosFamilia($codigo_familia){
             $lista_productos=[];
             $sql="SELECT cod, familia FROM producto WHERE familia ='".$codigo_familia."' ";
             $result=self::ejecutaConsulta($sql);
             $row = $result->fetch();
             while($row != null){
                  array_push($lista_productos,$row['cod']);
                  $row = $result->fetch();
              }
              return $lista_productos;
         }//function getProductosFamilia
      /*------------------ FIN GETPRODUCTOSFAMILIA ------------------*/

}//clase SERVICIOS

    /* ----------- SERVIDOR SOAP ----------- */
    //creamos una clase server desde la que podremos comunicar las páginas
    //$server = new SoapServer(null,array('uri'=>$uri));
    $server = new SoapServer($uri_xml);
    /* ----------- CLASES SOAP ----------- */
    //creamos la clase Servicios
    $server->setClass('Servicio_Fw');
    $server->handle();

?>
```

Ahora instalamos WSDL desde, **https://code.google.com/archive/p/wsdldocument/**, Descargamos la última versión y lo colocamos en una carpeta en el proyecto. El WSDLDocument.php nos ayuda a generar el archivo .wsdl.



Creamos un archivo que se llama generar_wsdl.php donde incluimos el WSDLDocument.php y Funciones.php.

Volvemos al link y copiamos el código que aparece en el apartado Proyect y lo copiamos en el archivo generar_wsdl.php.

generar_wsdl.php

```
<?php
require_once('../Funciones.php');//archivo que contiene las clases
require_once('src/WSDLDocument.php');
 /* ----------- URL ----------- */
$uri="http://localhost/DWES06/WSDL/";
/* ----------- SERVIDOR SOAP ----------- */
//creamos una clase server desde la que podremos comunicar las páginas
$server = new SoapServer(null,array('uri'=>$uri));
/* ----------- CLASES SOAP ----------- */
//creamos la clase Servicios
$server->setClass('Servicio_F');
$server->handle();
$wsdl = new WSDLDocument( "Servicio_F","http://localhost/DWES06/Servicio.php",$uri ); echo
$wsdl->saveXml();
?>
```

Con esto ya podemos generar el .wsdl.

Con el servidor local activo, vamos al archivo y lo ejecutamos. Deberán salirnos en pantalla los comentarios de la clase Servicios_F.

GET PVP Obtiene el PVP con el código de productoGET STOCK Obtiene el numero de unidades por producto en una tienda determinadaGET FAMILIAS Obtiene la lista de los códigos de las Familias de productosGET PRODUCTOS FAMILIA Obtiene una lista de codigos de prpducto según el codigo de familia

Con botón derecho le damos a Ver código fuente de la página y se nos abrirá un xml, lo copiamos y creamos un archivo llamado serviciow.wsdl y pegamos el contenido.

Ajuste de línea

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:soap-enc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:soap-env="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://localhost/DWES06/WSDL/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="http://localhost/DWES06/WSDL/"><wsdl:types><xsd:schema
targetNamespace="http://localhost/DWES06/WSDL/"><xsd:complexType name="stringArray"><xsd:complexContent><xsd:restriction
base="soap-enc:Array"><xsd:attribute ref="soap-enc:arrayType" wsdl:arrayType="xsd:string[]"/></xsd:restriction>
</xsd:complexContent></xsd:complexType></xsd:schema></wsdl:types><wsdl:message name="getPVPRequest"><wsdl:part name="codigo"
type="xsd:string"/></wsdl:message><wsdl:message name="getPVPResponse"><wsdl:part name="getPVPReturn" type="xsd:float"/>
</wsdl:message><wsdl:message name="getStockRequest"><wsdl:part name="codigo_producto" type="xsd:string"/><wsdl:part
name="codigo_tienda" type="xsd:int"/></wsdl:message><wsdl:message name="getStockResponse"><wsdl:part name="getStockReturn"
type="xsd:int"/></wsdl:message><wsdl:message name="getFamiliasRequest"/><wsdl:message name="getFamiliasResponse"><wsdl:part
name="getFamiliasReturn" type="tns:stringArray"/></wsdl:message><wsdl:message name="getProductosFamiliaRequest"><wsdl:part
name="codigo_familia" type="xsd:string"/></wsdl:message><wsdl:message name="getProductosFamiliaResponse"><wsdl:part
name="getProductosFamiliaReturn" type="tns:stringArray"/></wsdl:message><wsdl:portType name="Servicio_FPortType"><wsdl:operation
name="getPVP"><wsdl:documentation>GET PVP Obtiene el PVP con el código de producto</wsdl:documentation><wsdl:input
message="tns:getPVPRequest"/><wsdl:output message="tns:getPVPResponse"/></wsdl:operation><wsdl:operation name="getStock">
<wsdl:documentation>GET STOCK  Obtiene el numero de unidades por producto en una tienda determinada</wsdl:documentation><wsdl:input
message="tns:getStockRequest"/><wsdl:output message="tns:getStockResponse"/></wsdl:operation><wsdl:operation name="getFamilias">
<wsdl:documentation>GET FAMILIAS  Obtiene la lista de los códigos de las Familias de productos</wsdl:documentation><wsdl:input
message="tns:getFamiliasRequest"/><wsdl:output message="tns:getFamiliasResponse"/></wsdl:operation><wsdl:operation
name="getProductosFamilia"><wsdl:documentation>GET PRODUCTOS FAMILIA  Obtiene una lista de codigos de prpducto según el codigo de
familia</wsdl:documentation><wsdl:input message="tns:getProductosFamiliaRequest"/><wsdl:output
message="tns:getProductosFamiliaResponse"/></wsdl:operation></wsdl:portType><wsdl:binding name="Servicio_FBinding"
type="tns:Servicio_FPortType"><soap-env:binding xmlns="http://schemas.xmlsoap.org/wsdl/soap/" style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/><wsdl:operation name="getPVP"><soap-env:operation
xmlns="http://schemas.xmlsoap.org/wsdl/soap/" soapAction="http://localhost/DWES06/Servicio.php?method=getPVP" style="rpc"/>
<wsdl:input><soap-env:body xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></wsdl:input><wsdl:output><soap-env:body
xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
</wsdl:output></wsdl:operation><wsdl:operation name="getStock"><soap-env:operation xmlns="http://schemas.xmlsoap.org/wsdl/soap/"
soapAction="http://localhost/DWES06/Servicio.php?method=getStock" style="rpc"/><wsdl:input><soap-env:body
xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
</wsdl:input><wsdl:output><soap-env:body xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></wsdl:output></wsdl:operation><wsdl:operation name="getFamilias"><soap-
env:operation xmlns="http://schemas.xmlsoap.org/wsdl/soap/" soapAction="http://localhost/DWES06/Servicio.php?method=getFamilias"
style="rpc"/><wsdl:input><soap-env:body xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></wsdl:input><wsdl:output><soap-env:body
xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
</wsdl:output></wsdl:operation><wsdl:operation name="getProductosFamilia"><soap-env:operation
xmlns="http://schemas.xmlsoap.org/wsdl/soap/" soapAction="http://localhost/DWES06/Servicio.php?method=getProductosFamilia"
style="rpc"/><wsdl:input><soap-env:body xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></wsdl:input><wsdl:output><soap-env:body
xmlns="http://schemas.xmlsoap.org/wsdl/soap/" use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
</wsdl:output></wsdl:operation></wsdl:binding><wsdl:service name="Servicio_F"><wsdl:documentation/><wsdl:port name="Servicio_FPort"
binding="tns:Servicio_FBinding"><soap-env:address location="http://localhost/DWES06/Servicio.php"/></wsdl:port></wsdl:service>
</wsdl:definitions>
```

Creamos también clientew.php para poder comprobar que todo funciona. simplemente es copiar y pegar Cliente.php y modificarlo para que también coja la información del xml.

clientew.php

```
<?php
require_once('src/WSDLDocument.php');

require_once('wsdl2php/src/Servicio_F.php');

/* ----------- URL necesarias para el funcionamiento del SOAP ----------- */

//C:\xampp\htdocs\DWES06\WSDL\wsdl2php\src\Servicio_F.php
$url="http://localhost/DWES06/WSDL/wsdl2php/src/Servicio_F.php";
$uri="http://localhost/DWES06/WSDL";

$uri_xml="http://localhost/DWES06/WSDL/serviciow.wsdl";

//VARIABLES QUE VAMOS A UTILIZAR COMO EJEMPLO
$codigo="3DSNG";
$codigo_tienda="1";
$codigo_familia= "MEMFLA";

/* ----------- Cliente SOAP ----------- */
$cliente = new Servicio_F();
//$cliente = new SoapClient($uri_xml);
//$cliente = new SoapClient(null,array('location'=>$url,'uri'=>$uri));
```

```
/* ----------- LLAMADAS A LAS FUNCIONES ----------- */
//getPVP
$pvp = $cliente->getPVP($codigo);
//getStock
$unidades = $cliente->getStock($codigo, $codigo_tienda);
//getFamilias
$familias = $cliente->getFamilias();
//getProductosFamlia
$productos=$cliente->getProductosFamilia($codigo_familia);
?>

<!DOCTYPE html>
<html lang=”en”>
<head>
    <meta charset=”UTF-8”>
    <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>
    <title>Cliente</title>
</head>
<body>
    <!--Muestra los resultados de GETPVP  -->
    <p>El precio del producto <?php echo $codigo;?> es: <?php echo $pvp;?> </p>
    <hr>
    <!--Muestra los resultados de GETSTOCK  -->
    <p>En la tienda <?php echo $codigo_tienda;?> hay: <?php echo $unidades;?> del producto
<?php echo $codigo;?> </p>
    <hr>
    <!--Muestra los resultados de GETFAMILIAS  -->
    <p>La lsita de familias es : <?php var_dump($familias); ?></p>
    <hr>
    <!--Muestra los resultados de GETPRODUCTOSFAMILIAS  -->
    <p>La lsita de los codigo de producto de la famlia <?php echo $codigo_familia ?>  es :
<?php var_dump($productos); ?></p>
    <hr>

</body>
</html>
```

Cuando comprobemos que todo está correcto, comentamos en el cliente.php la parte del xml, pues solo es necesaria para comprobar que el archivo .wsdl está bien generado. Resultado que sale.

El precio del producto 3DSNG es: 270.00

En la tienda 1 hay: 2 del producto 3DSNG

La lsita de familias es : array(15) { [0]=> string(6) "CAMARA" [1]=> string(6) "CONSOL" [2]=> string(5) "EBOOK" [3]=> string(6) "IMPRES" [4]=> string(6) "MEMFLA" [5]=> string(3) "MP3" [6]=> string(6) "MULTIF" [7]=> string(6) "NETBOK" [8]=> string(6) "ORDENA" [9]=> string(6) "PORTAT" [10]=> string(6) "ROUTER" [11]=> string(3) "SAI" [12]=> string(6) "SOFTWA" [13]=> string(2) "TV" [14]=> string(6) "VIDEOC" }

La lsita de los codigo de producto de la famlia MEMFLA es : array(4) { [0]=> string(10) "KSTDT101G2" [1]=> string(12) "KSTDTG332GBR" [2]=> string(11) "KSTMSDHC8GB" [3]=> string(12) "TSSD16GBC10J" }

WSDL2PHP

Instalamos el composer para poder generar una clase php desde un wsdl, con la ayuda de este video **https://www.youtube.com/watch?v=vN-NT4cVR0A** (algunos pasos no serán necesarios)

Descargamos composer desde su página web **https://getcomposer.org/**

Una vez descargado, le damos doble click y durante la instalación le daremos a todo que sí, como en el video.

Una vez acabe la instalación, abrimos el cmd y escribimos **composer**. Si lo hemos instalado bien saldrá la ayuda de comandos.

Cerramos y volvemos a abrir el cmd, con el comando **cd** nos movemos entre carpetas hasta llegar hasta nuestro proyecto, donde creamos una carpeta que contendrá los archivos del composer necesarios para su funcionamiento. En mi caso la carpeta es wsdl2php.

El siguiente paso es introducir el comando **composer init**, donde nos hará una serie de preguntas para crear un archivo json. Debería quedar algo así.

composer.json

```
{
    “name”: “celia_rd/dwes06”,
    “description”: “de wsdl a php class”,
    “type”: “project”,
    “license”: “mit”,
    “authors”: [
        {
            “name”: “celia rd”
        }
    ],
    “minimum-stability”: “beta”,
    “require”: {
        “wsdl2phpgenerator/wsdl2phpgenerator”: “^3.4”
    }
}
```

Lo siguiente es validar este archivo con **composer validate**. Si hay algún error nos avisará, en mi caso me faltó la descripción, una vez que la añadí, me lo validó. Y para actualizar con poner **composer update**, es suficiente.

Una vez instalado y actualizado, escribimos en el cmd c**omposer requiere wsdl2phpgenerator/wsdl2phpgenerator**. Esto nos creará una carpeta vendor en la misma localización que el archivo json. Dentro de la carpeta estará el archivo autoload que nos servirá más tarde.

Creamos el archivo generar_php.php.

Después vamos a esta página donde podrá obtener el código necesario para general la clase php desde el wsdl. **https://github.com/wsdl2phpgenerator/wsdl2phpgenerator/blob/master/docs/usage-and-options.md**

Copiamos el código que necesitamos en generar_php.php.Además de incluir el archivo autoload.php, quedando así.

generar_php.php

```
<?php
    require(‘vendor/autoload.php’);
    $generator = new \Wsdl2PhpGenerator\Generator();
    $generator->generate(
        new \Wsdl2PhpGenerator\Config(array(
            ‘inputFile’ => ‘../serviciow.wsdl’, //donde esta el wsdl.xml
            ‘outputDir’ => ‘src/’
        ))
    );
?>
```

Vamos a nuestro navegador y lo ejecutamos. Si todo ha salido bien, en la carpeta que hemos elegido se habrá creado la clase. En mi caso elegí crear la carpeta src dentro de wsdl2php, asi que la clase Servicio _F.php se encontrará allí.

Servicio _F.php

```
<?php
class Servicio_F extends \SoapClient
{
    /**
     * @var array $classmap The defined classes
     */
    private static $classmap = array (
  ‘stringArray’ => ‘\\stringArray’,
);
```

```
    /**
     * @param array $options A array of config values
     * @param string $wsdl The wsdl file to use
     */
    public function __construct(array $options = array(), $wsdl = null)
    {

  foreach (self::$classmap as $key => $value) {
    if (!isset($options['classmap'][$key])) {
      $options['classmap'][$key] = $value;
    }
  }
      $options = array_merge(array (
  'features' => 1,
), $options);
      if (!$wsdl) {
        $wsdl = 'http://localhost/DWES06/WSDL/serviciow.wsdl';
      }
      parent::__construct($wsdl, $options);
    }

    /**
     * GET PVP Obtiene el PVP con el código de producto
     *
     * @param string $codigo
     * @return float
     */
    public function getPVP($codigo)
    {
      return $this->__soapCall('getPVP', array($codigo));
    }

    /**
     * GET STOCK  Obtiene el numero de unidades por producto en una tienda determinada
     *
     * @param string $codigo_producto
     * @param int $codigo_tienda
     * @return int
     */
    public function getStock($codigo_producto, $codigo_tienda)
    {
      return $this->__soapCall('getStock', array($codigo_producto, $codigo_tienda));
    }

    /**
     * GET FAMILIAS  Obtiene la lista de los códigos de las Familias de productos
     *
     * @return stringArray
     */
    public function getFamilias()
    {
      return $this->__soapCall('getFamilias', array());
    }

    /**
     * GET PRODUCTOS FAMILIA  Obtiene una lista de codigos de prpducto según el codigo de fami-
lia
     *
     * @param string $codigo_familia
     * @return stringArray
     */
    public function getProductosFamilia($codigo_familia)
    {
      return $this->__soapCall('getProductosFamilia', array($codigo_familia));
    }

}
```

Ahora en clienetw.php creamos una nueva clase llamado Servicio_F.

No debemos olvidarnos de cambiar la ruta del archivo wsdl o nos dará error.

Original

```
$options = array_merge(array (
features' => 1,
$options);
    if (!$wsdl) {
        $wsdl = '/../serviciow.wsdl';
    }
    parent::__construct($wsdl, $options);
}
```

Modificada

```
$options = array_merge(array (
atures' => 1,
ptions);
if (!$wsdl) {
    $wsdl = 'http://localhost/DWES06/WSDL/serviciow.wsdl'
}
parent::__construct($wsdl, $options);
```

Lo ejecutamos y comprobamos que sale exactamente igual que las otras dos veces.

El precio del producto 3DSNG es: 270.00

En la tienda 1 hay: 2 del producto 3DSNG

La lsita de familias es : array(15) { [0]=> string(6) "CAMARA" [1]=> string(6) "CONSOL" [2]=> string(5) "EBOOK" [3]=> string(6) "IMPRES" [4]=> string(6) "MEMFLA" [5]=> string(3) "MP3" [6]=> string(6) "MULTIF" [7]=> string(6) "NETBOK" [8]=> string(6) "ORDENA" [9]=> string(6) "PORTAT" [10]=> string(6) "ROUTER" [11]=> string(3) "SAI" [12]=> string(6) "SOFTWA" [13]=> string(2) "TV" [14]=> string(6) "VIDEOC" }

La lsita de los codigo de producto de la famlia MEMFLA es : array(4) { [0]=> string(10) "KSTDT101G2" [1]=> string(12) "KSTDTG332GBR" [2]=> string(11) "KSTMSDHC8GB" [3]=> string(12) "TSSD16GBC10J" }