Processing XML

tutorial #tutorial07

James L. Parry
B.C. Institute of Technology

Tutorial Goals

This tutorial is meant to give you some practice working with SimpleXML to process an XML document, as described in lesson 9.

I have prepared a starter project for you to build on. It is a webapp, meant to process the XML data you built last week.

Revisions & Notes

- How can I test if a string ends in a specific substring? There is a built in function in PHP to do exactly that.
  `substr_compare($str, $test, strlen($str)-strlen($test), strlen($test)) === 0;`
- Cleaned up the .htaccess and tested that the starter project works as intended.

Background

Barker Bob, with his Burger Bar, wants to hold a bigger, better, burger bonanza, in honour of his uncle Billy Bob.

Last week, you devised a suitable DTD for constraining orders, and that resulted in some test data.

Your job this week is to process a selected order and print out a receipt for it,.

Preparation

I have prepared a starter project.

We are using the same menu as last week, and its data is inside the menu.xml file inside the data folder.

Fork the github project, and clone it locally to work with, the same as you have done with the previous tutorials.

Don't forget to add the special order you made last week, to the data folder!

The End Result

Your webapp is a two-page one: the home page should present a list of the orders found in the data folder, and the order page should print out the details for the selected order.

There shall be **no PHP in your views**. You are welcome to add view fragments as you deem fit.

There shall be **no SimpleXML in your controllers**. You are welcome to add models or model methods, as you deem fit.

What Needs Doing?

1. Present a basic list of orders
2. Present an order's basic receipt
3. Dress up the order list and receipts

1. PRESENT A BASIC LIST OF ORDERS

This should be in your Welcome controller:

- Use the directory helper to get a list of files
- Filter out the .xml ones
- For each, add an output row with the file name, linked to "welcome/order/{filename}", where the filename is substituted appropriately.
- Each order will have a designated customer name
- An order might have special instructions, eg for delivery

Sample output:

```
Barker Bob's Burger Bar - Orders

order1
order2
order3

Select an order from the list above to see its
receipt.
```

2. PRESENT A BASIC ORDER RECEIPT

This will be implemented in the `Welcome::order($num)` method.

You will need to reference the `Menu` model, which has been started for you.

You will need an `Order` model, using techniques similar to that in the Menu model.

Show any special instructions for a burger below the burger details, any special instructions for an order before any burger details.

Sample output:

```
Order1 for Jim (eatin)

*Burger #1*
Base: beef burger
Cheese: gruyere (top)
Toppings: seasonal mushrooms
Sauces: heinz ketchup, hellmans mayo
```

Menu Model

Your Menu model has provision for patties, with some suggested alternative strategies. You don't need all of them!

Add handling for:

- Cheeses
- Toppings
- Sauces

Order Model

You will need an Order model, with provision for accessing the parts of an order and returning them in a useful format for your controllers.

I suggest a parameter for the Order constructor: the filename of the XML document an order comes from.

You have decisions to make, and you may want to defer them until you get to the appropriate point in the receipt printing!

- Store order parts as codes or records?
- Deal with order totalling here or in the controller?
- Prepare view parameters here or in the controller?

Your Order Receipt

I suggest doing things one step at a time:

- Restaurant header
- Order header
- Burger header
- Burger base
- Cheeses - only show if there are some
- Toppings - either "none" or comma-separated list
- Sauces - either "none" or comma-separated list

3. DRESS THINGS UP

Add the customer's name to the order list on the homepage.

```
Barker Bob's Burger Bar - Orders

order1 (Jim)
order2 (George)
order3 (Insert your name here)

Select an order from the list above to see its
receipt.
```

Add pricing totals for each burger and for each order as a whole.

```
Order1 for Jim (eatin)

*Burger #1*
Base: beef burger
Cheese: gruyere (top)
Toppings: seasonal mushrooms
Sauces: heinz ketchup, hellmans mayo
Burger total: $x.xx

Order TOTAL: $x.xx
```

Congratulations!

You have completed tutorial #tutorial07: Processing XML

If you would take a minute to provide some feedback, we would appreciate it!

The next activity in sequence is: tutorial07b Creating XML

You can use your browser's back button to return to the page you were on before starting this activity, or you can jump directly to the course homepage, organizer, or reference page.