Chapter 4 - Data retrieval - Retrieving a customer

Chapter 4 - Data retrieval: Retrieving a customer

Goal: List and display information from a customer.
Difficulty: *
Problem: How to create a system that allows customers using IDs to retrieve customer records?

Preparation

Duplicate the file `list_the_customers.php` that was created in the previous chapter, rename it to `R-CRUD.php`, and put it at the root of your Web server.
If you didn't finish the previous chapter, duplicate the file `0-CustomersList.php` and rename it `R-CRUD.php`.

In the XML data that we retrieved, which contains the list of customers, we will find all the XLinks providing access to customer information.

```
<customers>
        <customer id="1" xlink:href="http://example.com/api/customers/1" />
</customers>
```

Example

The XLink for the `customer` tag with ID 1 is: `http://example.com/api/customers/1`

This link retrieves a new XML file, which contains information about the customer who has ID 1.

In order to manage access to different customers, we will associate page identifiers with customers via a `GET` parameter named "id". The link `http://example.com/R-CRUD.php?id=1` we will display the file for customer 1.

We must change the table in the file created in the previous chapter to add a link to future customer files.
We will have to isolate the display from the display list of a particular customer.

In order to do this, we must isolate the display of the list by checking that the "id" `GET` parameter property is not present when viewing your list. We do this using `isset()`.

Calling the web service is exactly the same as displaying the list, except that if you need to add the ID element to the table whose value is the id of a customer.

Currently, we are using the `customers` or `clients` resources. If we'd been trying to change the `countries` resources, this ID would have been a country ID.

```
$opt['resource'] = 'customers';
$opt['id'] = $_GET['id'];
$xml = $webService->get($opt);
```

✓ Use `isset()` before setting an ID enables you to easily carry out everything in this chapter.

Accessing resources is performed as above for displaying the list, because the tags that interest us are children of the `customers` tag.

```
$resources = $xml->customers->children();
```

This path can be created in another way (here in an HTML table):

```
foreach ($resources as $key => $resource)
        echo 'Name of field: ' . $key . ' - Value: ' . $resource . '<br />';
```

You now have everything needed to create a script to both list and display information for a particular customer.

Try creating this `R-CRUD.php` script. If you encounter any difficulties, follow the example from the `1-Retrieve.php` file, which contains the result to which you should get.

In another chapter, we will see how to filter, sort and limit the number of items displayed in the list. If you're in a hurry to implement these features, you can find more information in Chapter 8.