# Another One Bytes the Dust

•••

Members: Josh, Clae, and Alex

# Project Selections




sigmah

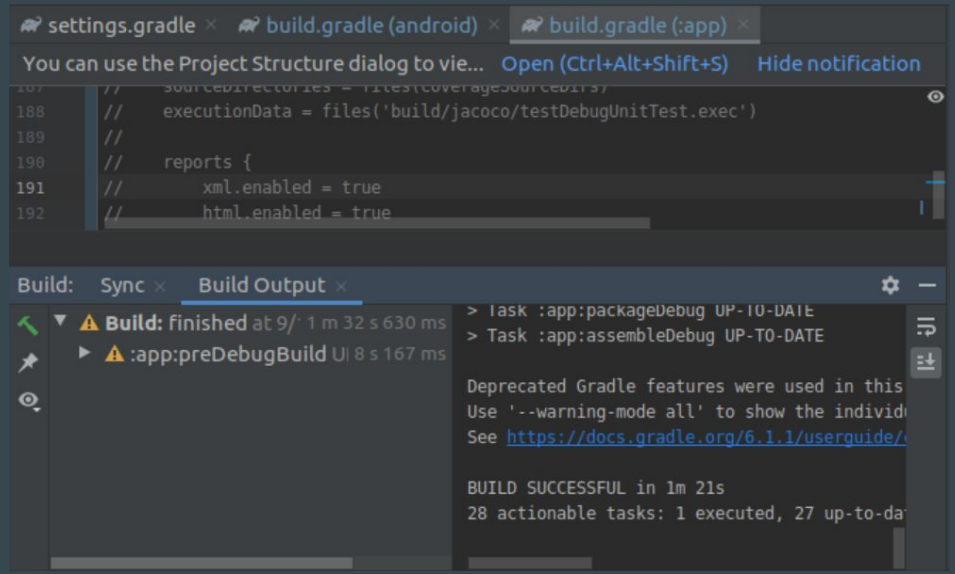
# Why not Sigmah?

```
BUILD FAILURE
------------------------------------------------------------------------
Total time: 2.591 s
Finished at: 2017-03-01T13:09:47+05:30
Final Memory: 21M/347M
------------------------------------------------------------------------
 Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:3.1
 /Users/vshukla/git/Prism/src/main/java/main/MainUITabbed.java:[26,22] pac
 /Users/vshukla/git/Prism/src/main/java/main/MainUITabbed.java:[93,41] can
 symbol: class Application
 /Users/vshukla/git/Prism/src/main/java/main/MainUITabbed.java:[93,67] can
 symbol: variable Application
 -> [Help 1]

 To see the full stack trace of the errors, re-run Maven with the -e switc
 Re-run Maven using the -X switch to enable full debug logging.

 For more information about the errors and possible solutions, please read
 [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MojoFailureExce
```


Glucosio

# Why not Glucosio?

Glucosio failed in the mobile app build due to one of the Jocoso dependencies no longer being supported and when searching for a solution we came up with no fix.




moodle
TM

# What is Moodle?

MOODLE: Modular Object-Oriented Dynamic Learning Environment

Wiki: Moodle is a free and open-source learning management system written in PHP and distributed under the GNU general public license.

Original Release Date: August 2002

# Why Moodle?

- Got it to build successfully
- Good documentation
- Many examples of others contributing
- Seemed similar to Oaks, thus familiarity with the system
- Clear testability with test folders, and PHPUnit setup
- Build in PHP, useful to learn
- Other team had success
- Hindsight: would have branch outside of H/FOSS

# Testing Plan

# Testing Process

- Form testing template
- Gather requirements
- Find desired methods of classes to test
- Form test cases for each method
- Build drivers to feed input into method for output
- Build script to automate drivers
- Collect output from drivers in script
- Compare outputs to expected outputs
- Display HTML / Write report

# Testing Template


```
"testId": "TC01",
"designedBy": "Clae, Josh & Alex",
"testDesignedDate": "10/04/2020",
"classTested": "type_base",
"methodTested": "convert_to_timestamp",
"testingInputs": "2017, 02, 11, 17, 34",
"Expected Output": "1486834460",
"success": "pass"
```

# Test Cases

# LW01-LW05.json

```
{
        "testId": "LW01",
        "requirement": "Method breaks up a string into multiple parts
        "driver": "breakUpLongWordsDriver.php",
        "classTested": "weblib",
        "methodTested": "break_up_long_words",
        "testingInputs": "arnoldmcdoyl@gmail.com*5*@",
        "expectedOutput": "arnol@dmcdoy@l@gmai@l.com"
}
```

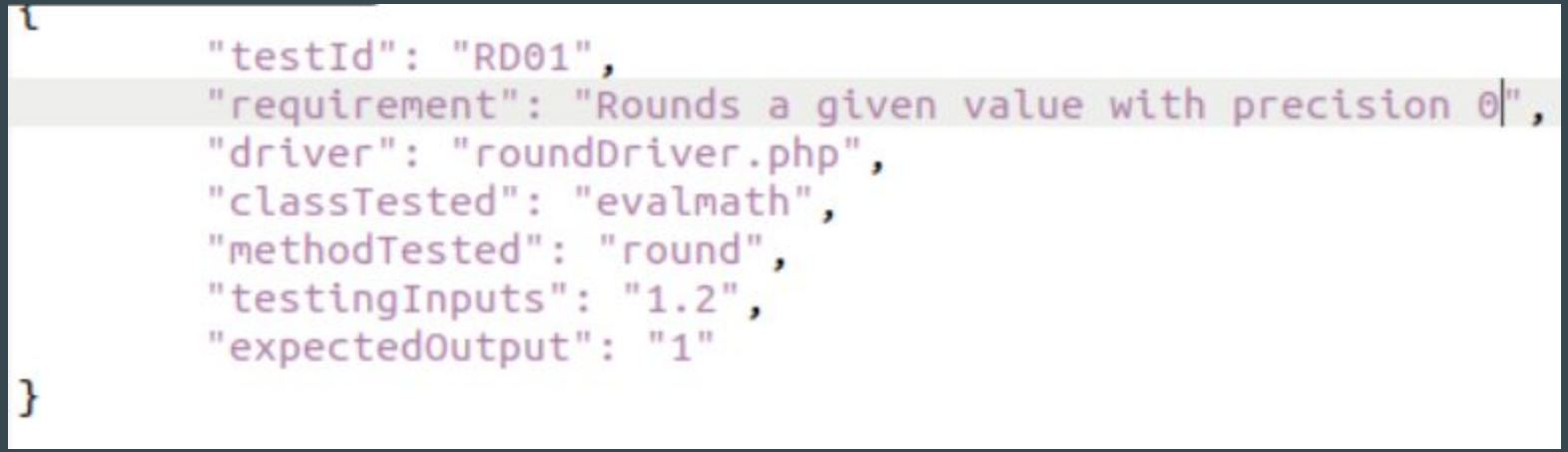
# NV01-NV05.json

```
{
        "testId": "NV01",
        "requirement": "Normalize a string to be a series of numbers",
        "driver": "normalizeDriver.php",
        "classTested": "environment_lib",
        "methodTested": "normalize_version",
        "testingInputs": "...135...",
        "expectedOutput": "135"
}
```

# PC01-PC05.json


```
{
        "testId": "PC01",
        "requirement": "Method parses and returns charset so all characters are lower-case",
        "driver": "parseCharsetDriver.php",
        "classTested": "core_text",
        "methodTested": "parse_charset",
        "testingInputs": "UTF-8",
        "expectedOutput": "utf-8"
}
```

# RD01-RD05.json



```
{
        "testId": "RD01",
        "requirement": "Rounds a given value with precision 0",
        "driver": "roundDriver.php",
        "classTested": "evalmath",
        "methodTested": "round",
        "testingInputs": "1.2",
        "expectedOutput": "1"
}
```

# TT01-TT05.json


```
{
        "testId": "TT01",
        "requirement": "Removes TRUSTTEXT label from string",
        "driver": "trustTextDriver.php",
        "classTested": "weblib",
        "methodTested": "trusttext_strip",
        "testingInputs": "hello#####TRUSTTEXT#####world",
        "expectedOutput": "helloworld"
}
```

# Testing Methods

- Class: weblib
  - Method: “trusttext_strip”
  - Method: “break_up_long_words”
- Class: evalmath
  - Method: “round”
- Class: core_text
  - Method: “parse_charset”
- Class: environment_lib
  - Method: “normalize_version”

# normalize_version

```php
/**
 * This function will normalize any version to just a serie of numbers
 * separated by dots. Everything else will be removed.
 *
 * @param string $version the original version
 * @return string the normalized version
 */
function normalize_version($version) {

/// 1.9 Beta 2 should be read 1.9 on enviromental checks, not 1.9.2
/// we can discard everything after the first space
    $version = trim($version);
    $versionarr = explode(" ",$version);
    if (!empty($versionarr)) {
        $version = $versionarr[0];
    }
/// Replace everything but numbers and dots by dots
    $version = preg_replace('/[^\.\d]/', '.', $version);
/// Combine multiple dots in one
/// [THIS IS AN INJECTION WHERE WE REMOVE THE LINE THAT REMOVES MULTIPLE DOTS]
    //$version = preg_replace('/(\.{2,})/', '.', $version);
/// Trim possible leading and trailing dots
    $version = trim($version, '.');

    return $version;
}
```

# round

```
597     static function round($val, $precision = 0) {
598         // return round($val, $precision);
599         /// [THIS IS AN INJECTION WHERE IF VAL IS LESS THAN 0, ERROR MESSAGE RETURNED]
600         if ($val > 0){
601             return round($val, $precision);
602         } else {
603             return "You have entered a number less than 0..";
604         }
605     }
```

# trusttext_strip

```
1620  /**
1621   * Legacy function, used for cleaning of old forum and glossary text only.
1622   *
1623   * @param string $text text that may contain legacy TRUSTTEXT marker
1624   * @return string text without legacy TRUSTTEXT marker
1625   */
1626  function trusttext_strip($text) {
1627      if (!is_string($text)) {
1628          // This avoids the potential for an endless loop below.
1629          throw new coding_exception('trusttext_strip parameter must be a string');
1630      }
1631      while (true) { // Removing nested TRUSTTEXT.
1632          $orig = $text;
1633          /// [THIS IS AN INJECTION WILL NOW HANDLE TRUSTEXT AS VALID TRUSTTEXT]
1634          $text = str_replace('#####TRUSTEXT#####', '#####TRUSTTEXT#####', $text);
1635          $text = str_replace('#####TRUSTTEXT#####', '', $text);
1636          if (strcmp($orig, $text) === 0) {
1637              return $text;
1638          }
1639      }
1640  }
```

# break_up_long_words

```
/**
 * Given some normal text this function will break up any
 * long words to a given size by inserting the given character
 *
 * It's multibyte savvy and doesn't change anything inside html tags.
 *
 * @param string $string the string to be modified
 * @param int $maxsize maximum length of the string to be returned
 * @param string $cutchar the string used to represent word breaks
 * @return string
 */
function break_up_long_words($string, $maxsize=20, $cutchar=' ') {

    // First of all, save all the tags inside the text to skip them.
    $tags = array();
    filter_save_tags($string, $tags);

    // Process the string adding the cut when necessary.
    $output = '';
    $length = core_text::strlen($string);
    $wordlength = 0;

    for ($i=0; $i<$length; $i++) {
        $char = core_text::substr($string, $i, 1);
        if ($char == 'A'){
            continue;
        }
        if ($char == ' ' or $char == "\t" or $char == "\n" or $char == "\r" or $char == "<" or $char == ">") {
            $wordlength = 0;
        } else {
            $wordlength++;
            if ($wordlength > $maxsize) {
                $output .= $cutchar;
                $wordlength = 0;
            }
        }
        $output .= $char;
    }
```

# parse_charset

```
124         * Please note it does not mean the returned charset is actually supported.
125         *
126         * @static
127         * @param string $charset raw charset name
128         * @return string normalised lowercase charset name
129         */
130        public static function parse_charset($charset) {
131            $charset = strtolower($charset);
132
133            // shortcuts so that we do not have to load typo3 on every page
134
135            if ($charset === 'utf8' or $charset === 'utf-8') {
136                //return 'utf-8';
137                /// [THIS IS AN INJECTION WHERE SET THE RETURN IF UTF8 to UTF7]
138                return 'utf-7';
139            }
140
141            if (preg_match('/^(cp|win|windows)-?(12[0-9]{2})$/', $charset, $matches)) {
142                return 'windows-'.$matches[2];
143            }
144
145            if (preg_match('/^iso-8859-[0-9]+$/', $charset, $matches)) {
146
147                return $charset;
148            }
149
150            if ($charset === 'euc-jp') {
151                return 'euc-jp';
152            }
153            if ($charset === 'iso-2022-jp') {
154                return 'iso-2022-jp';
155            }
156            if ($charset === 'shift-jis' or $charset === 'shift_jis') {
157                return 'shift_jis';
158            }
159            if ($charset === 'gb2312') {
160                return 'gb2312';
161            }
162            if ($charset === 'gb18030') {
163                return 'gb18030';
164            }
```

# Starting the Testing Suite

Naive Approach:

- Driver: reads in test cases, produces a big string to send to script
- Script: runs every driver
- To Run: move into `/scripts` and run `./runAllTests.py`

# Naive Approach

# Driver (Naive)

```php
class driverObject {
        var $tcID;
        var $requirement;
        var $driver;
        var $class;
        var $method;
        var $testingInput;
        var $expectedOutput;
        var $success;

        function settcID($id){
                $this->tcID = $id;
        }
        function gettcID(){
                echo $this->tcID."$$$";
        }
        function setRequirement($id){
```

```php
<?php
//error_reporting(0);
//TEST STRING TO LOWER & REGULAR EXPRESSIONS

define('CLI_SCRIPT', true);
require("driverObject.php");
require_once("../project/moodle1/config.php");
require("../project/moodle1/user/lib.php");
require("../project/moodle1/lib/classes/text.php");

// Create Object of our Selected "Core Text" Class
$text = new core_text;

// Decode Json Objects from Test Case Folder
$tc01 = json_decode(file_get_contents("../testCases/TC01.json"));
$tc02 = json_decode(file_get_contents("../testCases/TC02.json"));
$tc03 = json_decode(file_get_contents("../testCases/TC03.json"));
$tc04 = json_decode(file_get_contents("../testCases/TC04.json"));
$tc05 = json_decode(file_get_contents("../testCases/TC05.json"));
$obj = new driverObject;

// Fill Array with Empty Objects for Test Case Info
$finalArr = array($obj, $obj, $obj, $obj, $obj);

// Fill Array of Test Case Decode Json
$testCaseArr = array($tc01, $tc02, $tc03, $tc04, $tc05);
```

```php
for ($i = 0; $i < count($finalArr); $i++){

        $input = $testCaseArr[$i]->testingInputs;
        $expectedOut = $testCaseArr[$i]->expectedOutput;

        //Establish Object
        $finalArr[$i]->settcID($testCaseArr[$i]->testId);
        $finalArr[$i]->setRequirement($testCaseArr[$i]->requirement);
        $finalArr[$i]->setDriver($testCaseArr[$i]->driver);
        $finalArr[$i]->setClass($t
<p>Requirements: %s</p>estCaseArr[$i]->classTested);
        $finalArr[$i]->setMethod($testCaseArr[$i]->methodTested);
        $finalArr[$i]->setTestingInput($testCaseArr[$i]->testingInputs);
        $finalArr[$i]->setExpectedOutput($testCaseArr[$i]->expectedOutput);


        $parsed_charset = $text->parse_charset($input);

        if ($parsed_charset == $expectedOut){
                $finalArr[$i]->setSuccess("PASS");
        } else{
                $finalArr[$i]->setSuccess("FAIL");
        }

        $finalArr[$i]->gettcID();
        $finalArr[$i]->getRequirement();
        $finalArr[$i]->getDriver();
        $finalArr[$i]->getClass();
        $finalArr[$i]->getMethod();
        $finalArr[$i]->getTestingInput();
        echo $parsed_charset."$$$";
        $finalArr[$i]->getExpectedOutput();
        $finalArr[$i]->getSuccess();
}
```

# Produced String

```
CasesExecutables$ php TC01_05Driver.php
TC01$$$Method parses and returns charset so all characters are lower-case$$$TC01
_05Driver.php$$$core_text$$$parse_charset$$$UTF-8$$$utf-8$$$utf-8$$$PASS***TC02$
$$Method parses and returns charset so all characters are lower-case$$$TC01_05Dr
iver.php$$$core_text$$$parse_charset$$$IsO-2022-jP$$$iso-2022-jp$$$iso-2022-jp$$
$PASS***TC03$$$Method parses and returns charset so all characters are lower-cas
e$$$TC01_05Driver.php$$$core_text$$$parse_charset$$$utl-10$$$utl-10$$$utl-10$$$P
ASS***TC04$$$Method parses and returns charset so all characters are lower-case$
$$TC01_05Driver.php$$$core_text$$$parse_charset$$$utf8$$$utf-8$$$utf-8$$$PASS***
TC05$$$Method parses and returns charset so all characters are lower-case$$$TC01
_05Driver.php$$$core_text$$$parse_charset$$$shift-jis$$$shift_jis$$$shift_jis$$$
PASS***josh@josh-VirtualBox:~/Desktop/SE/Another-One-Bytes-the-Dust/TestAutomati
```

# Script (Naive)

```
#! /usr/bin/python3.8

import sys
import subprocess
import webbrowser
import time

result = subprocess.run(
        ['php', '../testCasesExecutables/TC01_05Driver.php'],
        stdout=subprocess.PIPE,
        check=True
)

htmlOpening = '''
<!DOCTYPE html>
<head>
<style>
.header{
        text-align: center;
        font-size: 100px;
        font-family: Arial, Helvetica, sans-serif;

}
.container{
        width: 90%;
        margin-left: auto;
        margin-right: auto;
        height: 380px;
        background-color: #f2f2f2;

</style>
<meta charset="utf-8">
<title>Test Report</title> </head>
<body>
<h1 class="header">AOBTD TESTING FRAMEWORK</h1>
f = open("testReport.html", "w")
f.write(htmlOpening)


x = str(result.stdout)
y = x.split('***')
#print(y)
outstring = ""
for case in y:
        attr = case.split('$$$')
        if (len(attr) == 9):
                whole = innerText % (attr[0].replace("b'", ""), attr[1], attr[2], attr[3], attr[4], attr[5], attr[6], attr[7], attr[8])
                f.write(whole)

f.write(htmlClosing)
f.close()
webbrowser.open("testReport.html")
```

Text Editor

# Refined Approach

- Driver: Performs simple task, takes input, runs method, produces output
- Script: Read in test cases, match to driver, run all subprocess, produce html
- To Run: Using simple bash script, run `./scripts/runAllTests.py` from root

# Driver (Refined)

```php
<?php
//error_reporting(0);
//TEST STRING TO LOWER & REGULAR EXPRESSIONS

define('CLI_SCRIPT', true);
require_once("../project/moodle1/config.php");
require("../project/moodle1/user/lib.php");

if (!$argv[1]) {
        throw new Exception('Inputs are invalid...');
}
try {
        $splitStrings = explode("*", $argv[1]);
        $output = break_up_long_words($splitStrings[0], intval($splitStrings[1]), $splitStrings[2]);
        print_r($output);
} catch (Exception $e) {
        print_r('Caught exception: '.$e->getMessage()."\n");
}

?>
```

# Script (Refined)

```
import webbrowser
import time
import glob
import functools
from runTestsFunctionsAndVars import *

# read in the test cases
arrayOfCases = (glob.glob("../testCases/*.json"));

# globals
cases = []
outputs = []
driversUsed = []
sortedCases = []

# run subprocess to drivers to get outputs
getOutputsFromDrivers(arrayOfCases, cases, driversUsed, outputs)

#add outputs as attribute to cases
addOutputsAsAttr(cases, outputs)

#sort cases according to drivers
sortDrivers(driversUsed, cases, sortedCases)

# write html opening to html report
f = open("../temp/testReport.html", "w+")
f.write(htmlOpening)

# add cases in html format to the html report
addCaseToHTML(sortedCases, f)

# add html closing to html report
f.write(htmlClosing)
f.close()

#open temp file in browser
webbrowser.open("../temp/testReport.html")
```

# Script Methods

```
def addOutputsAsAttr(cases, outputs):
        throughOutputs = 0;
        for case in cases:
                case['actualOutput'] = outputs[throughOutputs]
                throughOutputs += 1

def compareExp(actualOutput, expectedOutput):
        if (actualOutput == expectedOutput):
                return "Pass"
        else:
                return "Fail"

def getOutputsFromDrivers(arrayOfCases, cases, driversUsed, outputs):
        for case in arrayOfCases:
                f = open(case);
                data = json.load(f)
                cases.append(data)
                testingInput = ""
                driverName = ""
                for key, val in data.items():
                        if (key == "driver"):
                                driverName = str(val)
                                if driverName not in driversUsed:
                                        driversUsed.append(driverName)
                        if (key == "testingInputs"):
                                testingInput = str(val)
                driverName = '../testCasesExecutables/' + driverName
                result = subprocess.run(
                ['php', driverName, testingInput],
                stdout=subprocess.PIPE,
                check=True)
                outputs.append((str(result.stdout).replace("b'", ""))[:-1])
```

# Revelations

- Driver reading test cases is overcomplicated and doing too much (BAD)
- Every driver will have to read through all test cases (BAD)
- Hard-coded test case read in doesn’t account for new cases (BAD)

- Driver doing one simple task, take input, create output (GOOD)
- Script reading in test cases dynamically (GOOD)

# Running from Root

./scripts/runAllTests.sh

```
#!/usr/bin/bash

cd scripts
./runTests.py
```

# AOBTD TESTING FRAMEWORK

| Test ID | Requirements | Driver | Class Tested | Method Tested | Testing Input | Actual Output | Expected Output | Pass/Fail |
|---|---|---|---|---|---|---|---|---|
| NV01 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | ...135... | 135 | 135 | Pass |
| NV02 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | 1.2.3.4... | 1.2.3.4 | 1.2.3.4 | Pass |
| NV03 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | 1..2.3 | 1..2.3 | 1.2.3 | Fail |
| NV04 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | ..hello000world... | 000 | 000 | Pass |
| NV05 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | hello..2 | 2 | 2 | Pass |
| PC01 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | UTF-8 | utf-7 | utf-8 | Fail |
| PC02 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | IsO-2022-jP | iso-2022-jp | iso-2022-jp | Pass |
| PC03 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | cb-180 | cb-180 | cb-180 | Pass |
| PC04 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | utf8 | utf-7 | utf-8 | Fail |
| PC05 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | shift-jis | shift_jis | shift_jis | Pass |
| RD01 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | 1.2 | 1 | 1 | Pass |
| RD02 | Replaces characters in the string with HTML entity equivs | roundDriver.php | evalmath | round | 5.234 | 5 | 5 | Pass |
| RD03 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | -45.0 | You have entered a number less than 0.. | -45 | Fail |
| RD04 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | 743.01 | 743 | 743 | Pass |
| RD05 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | .001 | 0 | 0 | Pass |
| LW01 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | arnoldmcdoyl@gmail.com*5*@ | arnol@dmcdoy@l+AEA-gmai@l.com | arnol@dmcdoy@l@gmai@l.com | Fail |
| LW02 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | helloworldhelloworld*10*$$$ | helloworld$$$helloworld | helloworld$$$helloworld | Pass |
| LW03 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | anotherbytesdust*-1*% | %a%n%o%t%h%e%r%b%y%t%e%s%d%u%s%t | %a%n%o%t%h%e%r%b%y%t%e%s%d%u%s%t | Pass |
| LW04 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | simplestring*100*( | simplestring | simplestring | Pass |
| LW05 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | ABCDEFGHIJKL*3*@ | BCD@EFGH@IJKL | ABC@DEFG@HIJK@L | Fail |
| TT01 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | hello#####TRUSTTEXT#####world | helloworld | helloworld | Pass |
| TT02 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | #####TRUSTTEXT#####hey#####TRUSTTEXT##### | hey | hey | Pass |
| TT02 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | one####TRUSTTEXT####less | one####TRUSTTEXT####less | one####TRUSTTEXT####less | Pass |
| TT03 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | #####TRUSTEXT##### | | #####TRUSTEXT##### | Fail |
| TT05 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | ###one##TRUSTTEXT##### | ###one##TRUSTTEXT##### | ###one##TRUSTTEXT##### | Pass |

# Error Injection

Next, we break down the changes we made to the Moodle code to inject our errors and the results of running our testing suite with the altered code.

# normalize_version

```php
/**
 * This function will normalize any version to just a serie of numbers
 * separated by dots. Everything else will be removed.
 *
 * @param string $version the original version
 * @return string the normalized version
 */
function normalize_version($version) {

/// 1.9 Beta 2 should be read 1.9 on enviromental checks, not 1.9.2
/// we can discard everything after the first space
    $version = trim($version);
    $versionarr = explode(" ",$version);
    if (!empty($versionarr)) {
        $version = $versionarr[0];
    }
/// Replace everything but numbers and dots by dots
    $version = preg_replace('/[^\.\d]/', '.', $version);
/// Combine multiple dots in one
/// [THIS IS AN INJECTION WHERE WE REMOVE THE LINE THAT REMOVES MULTIPLE DOTS]
    //$version = preg_replace('/(\.{2,})/', '.', $version);
/// Trim possible leading and trailing dots
    $version = trim($version, '.');

    return $version;
}
```

# round

```
597     static function round($val, $precision = 0) {
598         // return round($val, $precision);
599         /// [THIS IS AN INJECTION WHERE IF VAL IS LESS THAN 0, ERROR MESSAGE RETURNED]
600         if ($val > 0){
601             return round($val, $precision);
602         } else {
603             return "You have entered a number less than 0..";
604         }
605     }
```

# trusttext_strip

```
1620  /**
1621   * Legacy function, used for cleaning of old forum and glossary text only.
1622   *
1623   * @param string $text text that may contain legacy TRUSTTEXT marker
1624   * @return string text without legacy TRUSTTEXT marker
1625   */
1626  function trusttext_strip($text) {
1627      if (!is_string($text)) {
1628          // This avoids the potential for an endless loop below.
1629          throw new coding_exception('trusttext_strip parameter must be a string');
1630      }
1631      while (true) { // Removing nested TRUSTTEXT.
1632          $orig = $text;
1633          /// [THIS IS AN INJECTION WILL NOW HANDLE TRUSTEXT AS VALID TRUSTTEXT]
1634          $text = str_replace('#####TRUSTEXT#####', '#####TRUSTTEXT#####', $text);
1635          $text = str_replace('#####TRUSTTEXT#####', '', $text);
1636          if (strcmp($orig, $text) === 0) {
1637              return $text;
1638          }
1639      }
1640  }
```

# break_up_long_words

```
/**
 * Given some normal text this function will break up any
 * long words to a given size by inserting the given character
 *
 * It's multibyte savvy and doesn't change anything inside html tags.
 *
 * @param string $string the string to be modified
 * @param int $maxsize maximum length of the string to be returned
 * @param string $cutchar the string used to represent word breaks
 * @return string
 */
function break_up_long_words($string, $maxsize=20, $cutchar=' ') {

    // First of all, save all the tags inside the text to skip them.
    $tags = array();
    filter_save_tags($string, $tags);

    // Process the string adding the cut when necessary.
    $output = '';
    $length = core_text::strlen($string);
    $wordlength = 0;

    for ($i=0; $i<$length; $i++) {
        $char = core_text::substr($string, $i, 1);
        if ($char == 'A'){
            continue;
        }
        if ($char == ' ' or $char == "\t" or $char == "\n" or $char == "\r" or $char == "<" or $char == ">") {
            $wordlength = 0;
        } else {
            $wordlength++;
            if ($wordlength > $maxsize) {
                $output .= $cutchar;
                $wordlength = 0;
            }
        }
        $output .= $char;
    }
```

# parse_charset

```
124         * Please note it does not mean the returned charset is actually supported.
125         *
126         * @static
127         * @param string $charset raw charset name
128         * @return string normalised lowercase charset name
129         */
130        public static function parse_charset($charset) {
131            $charset = strtolower($charset);
132
133            // shortcuts so that we do not have to load typo3 on every page
134
135            if ($charset === 'utf8' or $charset === 'utf-8') {
136                //return 'utf-8';
137                /// [THIS IS AN INJECTION WHERE SET THE RETURN IF UTF8 to UTF7]
138                return 'utf-7';
139            }
140
141            if (preg_match('/^(cp|win|windows)-?(12[0-9]{2})$/', $charset, $matches)) {
142                return 'windows-'.$matches[2];
143            }
144
145            if (preg_match('/^iso-8859-[0-9]+$/', $charset, $matches)) {
146
147                return $charset;
148            }
149
150            if ($charset === 'euc-jp') {
151                return 'euc-jp';
152            }
153            if ($charset === 'iso-2022-jp') {
154                return 'iso-2022-jp';
155            }
156            if ($charset === 'shift-jis' or $charset === 'shift_jis') {
157                return 'shift_jis';
158            }
159            if ($charset === 'gb2312') {
160                return 'gb2312';
161            }
162            if ($charset === 'gb18030') {
163                return 'gb18030';
164            }
```

# AOBTD TESTING FRAMEWORK

| Test ID | Requirements | Driver | Class Tested | Method Tested | Testing Input | Actual Output | Expected Output | Pass/Fail |
|---|---|---|---|---|---|---|---|---|
| NV01 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | ...135... | 135 | 135 | Pass |
| NV02 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | 1.2.3.4... | 1.2.3.4 | 1.2.3.4 | Pass |
| NV03 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | 1..2.3 | 1..2.3 | 1.2.3 | Fail |
| NV04 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | ..hello000world... | 000 | 000 | Pass |
| NV05 | Normalize a string to be a series of numbers | normalizeDriver.php | environment_lib | normalize_version | hello..2 | 2 | 2 | Pass |
| PC01 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | UTF-8 | utf-7 | utf-8 | Fail |
| PC02 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | IsO-2022-jP | iso-2022-jp | iso-2022-jp | Pass |
| PC03 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | cb-180 | cb-180 | cb-180 | Pass |
| PC04 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | utf8 | utf-7 | utf-8 | Fail |
| PC05 | Method parses and returns charset so all characters are lower-case | parseCharsetDriver.php | core_text | parse_charset | shift-jis | shift_jis | shift_jis | Pass |
| RD01 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | 1.2 | 1 | 1 | Pass |
| RD02 | Replaces characters in the string with HTML entity equivs | roundDriver.php | evalmath | round | 5.234 | 5 | 5 | Pass |
| RD03 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | -45.0 | You have entered a number less than 0.. | -45 | Fail |
| RD04 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | 743.01 | 743 | 743 | Pass |
| RD05 | Rounds a given value with precision 0 | roundDriver.php | evalmath | round | .001 | 0 | 0 | Pass |
| LW01 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | arnoldmcdoyl@gmail.com*5*@ | arnol@dmcdoy@l+AEA-gmai@l.com | arnol@dmcdoy@l@gmai@l.com | Fail |
| LW02 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | helloworldhelloworld*10*$$$ | helloworld$$$helloworld | helloworld$$$helloworld | Pass |
| LW03 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | anotherbytesdust*-1*% | %a%n%o%t%h%e%r%b%y%t%e%s%d%u%s%t | %a%n%o%t%h%e%r%b%y%t%e%s%d%u%s%t | Pass |
| LW04 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | simplestring*100*( | simplestring | simplestring | Pass |
| LW05 | Method breaks up a string into multiple parts with a symbol separator | breakUpLongWordsDriver.php | weblib | break_up_long_words | ABCDEFGHIJKL*3*@ | BCD@EFGH@IJKL | ABC@DEFG@HIJK@L | Fail |
| TT01 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | hello#####TRUSTTEXT#####world | helloworld | helloworld | Pass |
| TT02 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | #####TRUSTTEXT#####hey#####TRUSTTEXT##### | hey | hey | Pass |
| TT02 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | one####TRUSTTEXT####less | one####TRUSTTEXT####less | one####TRUSTTEXT####less | Pass |
| TT03 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | #####TRUSTEXT##### | | #####TRUSTEXT##### | Fail |
| TT05 | Removes TRUSTTEXT label from string | trustTextDriver.php | weblib | trusttext_strip | ###one##TRUSTTEXT##### | ###one##TRUSTTEXT##### | ###one##TRUSTTEXT##### | Pass |

# Injection Learning Outcomes

- Injecting a new return in parse_charset sent tremors through the entire project
- Be sure you know how the function is used before making additions
- Expected failure of the parse_charset test cases caused failure in other test cases that used parse_charset somewhere down the line
- Even though this was an exercise where we intentionally injected errors, it demonstrates the importance of speaking with a supervisor and having a full understanding of even the small changes when contributing to code.

# Conclusions

- Open source software can be difficult to work with
- Even small projects are often incredibly complex
- Simple is often better, when possible
- Communication and a schedule are integral to a well functioning team
- Changing one small piece of code can have devastating consequences
- Staying organized is hugely beneficial
- It’s vitally important to read the directions

# Poster

## Why moodle?

Moodle attracted the interest of our group both because of its level of documentation, and because we felt as a group that we could work effectively with this project. The prospect of working with php, a language nobody on our team has used other than on occasion, also excited all three of us. Sure enough, we fairly quickly worked through the entire build process and had Moodle up and running in about an hour.

## Test Plan

**Testing Process**

In order to test the desired method of the class we are interested in, we want to call a program at the command line that runs each test. The program will automatically run all of the tests one by one. The results of the tests will be compared to the expected results of the test cases, and then a test report with the results and other info will be displayed in a browser window.

**Requirements Traceability**

TC01 | Creating a valid timestamp

Timestamps are very important for every system, particularly for our endeavors in the Unix systems. In such systems, they are based on a running total of seconds that begins at the Unix epoch, which is advantageous as they can represent all timezones with a single measurement. As such, this facilitates users' ability to see a website (like Moodle) in real time as opposed to the timezone the server is located in.

**Tested Items**

"Convert_to_timestamp" in the "type_base" class of the Moodle project moodle\calendar\classes\type_base.php

**Testing Schedule**

For the next week, we need to establish core functionality between script, driver, and php testing classes. Once established, we will spend this first week testing the Unix Timestamp method of the type_base class with its many test cases. The following weeks will be dedicated to applying the same techniques to other test cases that we will discover. Methods that use any arithmetic to calculate grades, or calendar dates would be ideal. Successful connections to the database/browser could be another test. We are currently looking into this and several other possibilities.

**Test Recording Procedures**

All of the actual test results will be compared to the expected test results contained in various appropriately labeled json files.

The results of the comparisons described above will be displayed in a browser. The information will be properly and clearly labeled in a uniform manner with each test case getting its own portion of the report.

**Hardware and Software Requirements**

Linux OS

A computer

**Constraints**

We have a limited schedule to figure out the relation of different pieces to our framework.

## Building our first Driver & Script

Initial Driver

Initial Script

Output

**AOBTD TESTING FRAMEWORK**

### Hurdles Faced

As with every part of the project so far, we faced a number of hurdles we had to overcome. Aside from the typical struggles with technology/computers and simple errors like syntax errors, we faced two real challenges during this phase. The first involved a config file that we didn't realize had to be tailored to each individual user. The other problem was actually related to this issue, though is a different issue altogether involving the same file.

Our initial issue was relatively easy to solve. Having configured the script in a way that it should at least run, two of us were receiving a rather obscure error message about a file named config.php while the third group member's code was running perfectly. After some investigating, we realized that all three of us had our config.php files set with the Moodle database details for the group member whose code was working. After correcting these discrepancies by filling the config.php file with our own Moodle database details, our code ran without any issue, yet this wasn't the end of the hurdles we faced.

The other significant issue came with the resolution of our previous dilemma. Having personalized our c onfig.php files, we realized that pushing our updates to our github repo and pulling them would write over each other's config.php files, thus necessitating us to perform the same fix from the previous issue e ach time we pull from our repo. To solve this, it was decided that we should include our config.php file in a .gitIgnore folder so that github will ignore the config.php file when we push to and pull from our repo. These were the biggest issues we faced over the course of this deliverable, and thankfully we seem to have a handle on them.

## Full Testing Suite

Refactored parse_charset driver

The Refactored Script

parse_charset method signature in Moodle

```
public static function parse_charset($charset)
```

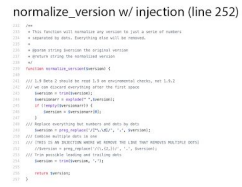
break_up_long_words driver

break_up_long_words method signature in Moodle

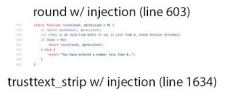
```
function break_up_long_words($string, $maxsize=20, $cutchars=' ')
```

## Error Insertion and Analysis

normalize_version w/ injection (line 252)

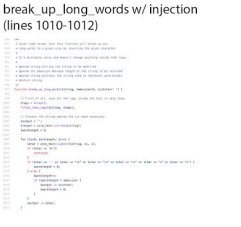


round w/ injection (line 603)

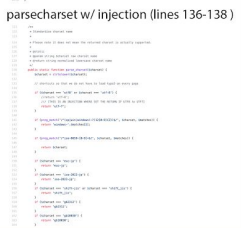


trusttext_strip w/ injection (line 1634)

break_up_long_words w/ injection (lines 1010-1012)



parsecharset w/ injection (lines 136-138 )



## Overall Experience

As a whole, this project was not what any of us expected. We all thought that working on an open source project and developing a program such as the one we wrote would be different than from how it actually was. Of course while we each had our own unique expectations, we were all still surprised by how even the simple things such as working in a team turned out to be different. Nonetheless it was still a very challenging and constructive endeavor, and it's safe to say that we all learned something.

I think one of the biggest misconceptions we had about working with an open source project is that we figured it would be, well, better. Moodle is obviously a large project with years of development behind it, something that in and of itself is impressive. And for the most part, Moodle not only functioned well, but appeared to be fairly polished in terms of documentation and project architecture. However, we were still surprised at how some small things within Moodle seemed wrong or misleading, such as how our round test method actually just truncated the input. Perhaps a more glaring mistake is that some methods lacked any significant documentation. Considering the size of the project, we also found this to be surprising.

One of the other surprising factors for us was the need for efficiency. We covered the refactor process fairly in depth previously in this report, but one small piece of that refactor process involved reworking some code that worked terribly inefficiently. While we actually never ran into this issue, we were aware that another project group with a similar strategy ran into an issue that caused their program to crash. Clearly this scared us enough to go and quickly solve the issue so that our program won't crash as a result of our being inefficient. In hindsight, this makes perfect sense, but it was a tad surprising to us that such a seemingly innocuous design choice could wreak havoc like that down the road.

One of the final surprises for us was learning how to work as a team. We pretty quickly divvied up the roles and duties we would each have throughout the course of this project, but we weren't always good at sticking to them. Ultimately we managed to push out a product that we can all be proud of, but there were moments when the roles shifted because things needed to get done. None of this is to say that we clashed as a team or anything like that, in fact we remained largely disagreement free and never degenerated into any sort of argument. Still, the changing team dynamics and responsibilities on the fly and as we encountered problems was a surprise to each of us.

# Q&A

Thank You!!!