How to test and debug a PHP project

In particular, proj2

PHP with echo statements that trace the execution of the code

```php
// calculate the future value
$future_value = $investment;
echo '$future_value: ' . $future_value . '<br>';
echo '$interest_rate: ' . $interest_rate . '<br>';
echo '$years: ' . $years . '<br>';
echo 'For loop for calculating future value is starting...<br>';
for ($i = 1; $i <= $years; $i++) {
    $future_value = ($future_value + ($future_value * $interest_rate));
echo '$i: ' . $i . '<br>';
echo '$future_value: ' . $future_value . '<br>';
}
```

The data displayed in a browser

![Future Value Calculator](image)

When echo can't be used

- Echo is great for simple debugging of client-side code, in cases where the controller includes a view file.
- If the controller does a redirect, the echoed text is lost.
- Demo: using echo in pizza2's day controller, in change-to-next day code that needs work for proj2.
  - Put echo 'in list handling' in controller code that forwards to day_list.php, see echo in output.
  - Put echo 'in init-db handling' in code that redirects to, see instead "in list handling" in output.
  - This is because the redirected request does "list" handling. The original response is just the REDIRECT, no HTML at all.
  - Note that the next-day handling here does not redirect, unlike the pizza1 solution.
- Also, can't use echo in rest/index.php, because it will be sent back to client and mess up that response.

Losing echo output on redirect: fixes

- One workaround: use error_log instead of echo.
- Another way: temporarily replace REDIRECT with simple exit.
- Or recode a little and include day_list.php
- Demo: with redirect→exit, now we see just the echo output in init-db handling

Luckily, plain text works as HTML! (our browsers are very forgiving)

REDIRECT can also hide error displays

- Just like echo output, the orange box output for PHP errors is discarded on REDIRECT.
- Of course, the PHP error log report is there.
- So check the error log whenever something goes wrong.
Error logs

- Check the error log whenever something goes wrong.
- Proj2: pizza2's error log is in a new place, in pizza2/php-errors.log (both under XAMPP and on tocat) 
- Proj2/proj2_server's error log for web services is in proj2_server/php-server-errors.log 
- On tocat (Linux) or home Mac, need to "chmod 777 *.log" in both pizza2 and proj2_server to allow writes to the logs. 
  - (Not needed on Windows)

Getting errors to display on tocat

```php
<?php
echo 'setting display_errors on';
ini_set('display_errors',1); // This is done in main.php
echo $foo; // Undefined variable
echo 'Done';
```

Useful Exception methods

Example from Tutor PHP Exception Tutorial:

```php
// catch (Exception) {
//   echo "Message: " . $e->getMessage() . "\n";
//   echo "File: " . $e->getFile() . "\n";
//   echo "Line: " . $e->getLine() . "\n";
//   echo "Trace: " . $e->getTraceAsString() . "\n";
// }
```

Message: Email is invalid
File: C:\wamp\www\test\validator.php
Line: 7
Trace:
0 C:\wamp\www\test\validator.php(11): Validator->validate_email('51485649')
1 C:\wamp\www\test\test.php(12): User->save()
2 [main]

Exception vs. Errors in PHP

- We saw that PDO objects throw exceptions, themselves Exception objects 
  - In pizza2, we are using the Guzzle component, which also throws on errors

- Exceptions are fairly new to PHP, with PHP 5, so exceptions are different from errors, the older non-OO setup.
- An uncaught exception causes a fatal error. Thus we should catch possible exceptions 
- A caught exception causes no error.

New error.php in pizza2 (no more database_error.php)

Looks at object type of Exception for label, so can handle Guzzle, PDO ...

Error Configuration

- XAMPP as installed is set up for development, not production use, so displays most errors.
  - php.ini has error configuration for the server, so we could convert the server to production use by editing it.
  - From php.ini of XAMPP: `error_reporting = E_ALL & ~E_DEPRECATED & ~E_STRICT` 
  - display_errors = On

- From php.ini on tocat (`/etc/php5/php.ini`, protected): `error_reporting = E_ALL & ~E_DEPRECATED & ~E_STRICT` 
  - display_errors = Off

- We can override the error configuration by calls in our PHP program. For proj2, that is done in main.php.
Example of error.php output: expected error on rerun of restclient/index.php

Guzzle\Http\Exception\ServerException Error
Server error: POST http://localhost/proj2/proj2_server/rest/products/
resulted in a 500 Internal Server Error: response at line 115 in file
C:\xampp\htdocs\cs637\proj2\pizza2\vendor\guzzlehttp\guzzle\src\Exception
.php
full backtrace:
#0 C:\xampp\htdocs\cs637\proj2\pizza2\vendor\guzzlehttp\guzzle\src\Middleware
\Exception.php(45):
#1 C:\xampp\htdocs\cs637\proj2\pizza2\vendor\guzzle\guzzle\src\Request
\Client.php(33): Guzzle\Http\Promise\Promise\wait()
#10 C:\xampp\htdocs\cs637\proj2\pizza2\vendor\guzzle\guzzle\src\Request\Client.
\php(69): Guzzle\Http\Promise\Promise\wait
#11 (main)
#0 include() called at
C:\xampp\htdocs\cs637\proj2\pizza2\restclient\index.php on line 33

Inventory tracking by pizza2

- Pizza2 tracks flour and sugar inventory
  - Each pizza ordered by a student uses one unit of flour, one unit of sugar.
  - So you need a new inventory table.
- When the system sees supplies low, it orders more, using web services.
  - And saves the supply order ids, so it can later check for their delivery one or two days later (using a web service
  - So you need another table for saving supply-order ids.
- Specifically, the day manager checks inventory at the start of each day, even day 1, and
  - checks if any older orders have been delivered, and credits their quantities to the inventory
  - sends off web services for new orders as needed.

Code that needs work in pizza2: mostly in day manager

- pizza2/database/createdb.sql, droppedb.sql: needs new code to set up/drop inventory table, undelivered_orders table.
- pizza2/index.php: needs to fail an order if there is insufficient inventory. Deducts from inventory for successful order.
- pizza2/day/index.php: when doing next_day action, needs to do the inventory management and ordering. On any access, needs to find out inventory levels, etc. Calls into web_services.php to do needed web services to server to order more flour, cheese.
- pizza2/day/order_list.php: now needs to display inventory information and undelivered supply orders
- pizza2/day/web_services.php: functions for each web service, using Guzzle to do the actual GET and POSTs. As in model, let the caller handle exceptions.
- pizza2/model/inventory_db.php: database actions for inventory management (also need to edit initial.php)

If doing both client and server

- Which to do first?
  - I'd do the server-side first, because the provided proj2_tests allow for pretty good testing of the server
  - Then the client will have an intelligent server to talk to, complain about erroneous requests
  - But the client side can be coded with the help of the supplied server

If working alone on pizza2

- Or implementing pizza2 before the server, you need to be able to see the supply requests going out.
  - Note that the provided server already prints one line on each incoming request, in proj2_server/php-server-errors.php:
    ```php
    starting Rest server request, method=POST, url = .../rest/products/
    This output is coming from line 32 of rest/index.php
    error_log('starting Rest server request, method=', $method .
    ' url = ...', $project_url);
    You can add to this, to print the body (in JSON) too:
    error_log('body: ', file_get_contents('php://input'));
    Or use jen, decode it to see it in PHP arrays.
  - Note if using Mac or Linux, you may need to chmod 777 */* to make the logs writable, and if they aren't there, first reconfigure them using touch php-
    errors.log in pizza2 and touch php-server-errors.log in proj2_server

Working on pizza2 without a finished server

- You have some experience with cooking up fake data ("mocks") from hw5
- Here you can temporarily code get_server_supply_orders() (or whatever you call it) to return a hard-coded array of supply orders for your inventory logic to work on.
- It could even use the day number to produce an appropriate sequence of orders
Code that needs work in proj2_server:
mainly in rest/index.php

- proj2_server/rest/index.php: top-level web service code: need to fix GET/POST /day code, add /orders services
- proj2_server/rest/.htaccess: needed to get the Apache web server to execute index.php for any incoming URL .../rest/... (no edits needed)
- proj2_server/model/order_db.php: provided code has new add_order(), this may be sufficient.
- proj2_server/model/day.php: needs to retrieve and update the system day on the server

Command-line curl

- We have seen PHP's libcurl in action
- Separately, we can use curl at the command line in Windows or Linux/Mac
- Download curl for Windows at http://curl.haxx.se/download.html
  (this is tricky, needs DLLs from Microsoft to work)
- Linux/Mac: should have curl already
- Also see tutorial there: http://curl.haxx.se/docs/httpscripting.html

Command-line curl example 1

From pa2.html:
curl localhost/cs637/username/proj2/proj2_server/rest/day/
This fires a GET to http://localhost/cs637 ...
i.e. does the Web service to get the current day from the server:

```
topcat$ curl localhost/cs637/username/proj2/proj2_server/rest/day/
topcat$
```

Result (no end-of-line after the number, so shows up at start of next line)

Note: will always return 6 until you fix its code.

Command-line curl example 2

From pa2.html:
curl -i -d 9 -H Content-Type:text/plain http://localhost/cs637/username/proj2/proj2_server/rest/day/
This fires a POST to http://localhost/cs637 ...
i.e. does the Web service to set the current day to 9 in the server, and overrides the default Content-Type (URL-encoded, as needed for params with POST)

Without -i or -v for status info, or verbose status info:
topcat$ curl -d 9 -H Content-Type:text/plain http://localhost/cs637/username/proj2/proj2_server/rest/day/
topcat$

Nothing at all seen—how can we tell it worked?

Command-line curl example 2

With -v for verbose:
topcat$ curl -v -d 9 -H Content-Type:text/plain http://localhost/cs637/username/proj2/proj2_server/rest/day/
```
* Hostname was NOT found in DNS cache
*   Trying 127.0.0.1...
* Connected to localhost [127.0.0.1] port 80 (#0)
> POST /cs637/username/proj2/proj2_server/rest/day/ HTTP/1.1
> User-Agent: curl/7.35.0
> Host: localhost
> Accept: */*
> Content-Type:text/plain
> Content-Length: 1
>
* upload completely sent off: 1 out of 1 bytes
< HTTP/1.1 200 OK
< Date: Mon, 04 May 2015 14:33:16 GMT
< Server: Apache/2.4.7 (Ubuntu) is not blacklisted
< X-Powered-By: PHP/5.5.9-1ubuntu4.9
< Content-Type: text/html
<
```

With -i for status info: less clutter, get the basic facts:
curl -i -d 9 -H Content-Type:text/plain http://localhost/cs637/username/proj2/proj2_server/rest/day/
```
HTTP/1.1 200 OK
Date: Sat, 21 Nov 2015 19:54:43 GMT
Server: Apache/2.4.9 (Win32) OpenSSL/1.0.1g PHP/5.5.11
X-Powered-By: PHP/5.5.11
Content-Length: 0

Content-Type: text/html
Command-line curl example 2

```bash
topcat$ curl -i -d 9 -H 'Content-Type:text/plain' http://localhost/cs637/eoneil/proj2/proj2_server/rest/day/
```

Explanation of arguments:
- `-i` return status info
- `-d` data, use method POST, with POST data "9". Defaults to Content-Type for encoded parameters, like form submission fields x=10&y=20 or whatever
- `-H Content-Type:text/plain` override the default Content-Type to this type, plain text

So this command does a POST to the URL with just "9" as POST data, and reports verbosely on the action

Shell scripts

We can automate command line work with shell scripts (even on Windows)

```bash
topcat$ more test1.sh
```

```bash
curl -i -d 9 -H 'Content-Type:text/plain' http://localhost/cs637/$1/proj2/proj2_server/rest/day/
```

Files in `eoneil` for `$1` in script:
- `chmod +x test1.sh`
- `test1.sh eoneil`
- `test1.sh`

For Windows: test1.cmd: use `%1%` instead of `$1`. See shell and .cmd files in proj2_tests directory.

proj2_tests

- Note: these test the server-side code only
  - serveretest0: tests provided functionality of proj2_server
  - serveretest1: test that server day number can be set, setting of day 0 reinitializes orders
  - serveretest2: sets day 0 to reinit orders, then sends an order in, gets the order back as order 1, or possibly a higher number on reruns
  - serveretest3: like serveretest2, then set server day to 3, get order back, should be delivered

serveretest0.sh run on topcat

```bash
topcat$ serveretest0.sh eoneil
```

```
6
```

GOAL: this should be 9

```
6
```

Grading run

- Run serveretest series
- Reinitialize: use dropdb, createdb of pizza2/database, send day 0 to proj2_server
- Run Selenium script:
  - Test0: initialize, see initial supply order for 60 units of flour, 80 units of cheese (or a little more)
  - Test1: order 10 pizzas and check inventory
  - Test2: order too many pizzas, check for error
  - Test3: reinit, advance day and check for supply order (in case project doesn't do initialization inventory check, see order at start of 2nd day)
  - Test4: check a supply order is delivered
- Possibly other tests…