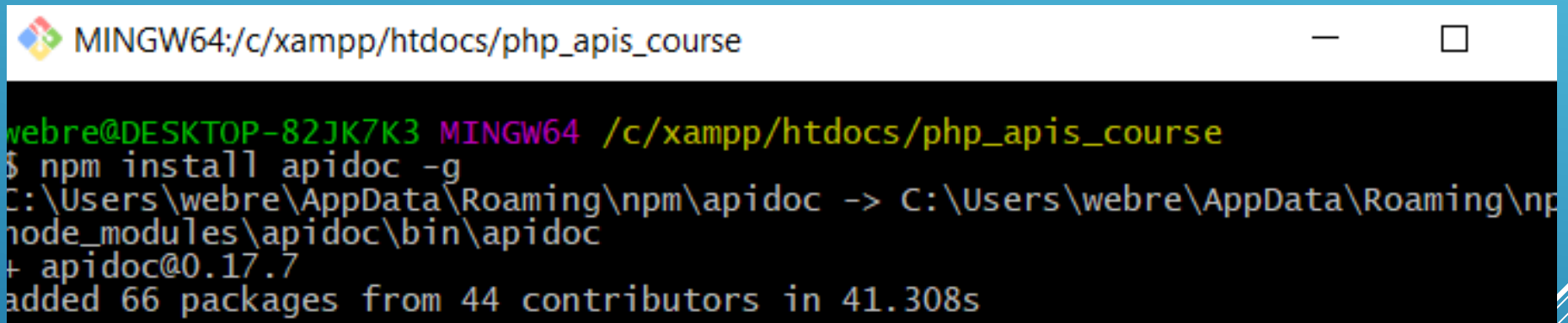




HOW THE OTHER TEAMS WILL UNDERSTAND
YOUR API TO CONSUME IT ???

- ▶ Install npm with nodejs
- ▶ Install APIDOC
- ▶ Npm install apidoc -g



```
MINGW64:/c:/xampp/htdocs/php_apis_course

webre@DESKTOP-82JK7K3 MINGW64 /c:/xampp/htdocs/php_apis_course
$ npm install apidoc -g
C:\Users\webre\AppData\Roaming\npm\apidoc -> C:\Users\webre\AppData\Roaming\np
node_modules\apidoc\bin\apidoc
+ apidoc@0.17.7
added 66 packages from 44 contributors in 41.308s
```

DOCUMENTATION STEPS

- ▶ Create a configuration file for apidoc like this one

```
{  
  "name": "Items-API",  
  "version": "0.1.0",  
  "description": "basic example API for Glasses Shop",  
  "title": "API for Glasses shop",  
  "url" : "http://localhost/php_apis_course/REST02.php/items"  
}
```

- ▶ Document your functions like this ->

```
/**  
 * @api {get} /itemapidoc /:id Request item information  
 * @apiName Getitem  
 * @apiVersion 1.1.1  
 * @apiGroup item *  
 * @apiParam {Number} id items unique ID. *  
 * @apiSuccess {Number} id of the Item.  
 * @apiSuccess {String} Product_code of the Item.  
 * @apiSuccessExample Success-Response:  
 *   HTTP/1.1 200 OK  
 *   {  
     "id": "100",  
     "product_code": "NWTCFV-100",  
     "product_name": "new_glass very new1 ",  
     "photo": "09.png",  
     "list_price": "14.00",  
     "reorder_level": "10",  
     "units_in_stock": "4",  
     "category": "sunglasses",  
     "country": "USA",  
     "rating": "4.60",  
     "discontinued": "1",  
     "date": "2018-08-28 22:52:14"
```

- ▶ Run command
apidoc -i Model/ -o apidoc/
- ▶ Enjoy the help for your API
- ▶ Send it to the front end and Mobile team or any team that will consume your REST API

The screenshot shows a web-based API documentation tool. On the left is a sidebar with a search bar labeled 'Filter...' and a list of endpoints under the 'item' resource: '/:id Request item information', 'Add a new item information', 'Delete an item', and 'Update an item information'. The main panel displays the selected endpoint, 'item - /:id Request item information', with a version '1.1.1' in the top right. A green 'GET' button is visible. Below this is a black box containing the URL 'http://localhost/php_apis_course/REST02.php/items/itemapidoc'. A 'Parameter' section follows, containing a table with one parameter: 'id' of type 'Number' and description 'items unique ID.'. At the bottom, a 'Success 200' section contains an empty table with headers 'Field', 'Type', and 'Description'.

Filter... x

item

/:id Request item information
Add a new item information
Delete an item
Update an item information

item

item - /:id Request item information 1.1.1

GET

http://localhost/php_apis_course/REST02.php/items/itemapidoc

Parameter

Field	Type	Description
id	Number	items unique ID.

Success 200

Field	Type	Description
-------	------	-------------



SOAP APIS

Part 4 (Last Part)

- ▶ XML is another data formats to use with API beside JSON
- ▶ Human Readable
- ▶ Easy to parse and Query
- ▶ Better structural

```
<?xml version="1.0"?>  
<list>  
  <item>eggs</item>  
  <item>bread</item>  
  <item>milk</item>  
  <item>bananas</item>  
  <item>bacon</item>  
  <item>cheese</item>  
</list>
```

PHP AND XML

1. You are Designing a Backend in your company for a Mobile and front end web team to have an interface to manage some data (CRUD operations)
2. You are working in a tourism API for hotel availability (like booking.com) and returning data about room types , availabilities, prices and amenities (internet , breakfast,..etc) this API will be consumed by many other websites .
3. You are making an API for prayer times per city that will be consumed by many websites

WILL YOU CHOOSE JSON OR XML IN THESE SITUATIONS AND WHY?

Code

Output

```
<?php
$list = array("eggs","milk","banana","chease");
//Creating The parent element
$xml = new SimpleXMLElement("<list/>");
foreach($list as $item){
    //adding child to the parent element
    $xml->addChild("item",$item);
}
//output
header('Content-type: text/xml');
echo $xml->asXML();
exit();
```

```
<?xml version="1.0"?>
<list>
  <item>eggs</item>
  <item>bread</item>
  <item>milk</item>
  <item>bananas</item>
  <item>bacon</item>
  <item>cheese</item>
</list>
```



CONVERT PHP ARRAY TO XML

1. SimpleXMLElement()
2. DOMDocument
3. XML Reader
4. XML Writer

- ▶ addChild(\$key,\$value)
- ▶ AddAttribute (\$key,\$value)
- ▶ AsXML(\$filename)

```
<hotels>
  <hotel name="Queens Hotel">
    <rooms>17</rooms>
    <price>150</price>
  </hotel>
  <hotel name="Kings Hotel">
    <rooms>12</rooms>
    <price>150</price>
```

XML CLASSES IN PHP



Code

```
$document = new SimpleXMLElement("<hotels />");  
  
$kings = $document->addChild("hotel");  
$kings->addAttribute("name", "Kings Hotel");  
$kings->addChild("rooms", 12);  
$kings->addChild("price", 150);
```

OutPut

```
<hotels>  
  <hotel name="Queens Hotel">  
    <rooms>17</rooms>  
    <price>150</price>  
  </hotel>  
  <hotel name="Kings Hotel">  
    <rooms>12</rooms>  
    <price>150</price>
```

SECOND EXAMPLE

Code

OutPut

```
<?php
$xml = new SimpleXMLElement(file_get_contents("http://localhost/php\_apis\_course/
foreach($xml->children() as $item) {
    $arr[] = $item->__toString();
}
var_dump($arr);
```

```
array(4) {
    [0]=>
    string(4) "eggs"
    [1]=>
    string(4) "milk"
    [2]=>
    string(6) "banana"
    [3]=>
    string(6) "chease"
}
```

PARSING XML TO PHP ARRAY

SOAP

- ▶ Simple Object access protocol
- ▶ RPC style services
- ▶ Uses WSDL
- ▶ Based on XML only
- ▶ One URL with methods as parameters
- ▶ SOAP UI

WSDL

- ▶ **A** - WSDL is the standard format for describing a web service.
- ▶ **B** - WSDL definition describes how to access a web service and what operations it will perform.
- ▶ **C** - WSDL is a language for describing how to interface with XML-based services

SOAP AND WSDL

REST

- ▶ Mainly JSON based but can use XML and text
- ▶ Only HTTP
- ▶ One end point per resource (pretty URL)
- ▶ offer an effective way for interacting with lightweight clients, such as smartphones.
- ▶ A REST client is more like a browser. It's a generic client that knows how to use a protocol and standardized methods,
- ▶ 70% of public APIs are REST

SOAP

- ▶ XML based
- ▶ HTTP and other protocols (SMTP)
- ▶ WSDL
- ▶ One end point
- ▶ SOAP, the client needs previous knowledge on everything it will be using, or it won't even begin the interaction

COMPARISON SHEET

- ▶ Simple Object Access Protocol
- ▶ Uses XML
- ▶ WSDL Example :
<http://api.radioreference.com/soap2/?wsdl&v=latest>

```
</operation>
▼<operation name="getTrsFlavor">
  ▼<documentation>
    Returns a trunked system flavor description based on flavor ID. If no flavor ID is specified all TRS flavors will be returned
  </documentation>
  <input message="tns:getTrsFlavorRequest"/>
  <output message="tns:getTrsFlavorResponse"/>
</operation>
▼<operation name="getTrsVoice">
  ▼<documentation>
    Returns a trunked system voice description based on voice ID. If no voice ID is specified all TRS voice types will be returned
  </documentation>
  <input message="tns:getTrsVoiceRequest"/>
  <output message="tns:getTrsVoiceResponse"/>
</operation>
▼<operation name="getCountryList">
  <documentation>Returns the list of countries</documentation>
  <input message="tns:getCountryListRequest"/>
  <output message="tns:getCountryListResponse"/>
</operation>
▼<operation name="getCountryInfo">
  ▼<documentation>
    Returns assigned state/province/region type and agency entities assigned to a country
  </documentation>
  <input message="tns:getCountryInfoRequest"/>
  <output message="tns:getCountryInfoResponse"/>
</operation>
▼<operation name="getStateInfo">
```

```
▼<operation name="getCountryList">
  <documentation>Returns the list of countries</documentation>
  <input message="tns:getCountryListRequest"/>
  <output message="tns:getCountryListResponse"/>
</operation>
```

SOAP

Consuming a SOAP SERVICE

- ▶ SOAPClient (?)

Think as PHP, what should the constructor of this class takes ?

Creating a SOAP Service

- ▶ SOAPServer(?)

- ▶ Think as PHP what else is needed ?

SOAP IN PHP



1. Make sure SOAP extension is installed in your PHP
 - ▶ Phpinfo
 - ▶ Extension_loaded
2. Create object from soapClient Class and pass the WSDL file of your service to its constructor
3. Call any method of the WSDL
4. Enjoy

CONSUMING A SOAP SERVICE




```
//phpinfo();  
$client = new SoapClient("http://api.radioreference.com/soap2/?wsdl&v=latest");  
if (extension_loaded("soap")) {  
    //The name of the operation as in the WSDL  
    $countries = $client->getCountryList();  
    foreach ($countries as $country) {  
        echo "<h4> Country : " . $country->countryName . "</h4>";  
        echo "<h6> Code : " . $country->countryName . "</h6>";  
    }  
}else{  
    die("soap is not loaded");  
}
```

LET SEE WE WANT TO CALL THE GET
COUNTRIES LIST FUNCTION



- ▶ Can you test same service via just SOAP UI ?



SOAP UI

OUTPUT

Country : Afghanistan

Code : Afghanistan

Country : Albania

Code : Albania

Country : Algeria

Code : Algeria

Country : American Samoa

Code : American Samoa

Country : Andorra

Code : Andorra

Country : Angola

Code : Angola

1. Prepare the class which includes your methods (api documentation is a MUST)
2. Install PHP2WSDL tool via our sweet composer
3. Generate the WSDL file for your service
4. Create the service
5. Test by SOAP UI

CREATING A SOAP WEB SERVICE VIA PHP STEP BY STEP

```
/**
 * Get all the courses
 *
 * @return array The collection of courses
 */
```

```
public function get_courses() {
```

```
    return $this->courses;
}
```

```
/**
 * Get specific course by id
 * @param int $id course id
 * @return array The course detail
 */
```

```
public function get_course_id($id){
    return $this->courses[$id];
}
```

PREPARE THE CLASS

NO DOCUMENTATION = WSDL PROBLEMS

```
{  
    "require": {  
        "guzzlehttp/guzzle": "~6.0",  
        "monolog/monolog": ">=1.12.0",  
        "php2wsdl/php2wsdl" : "~0.3"  
    },  
    "autoload": {  
        "classmap": ["Model/"],  
        "files": ["config.php"]  
    }  
}
```

INSTALL PHP2WSDL

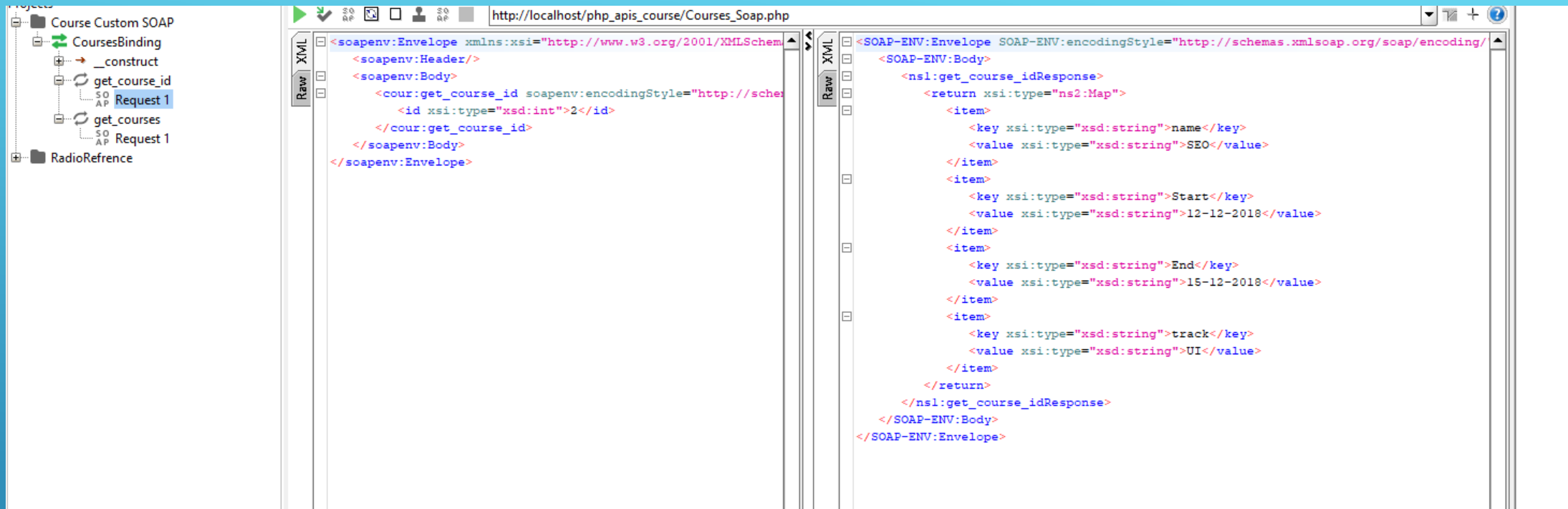
```
<?php
require("vendor/autoload.php");
$gen = new \PHP2WSDL\PHPClass2WSDL("Courses", "http://localhost/php_apis_cour
$gen->generateWSDL();
file_put_contents("wsdl_courses_final.xml", $gen->dump());
```

GENERATE WSDL

```
<?php
try{
require("vendor/autoload.php");
$server = new SoapServer("http://localhost/php_apis_course/wsdl_courses_final.xsd");
$server->setClass("Courses");
$server->handle();

}catch(SoapFault $e){
    var_dump($e);
}
```

THE SERVICE



TEST THE SERVICE BY SOAP UI