Unit Test Documentation

SaltOS 4.0 r1924

April 2025

Contents

1 Phpunit Libraries

1.1 Autoload file for the unit tests

```
lib/autoload.php
```

This file contains the code that initialize the unit tests

1.1.1 Importing namespaces

```
use PHPUnit\Framework\Assert;
```

1.1.2 Main autoloader code

This code emmulates the index.php by loading all autoload files excep the zindex.php, initialize the timer and the random generator

1.2 Unit test helper functions

```
lib/utestlib.php
```

This file contains the functions used by the web and cli unit tests to get the coverage, too contains the pcov and mime helpers.

1.2.1 Importing namespaces

```
use SebastianBergmann\CodeCoverage\Data\RawCodeCoverageData;
```

1.2.2 Test PCOV start helper

```
function test_pcov_start(): void
```

This function puts the flag needed to enable the pcov feature, to do it, the function creates a void file and puts permissions to allow that other procs can write inside of it

1.2.3 Test PCOV stop helper

```
function test_pcov_stop($index): void
```

This function gets the contents of the coverage file and removes it, too, it does all needed things to append the collected coverage to the current unit test coverage instance

- @index => index used to get the backtrace, it depends from where you are calling this function, generally is 2 but in some cases you need to use another value like 1.

1.2.4 Test WEB helper

```
function test_web_helper($rest, $data, $token, $lang)
```

This function performs the action defined by the rest verb sendind the data if it is provided and using the token for authentication actions.

As you can see in the code, the function detects if data is provided and send the request using GET or POST, in addition, an application/json content-type header is send when POST is used.

The token is sent using the TOKEN header to be used in the authentication process.

- @rest => The rest request, like update/customers/3
- @data => The data used as json in the SaltOS app
- @token => The token used if authentication is required

1.2.5 Test CLI helper

```
function test_cli_helper($rest, $data, $token, $lang, $user)
```

This function allow to execute SaltOS using the CLI SAPI, to do it, the function detects if data is provided, and executes the command and getting the output of the execution. If data exists, then the contents are stored in a file and passed the contents of the file to the stdin of the php process to emmulate the input channel used by the apache server.

As an example, this functions tries to execute the command using the follow formula:

1) php index.php $rest

2) cat /tmp/input | php index.php $rest

In addition, the token field is used to define the TOKEN environment variable that is used by SaltOS as variable to emmulate the TOKEN used by the apache for authenticate the SaltOS app.

- @rest => The rest request, like update/customers/3
- @data => The data used as json in the SaltOS app
- @token => The token used if authentication is required

1.2.6 Get Mime helper

```
function get_mime($buffer): string
```

This function returns the content type of the contents of the buffer

- @buffer => the contents that you want to check

1.2.7 External tests

```
function test_external_exec($test_file, $error_file, $error_words): void
```

This function tries to execute external php that triggers special conditions suck as errors, that are not allowed from the phpunit tests

- @test_file => the file that must be executed
- @error_file => the file that must contains the error_words
- @error_words => the words that must exists in the error_file

2 Phpunit Code

2.1 Test actions

```
test_actions.php
```

This test performs some tests to validate the correctness of the actions functions

2.1.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.1.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.1.3 Main class of this unit test

```
final class test_actions extends TestCase
```

2.1.4 actions test

```
public function test_actions(): void
```

This test performs some tests to validate the correctness of the actions functions

2.2 Test addlog

```
test_add.php
```

This test performs some tests to validate the correctness of the addlog functions

2.2.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.2.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.2.3 Main class of this unit test

```
final class test_add extends TestCase
```

2.2.4 addlog test

```
public function test_addlog(): void
```

This test performs some tests to validate the correctness of the addlog functions

2.2.5 adderror test

```
public function test_adderror(): void
```

This test performs some tests to validate the correctness of the adderror functions

2.3 Test apache

```
test_apache.php
```

This test performs some tests to validate the correctness of the apache configuration

2.3.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.3.2 Main class of this unit test

```
final class test_apache extends TestCase
```

2.3.3 addlog test

```
public function test_apache(): void
```

This test performs some tests to validate the correctness of the apache configuration, the main idea is that the path saltos/code4 of the web server must point to the web directory.

2.4 Test apps

```
test_apps.php
```

This test performs some tests to validate the correctness of the apps functions

2.4.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.4.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.4.3 Main class of this unit test

```
final class test_apps extends TestCase
```

2.4.4 apps test

```
public function test_apps(): void
```

This test performs some tests to validate the correctness of the apps functions

2.4.5 app test

```
public function test_app(): void
```

This test performs some tests to validate the correctness of the app functions

2.4.6 list test

```
public function test_list(): void
```

This test performs some tests to validate the correctness of the list functions

2.5 Test array

```
test_array.php
```

This test performs some tests to validate the correctness of the array functions

2.5.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.5.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.5.3 Main class of this unit test

```
final class test_array extends TestCase
```

2.5.4 array test

```
public function test_array(): void
```

This test performs some tests to validate the correctness of the array functions

2.6 Test ascii

```
test_ascii.php
```

This test performs some tests to validate the correctness of the ascii functions

2.6.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.6.2 Loading helper function

```
require_once 'php/lib/ascii.php';
```

This file contains the needed function used by the unit tests

2.6.3 Main class of this unit test

```
final class test_ascii extends TestCase
```

2.6.4 ascii test

```
public function test_ascii(): void
```

This test performs some tests to validate the correctness of the ascii functions

2.7 Test authupdate

```
test_authupdate.php
```

This test performs some tests to validate the correctness of the authupdate functions

2.7.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.7.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.7.3 Main class of this unit test

```
final class test_authupdate extends TestCase
```

2.7.4 authupdate test

```
public function test_authupdate(): void
```

This test performs some tests to validate the correctness of the authupdate functions

2.8 Test cli customers

```
test_cli_customers.php
```

This test performs all actions of the customers app suck as: create, insert, list, view, edit, update and delete, using the cli sapi interface

2.8.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.8.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.8.3 Main class of this unit test

```
final class test_cli_customers extends TestCase
```

2.8.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.8.5 Create

```
public function test_create(array $json): array
```

This function execute the creates rest request, and must to get the json with the layout without data

2.8.6 Insert

```
public function test_insert(array $json): array
```

This function execute the insert rest request, to do it send the json with the data that they want to insert and must to get the json with the status and the create_id.

2.8.7 List

```
public function test_list(array $json): array
```

This function execute the list rest request, to do it send the json with the search that they want to use in the list filter and receives the json with the data used to populate the table.

2.8.8 View

```
public function test_view(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.8.9 Edit

```
public function test_edit(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.8.10 Upgrade

```
public function test_update(array $json): array
```

This function execute the update rest request, to do it send the json with the data that they want to update and must to get the json with the status and the updated_id.

2.8.11 Delete

```
public function test_delete(array $json): void
```

This function execute the delete rest request, they must to get the json with the status and the deleted_id.

2.9 Test cli invoices

```
test_cli_invoices.php
```

This test performs all actions of the invoices app suck as: create, insert, list, view, edit, update and delete, using the cli sapi interface

2.9.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.9.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.9.3 Main class of this unit test

```
final class test_cli_invoices extends TestCase
```

2.9.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.9.5 Create

```
public function test_create(array $json): array
```

This function execute the creates rest request, and must to get the json with the layout without data

2.9.6 Insert

```
public function test_insert(array $json): array
```

This function execute the insert rest request, to do it send the json with the data that they want to insert and must to get the json with the status and the create_id.

2.9.7 List

```
public function test_list(array $json): array
```

This function execute the list rest request, to do it send the json with the search that they want to use in the list filter and receives the json with the data used to populate the table.

2.9.8 View

```
public function test_view(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.9.9 Edit

```
public function test_edit(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.9.10 Upgrade

```
public function test_update(array $json): array
```

This function execute the update rest request, to do it send the json with the data that they want to update and must to get the json with the status and the updated_id.

2.9.11 Delete

```
public function test_delete(array $json): void
```

This function execute the delete rest request, they must to get the json with the status and the deleted_id.

2.10 Test cli tokens (first part)

```
test_cli_tokens.php
```

This test performs some part of the actions related with the tokens suck as authtoken and checktoken, using the cli sapi interface

2.10.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.10.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.10.3 Main class of this unit test

```
final class test_cli_tokens extends TestCase
```

2.10.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.10.5 Checktoken

```
public function test_checktoken(array $json): array
```

This function execute the checktoken rest request, and must to get the json with the ok about the valid token that you are trying to check

2.10.6 Deauthtoken

```
public function test_deauthtoken(array $json): array
```

This function execute the deauthtoken rest request, and must to get the json with the ok about the valid token that you are deauthenticate

2.10.7 Checktoken ko

```
public function test_checktoken_ko(array $json): void
```

This function execute the checktoken rest request, and must to get the json with the ko about the invalid token that you are trying to check

2.11 Test color

```
test_color.php
```

This test performs some tests to validate the correctness of the color feature

2.11.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.11.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.11.3 Main class of this unit test

```
final class test_color extends TestCase
```

2.11.4 color test

```
public function test_color(): void
```

This test performs some tests to validate the correctness of the color feature

2.12 Test common

```
test_common.php
```

This test performs some tests to validate the correctness of the common functions

2.12.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.12.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.12.3 Main class of this unit test

```
final class test_common extends TestCase
```

2.12.4 common test

```
public function test_common(): void
```

This test performs some tests to validate the correctness of the common functions

2.13 Test config

```
test_config.php
```

This test performs some tests to validate the correctness of the config functions

2.13.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.13.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.13.3 Main class of this unit test

```
final class test_config extends TestCase
```

2.13.4 config test

```
public function test_config(): void
```

This test performs some tests to validate the correctness of the config functions

2.14 Test control

```
test_control.php
```

This test performs some tests to validate the correctness of the control functions

2.14.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.14.2 Loading helper function

```
require_once 'php/lib/control.php';
```

This file contains the needed function used by the unit tests

2.14.3 Main class of this unit test

```
final class test_control extends TestCase
```

2.14.4 control test

```
public function test_control(): void
```

This test performs some tests to validate the correctness of the control functions

2.14.5 log test

```
public function test_log(): void
```

This test performs some tests to validate the correctness of the log functions

2.14.6 version test

```
public function test_version(): void
```

This test performs some tests to validate the correctness of the version functions

2.14.7 indexing test

```
public function test_indexing(): void
```

This test performs some tests to validate the correctness of the indexing functions

2.15 Test cron

```
test_cron.php
```

This test performs some tests to validate the correctness of the cron functions

2.15.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.15.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.15.3 Main class of this unit test

```
final class test_cron extends TestCase
```

2.15.4 Wait cron

```
private function wait_cron(): void
```

This function waits 10 seconds until all cron processes are running

2.15.5 cron

```
public function test_cron(): void
```

This test performs some tests to validate the correctness of the cron functions

2.16 Test customers

```
test_customers.php
```

This test performs some tests to validate the correctness of the customers functions

2.16.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.16.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.16.3 Main class of this unit test

```
final class test_customers extends TestCase
```

2.16.4 customers test

```
public function test_customers(): void
```

This test performs some tests to validate the correctness of the customers functions

2.17 Test database drivers

```
test_database.php
```

2.17.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.17.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.17.3 Main class of this unit test

```
final class test_database extends TestCase
```

2.17.4 Helper

```
private function test_helper($obj): void
```

This function executes the follow queries and checks the correctness of the driver by comparing the results with the expected results.

The tests that performs are the follow:

- SELECT GROUP_CONCAT(a) test FROM (SELECT 1 a UNION SELECT 2 a UNION SELECT 3 a) a;
- SELECT REPLACE('abc', 'b', 'c') test

- SELECT LPAD('123', '5', '0') test
- SELECT CONCAT('a', 'b', 'c') test
- SELECT CONCAT_WS(',','a','b','c',null,true,false) test
- SELECT UNIX_TIMESTAMP('2024-02-01 12:34:56') test
- SELECT FROM_UNIXTIME(1706787296) test
- SELECT YEAR('2024-02-01 12:34:56') test
- SELECT MONTH('2024-02-01 12:34:56') test
- SELECT WEEK('2024-02-01 12:34:56', 1) test
- SELECT TRUNCATE(1.2345, 2) test
- SELECT DAY('2024-02-01 12:34:56') test
- SELECT DAYOFYEAR('2024-02-01 12:34:56') test
- SELECT DAYOFWEEK('2024-02-01 12:34:56') test
- SELECT HOUR('2024-02-01 12:34:56') test
- SELECT MINUTE('2024-02-01 12:34:56') test
- SELECT SECOND('2024-02-01 12:34:56') test
- SELECT MD5('fortuna') test
- SELECT REPEAT('abc',3) test
- SELECT FIND_IN_SET(3,'1,2,3,4,5') test
- SELECT FIND_IN_SET(6,'1,2,3,4,5') test
- SELECT FIND_IN_SET(3,'12345') test
- SELECT IF(true, 'ok', 'ko') test
- SELECT IF(false, 'ok', 'ko') test
- SELECT IF(null, 'ok', 'ko') test
- SELECT POW(2, 8) test
- SELECT DATE_FORMAT('2024-02-01 12:34:56', '%Y-%m-%d %H:%i:%s') test
- SELECT NOW() test

2.17.5 Last Insert Id helper

```
private function last_insert_id_helper($obj): void
```

This function is intended to check the correctness of the last_insert_id feature implemented in each database driver.

2.17.6 PDO MySQL driver

```
public function test_pdo_mysql(): void
```

This function checks the correctness of the sqlite3 driver by creating a database connection, sendint queries validating the expected results and closing the connection.

2.17.7 MySQL improved driver

```
public function test_mysqli(): void
```

This function checks the correctness of the sqlite3 driver by creating a database connection, sendint queries validating the expected results and closing the connection.

2.17.8 PDO SQLite driver

```
public function test_pdo_sqlite(): void
```

This function checks the correctness of the sqlite3 driver by creating a database connection, sendint queries validating the expected results and closing the connection.

2.17.9 SQLite3 driver

```
public function test_sqlite3(): void
```

This function checks the correctness of the sqlite3 driver by creating a database connection, sendint queries validating the expected results and closing the connection.

2.17.10 PDO mssql driver

```
private function test_pdo_mssql(): void
```

This function checks the correctness of the sqlserver driver by creating a database connection, sendint queries validating the expected results and closing the connection.

2.17.11 Database driver

```
public function test_database(): void
```

This function checks the correctness of the database driver by creating a database connection, sendint queries validating the expected results and closing the connection.

2.18 Test datetime

```
test_datetime.php
```

This test performs some tests to validate the correctness of the datetime functions

2.18.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.18.2 Main class of this unit test

```
final class test_datetime extends TestCase
```

2.18.3 datetime test

```
public function test_datetime(): void
```

This test performs some tests to validate the correctness of the datetime functions

2.19 Test dbschema

```
test_dbschema.php
```

This test performs some tests to validate the correctness of the dbschema functions

2.19.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.19.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.19.3 Main class of this unit test

```
final class test_dbschema extends TestCase
```

2.19.4 dbschema test

```
public function test_dbschema(): void
```

This test performs some tests to validate the correctness of the dbschema functions

2.19.5 sql test

```
public function test_sql(): void
```

This test performs some tests to validate the correctness of the sql functions

2.20 Test emails

```
test_emails.php
```

This test performs some tests to validate the correctness of the emails functions

2.20.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.20.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.20.3 Main class of this unit test

```
final class test_emails extends TestCase
```

2.20.4 emails test

```
public function test_emails(): void
```

This test performs some tests to validate the correctness of the emails functions

2.21 Test error

```
test_error.php
```

This test performs some tests to validate the correctness of the error functions

2.21.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.21.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.21.3 Main class of this unit test

```
final class test_error extends TestCase
```

2.21.4 error function test

```
public function test_error(): void
```

This test performs some tests to validate the correctness of the error functions

2.22 Test exec

```
test_exec.php
```

This test performs some tests to validate the correctness of the exec functions

2.22.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.22.2 Main class of this unit test

```
final class test_exec extends TestCase
```

2.22.3 exec test

```
public function test_exec(): void
```

This test performs some tests to validate the correctness of the exec functions

2.23 Test export

```
test_export.php
```

This test performs some tests to validate the correctness of the export functions

2.23.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.23.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.23.3 Main class of this unit test

```
final class test_export extends TestCase
```

2.23.4 Get data

```
private function get_data($nohead, $length): array
```

This function tries to create an array with rows using the numbers.csv file as source, it is able to add or remove the head row and to select the size of the result array of rows

- @nohead => the nohead field used in the import_file function
- @length => the size of the rows array

2.23.5 export xml test

```
public function test_export_xml(): void
```

This test performs some tests to validate the correctness of the export functions

2.23.6 export csv test

```
public function test_export_csv(): void
```

This test performs some tests to validate the correctness of the export functions

2.23.7 export xlsx test

```
public function test_export_xlsx(): void
```

This test performs some tests to validate the correctness of the export functions

2.23.8 export xls test

```
public function test_export_xls(): void
```

This test performs some tests to validate the correctness of the export functions

2.23.9 export ods test

```
public function test_export_ods(): void
```

This test performs some tests to validate the correctness of the export functions

2.23.10 //˜ * export bytes test

```
//˜ public function test_export_bytes(): void
```

//˜ * This test performs some tests to validate the correctness //˜ * of the export functions

2.23.11 export edi test

```
public function test_export_edi(): void
```

This test performs some tests to validate the correctness of the export functions

2.23.12 export json test

```
public function test_export_json(): void
```

This test performs some tests to validate the correctness of the export functions

2.23.13 tree2array test

```
public function test_export_tree2array(): void
```

This test performs some tests to validate the correctness of the tree2array function

2.23.14 getkeys test

```
public function test_export_getkeys(): void
```

This test performs some tests to validate the correctness of the getkeys function

2.23.15 external exec

```
public function test_export_external(): void
```

This test performs some tests to validate the correctness of the external exec

2.24 Test file

```
test_file.php
```

This test performs some tests to validate the correctness of the file functions

2.24.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.24.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.24.3 Main class of this unit test

```
final class test_file extends TestCase
```

2.24.4 file test

```
public function test_file(): void
```

This test performs some tests to validate the correctness of the file functions

2.25 Test files

```
test_files.php
```

This test performs some tests to validate the correctness of the files feature

2.25.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.25.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.25.3 Main class of this unit test

```
final class test_files extends TestCase
```

2.25.4 files test

```
public function test_files(): void
```

This test performs some tests to validate the correctness of the files feature

2.25.5 notes test

```
public function test_notes(): void
```

This test performs some tests to validate the correctness of the notes feature

2.25.6 fileslog test

```
public function test_fileslog(): void
```

This test performs some tests to validate the correctness of the fileslog feature

2.26 Test gc

```
test_gc.php
```

This test performs some tests to validate the correctness of the gc functions

2.26.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.26.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.26.3 Main class of this unit test

```
final class test_gc extends TestCase
```

2.26.4 gc exec

```
public function test_gc_exec(): void
```

This test performs some tests to validate the correctness of the gc functions

2.26.5 gc upload

```
public function test_gc_upload(): void
```

This test performs some tests to validate the correctness of the gc functions

2.26.6 gc upload

```
public function test_gc_trash(): void
```

This test performs some tests to validate the correctness of the gc functions

2.27 Test gdlib

```
test_gdlib.php
```

This test performs some tests to validate the correctness of the gdlib feature

2.27.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.27.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.27.3 Main class of this unit test

```
final class test_gdlib extends TestCase
```

2.27.4 gdlib test

```
public function test_gdlib(): void
```

This test performs some tests to validate the correctness of the gdlib feature

2.28 Test getdata

```
test_getdata.php
```

This test performs some tests to validate the correctness of the getdata functions

2.28.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.28.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.28.3 Main class of this unit test

```
final class test_getdata extends TestCase
```

2.28.4 getdata test

```
public function test_getdata(): void
```

This test performs some tests to validate the correctness of the getdata functions

2.29 Test gettext

```
test_gettext.php
```

This test performs some tests to validate the correctness of the gettext functions

2.29.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.29.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.29.3 Main class of this unit test

```
final class test_gettext extends TestCase
```

2.29.4 gettext test

```
public function test_gettext(): void
```

This test performs some tests to validate the correctness of the gettext functions

2.30 Test help

```
test_help.php
```

This test performs some tests to validate the correctness of the help functions

2.30.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.30.2 Loading helper function

```
require_once 'php/lib/help.php';
```

This file contains the needed function used by the unit tests

2.30.3 Main class of this unit test

```
final class test_help extends TestCase
```

2.30.4 help test

```
public function test_help(): void
```

This test performs some tests to validate the correctness of the help functions

2.31 Test html

```
test_html.php
```

This test performs some tests to validate the correctness of the html functions

2.31.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.31.2 Loading helper function

```
require_once 'apps/emails/php/html.php';
```

This file contains the needed function used by the unit tests

2.31.3 Main class of this unit test

```
final class test_html extends TestCase
```

2.31.4 html test

```
public function test_html(): void
```

This test performs some tests to validate the correctness of the html functions

2.32 Test IDs

```
test_ids.php
```

This test performs some tests to validate the correctness of the check_ids feature

2.32.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.32.2 Main class of this unit test

```
final class test_ids extends TestCase
```

2.32.3 IDs test

```
public function test_ids(): void
```

This test performs some tests to validate the correctness of the check_ids feature

2.33 Test image

```
test_image.php
```

This test performs some tests to validate the correctness of the barcode, qrcode, captcha and score features

2.33.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.33.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.33.3 Main class of this unit test

```
final class test_image extends TestCase
```

2.33.4 barcode test

```
public function test_barcode(): void
```

This test performs some tests to validate the correctness of the barcode feature

2.33.5 qrcode test

```
public function test_qrcode(): void
```

This test performs some tests to validate the correctness of the qrcode feature

2.33.6 captcha test

```
public function test_captcha(): void
```

This test performs some tests to validate the correctness of the captcha feature

2.33.7 score test

```
public function test_score(): void
```

This test performs some tests to validate the correctness of the score feature

2.34 Test import

```
test_import.php
```

This test performs some tests to validate the correctness of the import functions

2.34.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.34.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.34.3 Main class of this unit test

```
final class test_import extends TestCase
```

2.34.4 import xml test

```
public function test_import_xml(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.5 import csv test

```
public function test_import_csv(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.6 import xlsx test

```
public function test_import_xlsx(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.7 import xls test

```
public function test_import_xls(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.8 import ods test

```
public function test_import_ods(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.9 import bytes test

```
public function test_import_bytes(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.10 import edi test

```
public function test_import_edi(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.11 import json test

```
public function test_import_json(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.12 import helper test

```
public function test_import_helper(): void
```

This test performs some tests to validate the correctness of the import functions

2.34.13 external exec

```
public function test_import_external(): void
```

This test performs some tests to validate the correctness of the external exec

2.35 Test indexing

```
test_indexing.php
```

This test performs some tests to validate the correctness of the indexing functions

2.35.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.35.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.35.3 Main class of this unit test

```
final class test_indexing extends TestCase
```

2.35.4 indexing test

```
public function test_indexing(): void
```

This test performs some tests to validate the correctness of the indexing functions

2.36 Test iniset

```
test_iniset.php
```

This test performs some tests to validate the correctness of the iniset functions

2.36.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.36.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.36.3 Main class of this unit test

```
final class test_iniset extends TestCase
```

2.36.4 iniset test

```
public function test_iniset(): void
```

This test performs some tests to validate the correctness of the iniset functions

2.37 Test invoices

```
test_invoices.php
```

This test performs some tests to validate the correctness of the invoices functions

2.37.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.37.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.37.3 Main class of this unit test

```
final class test_invoices extends TestCase
```

2.37.4 invoices test

```
public function test_invoices(): void
```

This test performs some tests to validate the correctness of the invoices functions

2.38 Test log

```
test_log.php
```

This test performs some tests to validate the correctness of the log functions

2.38.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.38.2 Main class of this unit test

```
final class test_log extends TestCase
```

2.38.3 log test

```
public function test_log(): void
```

This test performs some tests to validate the correctness of the log functions

2.39 Test math

```
test_math.php
```

This test performs some tests to validate the correctness of the math functions

2.39.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.39.2 Loading helper function

```
require_once 'php/lib/math.php';
```

This file contains the needed function used by the unit tests

2.39.3 Main class of this unit test

```
final class test_math extends TestCase
```

2.39.4 math test

```
public function test_math(): void
```

This test performs some tests to validate the correctness of the math functions

2.40 Test memory

```
test_memory.php
```

This test performs some tests to validate the correctness of the memory functions

2.40.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.40.2 Main class of this unit test

```
final class test_memory extends TestCase
```

2.40.3 memory test

```
public function test_memory(): void
```

This test performs some tests to validate the correctness of the memory functions

2.41 Test mime

```
test_mime.php
```

This test performs some tests to validate the correctness of the mime functions

2.41.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.41.2 Loading helper function

```
require_once 'apps/emails/php/html.php';
```

This file contains the needed function used by the unit tests

2.41.3 Main class of this unit test

```
final class test_mime extends TestCase
```

2.41.4 mime test

```
public function test_mime(): void
```

This test performs some tests to validate the correctness of the mime functions

2.42 Test nssdb

```
test_nssdb.php
```

This test performs some tests to validate the correctness of the nssdb functions

2.42.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.42.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.42.3 Main class of this unit test

```
final class test_nssdb extends TestCase
```

2.42.4 nssdb test

```
public function test_nssdb(): void
```

This test performs some tests to validate the correctness of the nssdb functions

2.42.5 certs test

```
public function test_certs(): void
```

This test performs some tests to validate the correctness of the certs functions

2.42.6 actions test

```
public function test_actions(): void
```

This test performs some tests to validate the correctness of the certs functions

2.43 Test output

```
test_output.php
```

This test performs some tests to validate the correctness of the output functions

2.43.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.43.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.43.3 Main class of this unit test

```
final class test_output extends TestCase
```

2.43.4 output test

```
public function test_output(): void
```

This test performs some tests to validate the correctness of the output functions

2.43.5 colorize test

```
public function test_colorize(): void
```

This test performs some tests to validate the correctness of the colorize functions

2.44 Test password

```
test_password.php
```

This test performs some tests to validate the correctness of the password functions

2.44.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.44.2 Loading helper function

```
require_once 'php/lib/password.php';
```

This file contains the needed function used by the unit tests

2.44.3 Main class of this unit test

```
final class test_password extends TestCase
```

2.44.4 password test

```
public function test_password(): void
```

This test performs some tests to validate the correctness of the password functions

2.45 Test pdf

```
test_pdf.php
```

This test performs some tests to validate the correctness of the pdf functions

2.45.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.45.2 Loading helper function

```
require_once 'php/lib/pdf.php';
```

This file contains the needed function used by the unit tests

2.45.3 Main class of this unit test

```
final class test_pdf extends TestCase
```

2.45.4 pdf test

```
public function test_pdf(): void
```

This test performs some tests to validate the correctness of the pdf functions

2.46 Test perms

```
test_perms.php
```

This test performs some tests to validate the correctness of the perms functions

2.46.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.46.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.46.3 Main class of this unit test

```
final class test_perms extends TestCase
```

2.46.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.46.5 perms test

```
public function test_perms(array $json): void
```

This test performs some tests to validate the correctness of the perms functions

2.47 Test ping

```
test_ping.php
```

This test performs some tests to validate the correctness of the ping functions

2.47.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.47.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.47.3 Main class of this unit test

```
final class test_ping extends TestCase
```

2.47.4 ping test

```
public function test_ping(): void
```

This test performs some tests to validate the correctness of the ping functions

2.48 Test push

```
test_push.php
```

This test performs some tests to validate the correctness of the push functions

2.48.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.48.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.48.3 Main class of this unit test

```
final class test_push extends TestCase
```

2.48.4 push

```
public function test_push(): void
```

This test performs some tests to validate the correctness of the push functions

2.49 Test random

```
test_random.php
```

This test performs some tests to validate the correctness of the random functions

2.49.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.49.2 Main class of this unit test

```
final class test_random extends TestCase
```

2.49.3 random test

```
public function test_random(): void
```

This test performs some tests to validate the correctness of the random functions

2.50 Test mime

```
test_replace.php
```

This test performs some tests to validate the correctness of the str_replace instead of the preg_replace

2.50.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.50.2 Main class of this unit test

```
final class test_replace extends TestCase
```

2.50.3 replace test

```
public function test_replace(): void
```

This test performs some tests to validate the correctness of the str_replace_assoc instead of the replace function

2.51 Test roundcube library

```
test_roundcube.php
```

2.51.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.51.2 Main class of this unit test

```
final class test_roundcube extends TestCase
```

2.51.3 html2text

```
public function test_html2text(): void
```

This function checks the correctness of the html2text method provided by the roundcube library.

2.52 Test semaphore

```
test_semaphore.php
```

This test performs some tests to validate the correctness of the semaphore functions

2.52.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.52.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.52.3 Main class of this unit test

```
final class test_semaphore extends TestCase
```

2.52.4 semaphore test

```
public function test_semaphore(): void
```

This test performs some tests to validate the correctness of the semaphore functions

2.53 Test server

```
test_server.php
```

This test performs some tests to validate the correctness of the server functions

2.53.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.53.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.53.3 Main class of this unit test

```
final class test_server extends TestCase
```

2.53.4 server test

```
public function test_server(): void
```

This test performs some tests to validate the correctness of the server functions

2.54 Test sql

```
test_sql.php
```

This test performs some tests to validate the correctness of the sql functions

2.54.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.54.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.54.3 Main class of this unit test

```
final class test_sql extends TestCase
```

2.54.4 SQL test

```
public function test_sql(): void
```

This function performs some tests to validate the correctness of the sql functions

2.55 Test strings

```
test_strings.php
```

This test performs some tests to validate the correctness of the strings functions

2.55.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.55.2 Main class of this unit test

```
final class test_strings extends TestCase
```

2.55.3 strings test

```
public function test_strings(): void
```

This test performs some tests to validate the correctness of the strings functions

2.56 Test mime

```
test_strtr.php
```

This test performs some tests to validate the correctness of the str_replace_assoc instead of the strtr function

2.56.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.56.2 Main class of this unit test

```
final class test_strtr extends TestCase
```

2.56.3 strtr test

```
public function test_strtr(): void
```

This test performs some tests to validate the correctness of the str_replace_assoc instead of the strtr function

2.57 Test system

```
test_system.php
```

This test performs some tests to validate the correctness of the system functions

2.57.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.57.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.57.3 Main class of this unit test

```
final class test_system extends TestCase
```

2.57.4 system test

```
public function test_system(): void
```

This test performs some tests to validate the correctness of the system functions

2.58 Test tokens

```
test_tokens.php
```

This test performs some tests to validate the correctness of the token generator and the token format checker

2.58.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.58.2 Main class of this unit test

```
final class test_tokens extends TestCase
```

2.58.3 Tokens test

```
public function test_tokens(): void
```

This function performs some tests to validate the correctness of the token generator and the token format checker

2.59 Test unoconv

```
test_unoconv.php
```

This test performs some tests to validate the correctness of the unoconv functions

2.59.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.59.2 Loading helper function

```
require_once 'php/lib/unoconv.php';
```

This file contains the needed function used by the unit tests

2.59.3 Main class of this unit test

```
final class test_unoconv extends TestCase
```

2.59.4 Pdf test helper

```
private function test_pdf($input): void
```

This function tries to do the test with unoconv2pdf, checks that the input not exists and the output exists to validate the correctness of the function

- @input => the input file to use in the test

2.59.5 Txt test helper

```
private function test_txt($input): void
```

This function tries to do the test with unoconv2txt, checks that the input not exists and the output exists to validate the correctness of the function

- @input => the input file to use in the test

2.59.6 unoconv test

```
public function test_unoconv(): void
```

This test performs some tests to validate the correctness of the unoconv functions

2.59.7 ocr test

```
public function test_ocr(): void
```

This test performs some tests to validate the correctness of the ocr functions

2.59.8 commands test

```
public function test_commands(): void
```

This test performs some tests to validate the correctness of the commands functions

2.60 Test upload

```
test_upload.php
```

This test performs some tests to validate the correctness of the upload functions

2.60.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.60.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.60.3 Main class of this unit test

```
final class test_upload extends TestCase
```

2.60.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.60.5 Addfiles

```
public function test_addfiles(array $json): array
```

This function execute the addfiles rest request, and must to get the json with the correct data

2.60.6 Delfiles

```
public function test_delfiles(array $json): void
```

This function execute the delfiles rest request, and must to get the json with the correct data

2.60.7 Human size

```
public function test_human_size(): void
```

This function only try to execute the get_human_size function

2.61 Test users

```
test_user.php
```

This test performs some tests to validate the correctness of the users functions

2.61.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.61.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.61.3 Main class of this unit test

```
final class test_user extends TestCase
```

2.61.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.61.5 users test

```
public function test_user(array $json): void
```

This test performs some tests to validate the correctness of the users functions

2.61.6 browser test

```
public function test_browser(): void
```

This test performs some tests to validate the correctness of the browser functions

2.61.7 geoip test

```
public function test_geoip(): void
```

This test performs some tests to validate the correctness of the geoip functions

2.61.8 security test

```
public function test_security(): void
```

This test performs some tests to validate the correctness of the security functions

2.62 Test users

```
test_users.php
```

This test performs some tests to validate the correctness of the users functions

2.62.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.62.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.62.3 Main class of this unit test

```
final class test_users extends TestCase
```

2.62.4 days test

```
public function test_days(): void
```

This test performs some tests to validate the correctness of the days functions

2.62.5 users test

```
public function test_users(): void
```

This test performs some tests to validate the correctness of the users functions

2.62.6 groups test

```
public function test_groups(): void
```

This test performs some tests to validate the correctness of the groups functions

2.63 Test version

```
test_version.php
```

This test performs some tests to validate the correctness of the version related functions

2.63.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.63.2 Main class of this unit test

```
final class test_version extends TestCase
```

2.63.3 version test

```
public function test_version(): void
```

This function performs some tests to validate the correctness of the version related functions

2.64 Test web customers

```
test_web_customers.php
```

This test performs all actions of the customers app suck as: create, insert, list, view, edit, update and delete, using the apache sapi interface

2.64.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.64.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.64.3 Main class of this unit test

```
final class test_web_customers extends TestCase
```

2.64.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.64.5 Create

```
public function test_create(array $json): array
```

This function execute the creates rest request, and must to get the json with the layout without data

2.64.6 Insert

```
public function test_insert(array $json): array
```

This function execute the insert rest request, to do it send the json with the data that they want to insert and must to get the json with the status and the create_id.

2.64.7 List

```
public function test_list(array $json): array
```

This function execute the list rest request, to do it send the json with the search that they want to use in the list filter and receives the json with the data used to populate the table.

2.64.8 View

```
public function test_view(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.64.9 Edit

```
public function test_edit(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.64.10 Upgrade

```
public function test_update(array $json): array
```

This function execute the update rest request, to do it send the json with the data that they want to update and must to get the json with the status and the updated_id.

2.64.11 Delete

```
public function test_delete(array $json): void
```

This function execute the delete rest request, they must to get the json with the status and the deleted_id.

2.65 Test web invoices

```
test_web_invoices.php
```

This test performs all actions of the invoices app suck as: create, insert, list, view, edit, update and delete, using the apache sapi interface

2.65.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.65.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.65.3 Main class of this unit test

```
final class test_web_invoices extends TestCase
```

2.65.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.65.5 Create

```
public function test_create(array $json): array
```

This function execute the creates rest request, and must to get the json with the layout without data

2.65.6 Insert

```
public function test_insert(array $json): array
```

This function execute the insert rest request, to do it send the json with the data that they want to insert and must to get the json with the status and the create_id.

2.65.7 List

```
public function test_list(array $json): array
```

This function execute the list rest request, to do it send the json with the search that they want to use in the list filter and receives the json with the data used to populate the table.

2.65.8 View

```
public function test_view(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.65.9 Edit

```
public function test_edit(array $json): array
```

This function execute the view rest request, intended to retrieve the detail of the app with the layout needed to render it.

2.65.10 Upgrade

```
public function test_update(array $json): array
```

This function execute the update rest request, to do it send the json with the data that they want to update and must to get the json with the status and the updated_id.

2.65.11 Delete

```
public function test_delete(array $json): void
```

This function execute the delete rest request, they must to get the json with the status and the deleted_id.

2.66 Test web tokens (first part)

```
test_web_tokens.php
```

This test performs some part of the actions related with the tokens suck as authtoken and checktoken, using the apache sapi interface

2.66.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.66.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.66.3 Main class of this unit test

```
final class test_web_tokens extends TestCase
```

2.66.4 Authtoken

```
public function test_authtoken(): array
```

This function execute the authtoken rest request, and must to get the json with the valid token to continue in the nexts unit tests

2.66.5 Checktoken

```
public function test_checktoken(array $json): array
```

This function execute the checktoken rest request, and must to get the json with the ok about the valid token that you are trying to check

2.66.6 Deauthtoken

```
public function test_deauthtoken(array $json): array
```

This function execute the deauthtoken rest request, and must to get the json with the ok about the valid token that you are deauthenticate

2.66.7 Checktoken ko

```
public function test_checktoken_ko(array $json): void
```

This function execute the checktoken rest request, and must to get the json with the ko about the invalid token that you are trying to check

2.67 Test XML

```
test_xml.php
```

This test performs some tests to validate the correctness of the xml related functions

2.67.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.67.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.67.3 Main class of this unit test

```
final class test_xml extends TestCase
```

2.67.4 XML test

```
public function test_xml(): void
```

This function performs some tests to validate the correctness of the xml related functions

2.68 Test YAML

```
test_yaml.php
```

This test performs some tests to validate the correctness of the yaml related functions

2.68.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.68.2 Main class of this unit test

```
final class test_yaml extends TestCase
```

2.68.3 YAML test

```
public function test_yaml(): void
```

This function performs some tests to validate the correctness of the yaml related functions

2.69 Test zindex

```
test_zindex.php
```

This test performs some tests to validate the correctness of the zindex functions

2.69.1 Importing namespaces

```
use PHPUnit\Framework\TestCase;
```

2.69.2 Loading helper function

```
require_once 'lib/utestlib.php';
```

This file contains the needed function used by the unit tests

2.69.3 Main class of this unit test

```
final class test_zindex extends TestCase
```

2.69.4 zindex test

```
public function test_zindex(): void
```

This test performs some tests to validate the correctness of the zindex functions

3 Jest Libraries

3.1 Setup file for the unit tests

```
lib/jest.setup.js
```

This file contains the code that initialize the unit tests

3.1.1 Detect what type of test is running, and load the needed setup

```
if ('fetch' in global) {
```

3.2 Setup file for the unit tests

```
lib/jsdom.setup.js
```

This file contains the code that initialize the unit tests

3.2.1 Needed by all user interface features

```
global.bootstrap = require('../../code/web/lib/bootstrap/bootstrap.bundle.
    min.js');
```

3.2.2 Needed by bootstrap, core and proxy modules

```
global.md5 = require('../../code/web/lib/md5/md5.min.js');
```

3.2.3 Needed by bootstrap module

```
global.window.matchMedia = function() {
```

3.2.4 This is the same that object.js for the global scope

```
global.saltos = {};
```

3.2.5 Load all files of the project

```
/*const files = `core,bootstrap,storage,hash,token,auth,window,
```

3.2.6 My Require

```
global.myrequire = (file, fns)
```

This function is intended to add the needed module.exports at the end of the file that you want to process, this is a temporary action used only for the require action, and at the end, before the return, the original code is saved as a restore action to maintain the original code file

3.3 Setup file for the unit tests

```
lib/node.setup.js
```

This file contains the code that initialize the unit tests

3.3.1 This is the same that object.js for the global scope

```
global.saltos = {};
```

3.3.2 Needed by core module

```
global.window = {
```

3.3.3 Needed by core module

```
global.document = {
```

3.3.4 Load all files of the project

```
require('../../code/web/js/core.js');
```

3.3.5 My Pause

```
global.mypause = (page, delay)
```

This function is intended to do a pause inside the browser, to do it, we use a string instead of real code because istanbul tries to inject code and fails in runtime, one solution can be to put "istanbul ignore next" in a comment before the next page.evaluate, but I prefer to use a string in the page.evaluate because it is more simple for me

4 Jest Code

4.1 Apps unit tests

```
test_apps.js
```

This file contains the apps unit tests

4.1.1 Puppeteer setup

```
const puppeteer = require('puppeteer');
```

This lines contain the needed setup for run puppeteer and take screenshots

4.1.2 Global variables

```
let browser;
```

This variables contains the browser and page links

4.1.3 Before All

```
beforeAll(async ()
```

This function contains all code executed before all tests, in this case the features provided by this function includes the launch of the browser, set the screen size and start the javascript coverage

4.1.4 After All

```
afterAll(async ()
```

This function contains all code executed after all tests, in this case the features provided by this function include the stop of the javsacript coverage recording, the save feature to the desired storage path and the browser close

4.1.5 Workflow variables

```
let testFailed = false;
```

This variables allow to control the workflow of the test, the main idea is to skip all tests when one test fails

4.1.6 Before Each

```
beforeEach(()
```

This function contains all code executed before each test, in this case the features provided by this function includes the control of the workflow

4.1.7 After Each

```
afterEach(()
```

This function contains all code executed after each test, in this case the features provided by this function includes the control of the workflow

4.1.8 App Login

```
describe('App Login', ()
```

This test is intended to validate the correctness of the login screen and their particularities, too tries to check the dashboard and the logout to close the loop of tests

4.1.9 Action Login

```
test('Action Login', async ()
```

This function tries to do a test to validate that the login screen appear correctly without issues

4.1.10 Action Login Ko Red

```
test('Action Login Ko Red', async ()
```

This function tries to validate the form when no data is found

4.1.11 Action Login Ko Green

```
test('Action Login Ko Green', async ()
```

This function tries to validate the form when invalid data is found

4.1.12 Action Dashboard

```
test('Action Dashboard', async ()
```

This function tries to execute the login with valid credentials, to validate the correctness, the dashboard will appear

4.1.13 Action Reload

```
test('Action Reload', async ()
```

This function tries to do a reload of the previous page, the validation of this test must accomplish when a valid dashbord appear

4.1.14 Action Logout

```
test('Action Logout', async ()
```

This test is intended to check the correctness of the logout feature, to to it the test tries to locate the logout button placed in the dropdown menu of the nabvar and click to the third element

4.2 Bootstrap unit tests

```
test_bootstrap.js
```

This file contains the bootstrap unit tests

4.2.1 Puppeteer setup

```
const puppeteer = require('puppeteer');
```

This lines contain the needed setup for run puppeteer and take screenshots

4.2.2 Bootstrap

```
describe('Bootstrap', ()
```

This test contains the code needed to create all widgets and validate the correctness of them

4.2.3 Global variables

```
let browser;
```

This variables contains the browser and page links

4.2.4 Before All

```
beforeAll(async ()
```

This function contains all code executed before all tests

4.2.5 After All

```
afterAll(async ()
```

This function contains all code executed after all tests

4.2.6 Prepare the test.each iterator

```
const fs = require('fs');
```

4.2.7 Real test

```
test.each(json)('$label', async field
```

This function executes the real test for each json item, its able to create a widget and validate the correctness of the widget comparing the new widget screenshot to the backup widget screenshot

4.3 Core unit tests

```
test_core.js
```

This file contains comprehensive unit tests for the core functionality of the SaltOS framework, including error handling, AJAX operations, utility functions, and service worker integration.

4.3.1 Load all needed files of the project

```
const files = `core,gettext,token`.split(',');
```

4.3.2 Reset mocks before each test

```
beforeEach(()
```

Initializes mock implementations for console.log and fetch API before each test case runs

4.3.3 Restore mocks after each test

```
afterEach(()
```

Restores original implementations of mocked functions after each test case completes

4.3.4 Test suite for error and log reporting

```
describe('saltos.core.adderror/addlog', ()
```

Contains tests for core error handling and logging functionality

4.3.5 Setup before each test in this suite

```
beforeEach(()
```

Mocks core dependencies including AJAX, token, and gettext functions

4.3.6 Test error reporting functionality

```
test('saltos.core.adderror should send error details to the server', async
   ()
```

Verifies that error details including stack traces are properly processed and sent to the server

4.3.7 Test log reporting functionality

```
test('saltos.core.addlog should send log message to the server', ()
```

Verifies that log messages are properly formatted and sent to the server

4.3.8 Test suite for global error handling

```
describe('window.addEventListener for error and unhandledrejection', ()
```

Contains tests for window-level error and unhandled promise rejection handlers

4.3.9 Setup before each test in this suite

```
beforeEach(()
```

Mocks the core error reporting function

4.3.10 Test global error handling

```
test('should call saltos.core.adderror when an error occurs', ()
```

Verifies that window error events are properly captured and reported

4.3.11 Test unhandled promise rejection handling

```
test('should call saltos.core.adderror when an unhandledrejection occurs',
   ()
```

Verifies that unhandled promise rejections are properly captured and reported

4.3.12 Test parameter validation functionality

```
test('saltos.core.check_params', ()
```

Verifies that missing parameters are properly initialized with default values

4.3.13 Test unique ID generation

```
test('saltos.core.uniqid', ()
```

Verifies that generated IDs are unique and follow expected format

4.3.14 Test suite for visibility detection

```
describe('when_visible', ()
```

Contains tests for the when_visible utility function

4.3.15 Setup before each test in this suite

```
beforeEach(()
```

Configures fake timers and clears DOM

4.3.16 Cleanup after each test in this suite

```
afterEach(()
```

Clears timers and restores real timer implementation

4.3.17 Test basic visibility detection

```
test('executes the callback when the object becomes visible', ()
```

Verifies callback is executed when element becomes visible

4.3.18 Test visibility detection with ID

```
test('works setting the id as test-div', ()
```

Verifies function works with elements that have IDs

4.3.19 Test visibility detection with string ID

```
test('works with an string id instead of an object', ()
```

Verifies function works when passed an element ID string

4.3.20 Test invalid input handling

```
test('throws an error for unsupported obj type', ()
```

Verifies function throws error for unsupported input types

4.3.21 Test delayed element attachment

```
test('append the object after some iterations', ()
```

Verifies function works when element is added to DOM after initialization

4.3.22 Test element removal handling

```
test('throws an error if the object disappears before being visible', ()
```

Verifies function throws error if element is removed before becoming visible

4.3.23 Test key code extraction

```
test('saltos.core.get_keycode', ()
```

Verifies function correctly extracts key codes from different event properties

4.3.24 Test key name resolution

```
test('saltos.core.get_keyname', ()
```

Verifies function correctly maps key codes to common key names

4.3.25 Test suite for HTML element creation

```
describe('saltos.core.html', ()
```

Contains tests for the html utility function

4.3.26 Test single argument usage

```
test('creates a div with inner HTML when only one argument is passed', ()
```

Verifies function creates element from HTML string with optimization

4.3.27 Test two argument usage

```
test('creates the specified element with inner HTML when two arguments are
    passed', ()
```

Verifies function creates specified element type with content

4.3.28 Test HTML trimming

```
test('trims the inner HTML before setting it', ()
```

Verifies function trims whitespace from HTML content

4.3.29 Test multiple children handling

```
test('does not optimize if there are multiple children', ()
```

Verifies function preserves container element when multiple children exist

4.3.30 Test single child optimization

```
test('optimizes and returns the single child if present', ()
```

Verifies function optimizes by returning single child element directly

4.3.31 Test suite for AJAX functionality

```
describe('saltos.core.ajax', ()
```

Contains comprehensive tests for the core AJAX implementation

4.3.32 Test successful GET request

```
test('makes a successful GET request and calls success callback', async ()
```

Verifies proper request construction and success callback handling

4.3.33 Test error response handling

```
test('handles non-200 responses and calls error callback', async ()
```

Verifies proper error callback invocation for non-200 responses

4.3.34 Test POST request handling

```
test('makes a POST request with a body', async ()
```

Verifies proper construction of POST requests with body content

4.3.35 Test request abortion

```
test('calls abort callback when request is aborted', async ()
```

Verifies proper handling of aborted requests

4.3.36 Test network failure handling

```
test('calls error callback on network failure', async ()
```

Verifies proper error callback invocation for network failures

4.3.37 Test unexpected error handling

```
test('throws an error for unexpected failures', async ()
```

Verifies proper propagation of unexpected errors

4.3.38 Test unsupported method handling

```
test('throws an error for unsupported HTTP method', ()
```

Verifies proper error throwing for unsupported HTTP methods

4.3.39 Test XML response handling

```
test('handles XML response correctly', async ()
```

Verifies proper parsing of XML response content

4.3.40 Test plain text response handling

```
test('handles plain text response correctly', async ()
```

Verifies proper handling of plain text responses

4.3.41 Test key normalization

```
test('saltos.core.fix_key', ()
```

Verifies function properly removes suffixes from keys

4.3.42 Test object copying

```
test('saltos.core.copy_object', ()
```

Verifies function creates proper shallow copies of objects

4.3.43 Test suite for resource loading

```
describe('saltos.core.require', ()
```

Contains tests for the require functionality

4.3.44 Setup before each test in this suite

```
beforeEach(()
```

Configures fake timers for testing asynchronous operations

4.3.45 Cleanup after each test in this suite

```
afterEach(()
```

Restores real timer implementation

4.3.46 Test JavaScript file loading

```
test('loads a JavaScript file successfully', async ()
```

Verifies proper loading and injection of JavaScript files

4.3.47 Test CSS file loading

```
test('loads a CSS file successfully', async ()
```

Verifies proper loading and injection of CSS files

4.3.48 Test cached resource handling

```
test('does not reload already loaded files', async ()
```

Verifies that already loaded resources are not reloaded

4.3.49 Test pending resource handling

```
test('waits when file is already loading', async ()
```

Verifies proper waiting behavior for resources already being loaded

4.3.50 Test module file loading

```
test('loads a JavaScript module file successfully', async ()
```

Verifies proper handling of JavaScript module files

4.3.51 Test hashed resource loading

```
test('loads a CSS file with hash successfully', async ()
```

Verifies proper handling of resources with cache-busting hashes

4.3.52 Test boolean evaluation

```
test('saltos.core.eval_bool', ()
```

Verifies proper conversion of various values to boolean

4.3.53 Test string conversion

```
test('saltos.core.toString', ()
```

Verifies proper string conversion of various values

4.3.54 Test attribute value detection

```
test('saltos.core.is_attr_value', ()
```

Verifies proper identification of attribute-value objects

4.3.55 Test attribute-value joining

```
test('saltos.core.join_attr_value', ()
```

Verifies proper merging of attribute and value objects

4.3.56 saltos.core.encode_bad_chars

```
test('saltos.core.encode_bad_chars', ()
```

This function performs the test of the encode_bad_chars function

4.3.57 saltos.core.__get_code_from_file_and_line

```
test('saltos.core.__get_code_from_file_and_line', ()
```

This function performs the test of the __get_code_from_file_and_line function

4.3.58 saltos.core.timestamp

```
test('saltos.core.timestamp', ()
```

This function performs the test of the timestamp function

4.3.59 saltos.core.human_size

```
test('saltos.core.human_size', ()
```

This function performs the test of the human_size function

4.3.60 saltos.core.is_number

```
test('saltos.core.is_number', ()
```

This function performs the test of the is_number function

4.3.61 Core Module Tests

```
describe('Core Module Tests', ()
```

This test suite validates the core functionalities of the SaltOS framework, ensuring that essential features operate correctly under different scenarios.

4.3.62 Test service worker registration

```
test('Registers service worker if supported and on HTTPS', async ()
```

This test checks if the service worker is registered properly when the browser supports service workers and the application is running over HTTPS.

4.3.63 Test proxy function messaging

```
test('Proxy function sends message to service worker', ()
```

This test ensures that the proxy function sends the correct message to the service worker's controller using the 'postMessage' method.

4.3.64 Test proxy synchronization on online events

```
test('Triggers proxy sync on online event', ()
```

This test verifies that the proxy synchronization is triggered correctly when the browser's online event is fired.

4.3.65 Test network protocol detection

```
test('Check network detects protocols correctly', async ()
```

This test checks that the 'check_network' function accurately detects HTTP and HTTPS protocols by simulating network conditions.

4.4 Bootstrap unit tests

```
test_customers.js
```

This file contains the bootstrap unit tests

4.4.1 Puppeteer setup

```
const puppeteer = require('puppeteer');
```

This lines contain the needed setup for run puppeteer and take screenshots

4.4.2 Global variables

```
let browser;
```

This variables contains the browser and page links

4.4.3 Before All

```
beforeAll(async ()
```

This function contains all code executed before all tests, in this case the features provided by this function includes the launch of the browser, set the screen size and start the javascript coverage

4.4.4 After All

```
afterAll(async ()
```

This function contains all code executed after all tests, in this case the features provided by this function include the stop of the javsacript coverage recording, the save feature to the desired storage path and the browser close

4.4.5 Workflow variables

```
let testFailed = false;
```

This variables allow to control the workflow of the test, the main idea is to skip all tests when one test fails

4.4.6 Before Each

```
beforeEach(()
```

This function contains all code executed before each test, in this case the features provided by this function includes the control of the workflow

4.4.7 After Each

```
afterEach(()
```

This function contains all code executed after each test, in this case the features provided by this function includes the control of the workflow

4.4.8 App Customers

```
describe('App Customers', ()
```

This test is intended to validate the correctness of the customers application by execute the list, more, reset, create, cancel, view, close, edit, back, insert, update and delete features and validate with the expected screenshot

4.4.9 Action List

```
test('Action List', async ()
```

This part of the test tries to load the list screen

4.4.10 Action List

```
test('Action More', async ()
```

This part of the test tries to validate the correctness of the more feature

4.4.11 Action Reset

```
test('Action Reset', async ()
```

This part of the test tries to validate the correctness of the reset feature

4.4.12 Action Create

```
test('Action Create', async ()
```

This part of the test tries to load the create screen

4.4.13 Action Cancel

```
test('Action Cancel', async ()
```

This part of the test tries to validate the correctness of the cancel feature

4.4.14 Action View

```
test('Action View', async ()
```

This part of the test tries to load the view screen

4.4.15 Action Close

```
test('Action Close', async ()
```

This part of the test tries to validate the correctness of the close feature

4.4.16 Action Edit

```
test('Action Edit', async ()
```

This part of the test tries to load the edit screen

4.4.17 Action Go Back

```
test('Action Go Back', async ()
```

This part of the test tries to validate the correctness of the go back feature

4.4.18 Action Insert

```
test('Action Insert', async ()
```

This part of the test tries to validate the correctness of the insert feature

4.4.19 Action Update

```
test('Action Update', async ()
```

This part of the test tries to validate the correctness of the update feature

4.4.20 Action Delete

```
test('Action Delete', async ()
```

This part of the test tries to validate the correctness of the delete feature

4.5 Bootstrap unit tests

```
test_emails.js
```

This file contains the bootstrap unit tests

4.5.1 Puppeteer setup

```
const puppeteer = require('puppeteer');
```

This lines contain the needed setup for run puppeteer and take screenshots

4.5.2 Global variables

```
let browser;
```

This variables contains the browser and page links

4.5.3 Before All

```
beforeAll(async ()
```

This function contains all code executed before all tests, in this case the features provided by this function includes the launch of the browser, set the screen size and start the javascript coverage

4.5.4 After All

```
afterAll(async ()
```

This function contains all code executed after all tests, in this case the features provided by this function include the stop of the javscript coverage recording, the save feature to the desired storage path and the browser close

4.5.5 Workflow variables

```
let testFailed = false;
```

This variables allow to control the workflow of the test, the main idea is to skip all tests when one test fails

4.5.6 Before Each

```
beforeEach(()
```

This function contains all code executed before each test, in this case the features provided by this function includes the control of the workflow

4.5.7 After Each

```
afterEach(()
```

This function contains all code executed after each test, in this case the features provided by this function includes the control of the workflow

4.5.8 App Emails

```
describe('App Emails', ()
```

This test is intended to validate the correctness of the emails application by execute the list, control+f, profile, help, filter, create and view features and validate with the expected screenshot

4.5.9 Action List

```
test('Action List', async ()
```

This action tries to sets the emails as new and not new of all emails that appear in the screen

4.5.10 Action Control F

```
test('Action Control F', async ()
```

This part of the test tries to validate the control+f focus

4.5.11 Action Profile

```
test('Action Profile', async ()
```

This part of the test tries to load the profile screen

4.5.12 Action Help

```
test('Action Help', async ()
```

This part of the test tries to load the help screen

4.5.13 Action Filter

```
test('Action Filter', async ()
```

This part of the test tries to load the filter screen

4.5.14 Action Create

```
test('Action Create', async ()
```

This part of the test tries to load the create screen

4.5.15 Action View

```
test('Action View', async ()
```

This part of the test tries to load the view screen

4.6 Filter unit tests

```
test_filter.js
```

This file contains unit tests for the filter management system including initialization, loading, saving, and UI interactions

4.6.1 Load all needed files of the project

```
const files = `app,backup,core,driver,filter,form,gettext,hash,storage,
   token`.split(',');
```

4.6.2 Reset mocks before each test

```
beforeEach(()
```

Ensures all Jest mocks are reset before each test case runs

4.6.3 Restore mocks after each test

```
afterEach(()
```

Ensures all Jest mocks are restored to their original implementations after each test case completes

4.6.4 Test suite for filter initialization

```
describe('saltos.filter.init()', ()
```

Contains tests for initializing the filter cache from server data

4.6.5 Setup before each test in this suite

```
beforeEach(()
```

Mocks window location and sets up AJAX response for filter data

4.6.6 Test filter cache initialization

```
test('should initialize filter cache', async ()
```

Verifies that the init function makes an AJAX call to load filters and properly populates the cache

4.6.7 Test cache reuse

```
test('should not make ajax call if cache is already populated', async ()
```

Verifies that subsequent init calls don't make AJAX requests when cache is already populated

4.6.8 Test suite for filter loading

```
describe('saltos.filter.load()', ()
```

Contains tests for loading filter data into forms and triggering searches

4.6.9 Setup before each test in this suite

```
beforeEach(()
```

Mocks form data and driver search functions

4.6.10 Test loading existing filter

```
test('should load filter data and trigger search', ()
```

Verifies that loading a filter populates form data and triggers a search

4.6.11 Test loading non-existent filter

```
test('should only trigger search if filter not found', ()
```

Verifies that loading an unknown filter only triggers search without modifying form data

4.6.12 Test suite for filter updates

```
describe('saltos.filter.update()', ()
```

Contains tests for updating the local filter cache

4.6.13 Test updating filter cache

```
test('should update filter cache', ()
```

Verifies that update modifies the filter cache correctly

4.6.14 Test suite for filter saving

```
describe('saltos.filter.save()', ()
```

Contains tests for saving filters to both cache and server

4.6.15 Test saving new filter

```
test('should save new filter to cache and server', ()
```

Verifies that new filters are added to cache and saved to server

4.6.16 Test updating existing filter

```
test('should update existing filter if data changed', ()
```

Verifies that modified filters are updated in cache and server

4.6.17 Test unchanged filter

```
test('should not make server call if data unchanged', ()
```

Verifies that unchanged filters don't trigger server updates

4.6.18 Test filter deletion

```
test('should delete filter when data is null', ()
```

Verifies that filters are removed from cache and server when set to null

4.6.19 Test deleting non-existent filter

```
test('should not delete non-existent filter', ()
```

Verifies that attempting to delete unknown filters doesn't trigger server calls

4.6.20 Test suite for filter UI buttons

```
describe('saltos.filter.button()', ()
```

Contains tests for all filter-related UI actions

4.6.21 Setup before each test in this suite

```
beforeEach(()
```

Sets up DOM elements and mocks filter functions

4.6.22 Test load action

```
test('should handle load action', ()
```

Verifies that load button loads selected filter and resets selection

4.6.23 Test update action

```
test('should handle update action', ()
```

Verifies that update button saves current form data to selected filter

4.6.24 Test delete action

```
test('should handle delete action', ()
```

Verifies that delete button removes selected filter and refreshes UI

4.6.25 Test create action

```
test('should handle create action', ()
```

Verifies that create button saves current form data as new filter

4.6.26 Test rename action

```
test('should handle rename action', ()
```

Verifies that rename button moves filter data to new name and removes old filter entry

4.6.27 Test invalid action

```
test('should do nothing for invalid actions', ()
```

Verifies that unknown actions don't modify filter cache

4.6.28 Test suite for filter selection UI

```
describe('saltos.filter.select()', ()
```

Contains tests for updating filter dropdown and tree view

4.6.29 Setup before each test in this suite

```
beforeEach(()
```

Populates filter cache with test data

4.6.30 Test UI update

```
test('should update select options and jstree', ()
```

Verifies that select updates dropdown options and tree view

4.6.31 Test missing tree element

```
test('should handle missing form elements gracefully', ()
```

Verifies graceful handling when tree element is missing

4.6.32 Test missing select element

```
test('should handle missing select elements gracefully', ()
```

Verifies graceful handling when select element is missing

4.6.33 Test missing form elements

```
test('should handle missing form elements gracefully', ()
```

Verifies graceful handling when all form elements are missing

4.7 Core unit tests

```
test_gettext.js
```

This file contains the core unit tests

4.7.1 Load all needed files of the project

```
const files = `bootstrap,core,gettext,storage`.split(',');
```

4.7.2 berofeEach used in this test

```
beforeEach(()
```

4.7.3 afterEach used in this test

```
afterEach(()
```

4.7.4 saltos.gettext.bootstrap.set/get/get_short/unset

```
test('saltos.gettext.set/get/get_short/unset', ()
```

This function performs the tests of the set/get/get_short and unset functions

4.7.5 saltos.gettext.bootstrap.field

```
test('saltos.gettext.bootstrap.field', ()
```

This function performs the tests of the field function

4.7.6 saltos.gettext.bootstrap.modal

```
test('saltos.gettext.bootstrap.modal', ()
```

This function performs the tests of the modal function

4.7.7 saltos.gettext.bootstrap.toast

```
test('saltos.gettext.bootstrap.toast', ()
```

This function performs the tests of the toast function

4.7.8 saltos.gettext.bootstrap.menu

```
test('saltos.gettext.bootstrap.menu', ()
```

This function performs the tests of the menu function

4.7.9 saltos.gettext.bootstrap.offcanvas

```
test('saltos.gettext.bootstrap.offcanvas', ()
```

This function performs the tests of the offcanvas function

4.8 Hash unit tests

```
test_hash.js
```

This file contains unit tests for hash management functionality including hash parsing, manipulation, and event triggering

4.8.1 Load all needed files of the project

```
const files = `hash`.split(',');
```

4.8.2 Reset mocks before each test

```
beforeEach(()
```

Initializes mock implementations for history API functions and resets all mocks between test cases

4.8.3 Restore mocks after each test

```
afterEach(()
```

Restores original implementations of mocked functions after each test case completes

4.8.4 Test suite for hash helper function

```
describe('saltos.hash.__helper', ()
```

Contains tests for the internal hash normalization function that cleans hash strings by removing special characters

4.8.5 Test hash with leading

```
test('should remove leading #', ()
```

Verifies the helper removes the # character from the beginning

4.8.6 Test hash with leading /

```
test('should remove leading /', ()
```

Verifies the helper removes the / character from the beginning

4.8.7 Test hash with leading #/

```
test('should remove both leading # and /', ()
```

Verifies the helper removes both # and / characters from the beginning

4.8.8 Test empty hash

```
test('should return empty string for empty input', ()
```

Verifies the helper returns empty string for empty input

4.8.9 Test clean hash

```
test('should not modify clean hash', ()
```

Verifies the helper returns unchanged input when no special characters

4.8.10 Test suite for hash getter function

```
describe('saltos.hash.get', ()
```

Contains tests for retrieving the current window hash with proper formatting

4.8.11 Test getting current hash

```
test('should return current hash without #', ()
```

Verifies the function returns hash without # prefix

4.8.12 Test getting empty hash

```
test('should return empty string when no hash', ()
```

Verifies the function returns empty string when no hash exists

4.8.13 Test getting hash with leading /

```
test('should remove leading /', ()
```

Verifies the function removes leading / from the hash

4.8.14 Test suite for hash setter function

```
describe('saltos.hash.set', ()
```

Contains tests for replacing the current hash using history.replaceState

4.8.15 Test setting new hash

```
test('should set new hash with proper format', ()
```

Verifies the function updates hash with proper format and uses replaceState

4.8.16 Test setting same hash

```
test('should not set same hash', ()
```

Verifies the function doesn't update history when hash doesn't change

4.8.17 Test setting hash with

```
test('should handle hash with #', ()
```

Verifies the function properly handles input containing #

4.8.18 Test setting hash with /

```
test('should handle hash with /', ()
```

Verifies the function properly handles input containing /

4.8.19 Test setting empty hash

```
test('should handle empty hash', ()
```

Verifies the function properly handles empty hash input

4.8.20 Test suite for hash adder function

```
describe('saltos.hash.add', ()
```

Contains tests for adding new hash entries using history.pushState

4.8.21 Test adding new hash

```
test('should add new hash with proper format', ()
```

Verifies the function updates hash with proper format and uses pushState

4.8.22 Test adding same hash

```
test('should not add same hash', ()
```

Verifies the function doesn't update history when hash doesn't change

4.8.23 Test adding hash with

```
test('should handle hash with #', ()
```

Verifies the function properly handles input containing #

4.8.24 Test adding hash with /

```
test('should handle hash with /', ()
```

Verifies the function properly handles input containing /

4.8.25 Test adding empty hash

```
test('should handle empty hash', ()
```

Verifies the function properly handles empty hash input

4.8.26 Test suite for URL hash extraction

```
describe('saltos.hash.url2hash', ()
```

Contains tests for extracting hash fragments from URLs

4.8.27 Test extracting hash from URL

```
test('should extract hash from URL', ()
```

Verifies the function extracts hash fragment correctly

4.8.28 Test extracting hash with /

```
test('should handle URL with / after #', ()
```

Verifies the function handles hashes containing /

4.8.29 Test URL without hash

```
test('should return empty string when no hash', ()
```

Verifies the function returns empty string when no hash exists

4.8.30 Test suite for hash change events

```
describe('saltos.hash.trigger', ()
```

Contains tests for triggering hashchange events

4.8.31 Test triggering hash change

```
test('should dispatch hashchange event', ()
```

Verifies the function dispatches a hashchange event

4.9 Bootstrap unit tests

```
test_invoices.js
```

This file contains the bootstrap unit tests

4.9.1 Puppeteer setup

```
const puppeteer = require('puppeteer');
```

This lines contain the needed setup for run puppeteer and take screenshots

4.9.2 Global variables

```
let browser;
```

This variables contains the browser and page links

4.9.3 Before All

```
beforeAll(async ()
```

This function contains all code executed before all tests, in this case the features provided by this function includes the launch of the browser, set the screen size and start the javascript coverage

4.9.4 After All

```
afterAll(async ()
```

This function contains all code executed after all tests, in this case the features provided by this function include the stop of the javsacript coverage recording, the save feature to the desired storage path and the browser close

4.9.5 Workflow variables

```
let testFailed = false;
```

This variables allow to control the workflow of the test, the main idea is to skip all tests when one test fails

4.9.6 Before Each

```
beforeEach(()
```

This function contains all code executed before each test, in this case the features provided by this function includes the control of the workflow

4.9.7 After Each

```
afterEach(()
```

This function contains all code executed after each test, in this case the features provided by this function includes the control of the workflow

4.9.8 App Invoices

```
describe('App Invoices', ()
```

This test is intended to validate the correctness of the invoices application by execute the list, checkbox, create, view and edit features and validate with the expected screenshot

4.9.9 Action List

```
test('Action List', async ()
```

This part of the test tries to load the list screen

4.9.10 Action List

```
test('Action Checkbox', async ()
```

This part of the test tries to validate the correctness of the checkbox feature

4.9.11 Action Create

```
test('Action Create', async ()
```

This part of the test tries to load the create screen

4.9.12 Action View

```
test('Action View', async ()
```

This part of the test tries to load the view screen

4.9.13 Action Edit

```
test('Action Edit', async ()
```

This part of the test tries to load the edit screen

4.10 Proxy unit tests

```
test_proxy.js
```

This file contains the proxy unit tests

4.10.1 berofeEach used in this test

```
beforeEach(()
```

4.10.2 afterEach used in this test

```
afterEach(()
```

4.10.3 Load the needed environment of the proxy part

```
Object.assign(global, myrequire(
```

4.10.4 md5

```
test('md5', ()
```

This function performs the test of the md5 function

4.10.5 human_size

```
test('human_size', ()
```

This function performs the test of the human_size function

4.11 Push unit tests

```
test_push.js
```

This file contains unit tests for the push notification functionality and favicon status updates in the SaltOS framework.

4.11.1 Load all needed files of the project

```
const files = `app,bootstrap,core,gettext,push,storage,token`.split(',');
```

4.11.2 Reset mocks before each test

```
beforeEach(()
```

Ensures all Jest mocks are reset before each test case runs

4.11.3 Restore mocks after each test

```
afterEach(()
```

Ensures all Jest mocks are restored to their original implementations after each test case completes

4.11.4 Test suite for saltos.push.fn functionality

```
describe('saltos.push.fn', ()
```

Contains tests for the push notification system's main function, including various execution conditions and response handling

4.11.5 Setup before each test in this suite

```
beforeEach(()
```

Initializes mock implementations and resets push notification state

4.11.6 Test push function when already executing

```
test('should return early if executing is true', ()
```

Verifies that the push function exits early when a push operation is already in progress

4.11.7 Test push function without token

```
test('should return early if token is not available', ()
```

Verifies that the push function exits early when no authentication token is available

4.11.8 Test push function when offline

```
test('should return early if navigator is offline', ()
```

Verifies that the push function exits early when the browser is offline

4.11.9 Test push function with positive count

```
test('should return early if count is greater than or equal to 0', ()
```

Verifies that the push function exits early when the countdown counter hasn't reached zero

4.11.10 Test push function with successful response

```
test('should call ajax and handle success response', ()
```

Verifies that the push function makes an AJAX call and properly handles a successful response

4.11.11 Test push function with error response

```
test('should handle error response', ()
```

Verifies that the push function properly handles an error response from the server

4.11.12 Test suite for saltos.favicon.fn functionality

```
describe('saltos.favicon.fn', ()
```

Contains tests for the favicon status update functionality, including visibility state handling

4.11.13 Test favicon function activation

```
test('should start interval if bool is true and executing is false', ()
```

Verifies that the favicon update interval starts when the function is activated

4.11.14 Test favicon function deactivation

```
test('should clear interval if bool is false and executing is true', ()
```

Verifies that the favicon update interval stops when the function is deactivated

4.12 Storage unit tests

```
test_storage.js
```

This file contains the storage unit tests

4.12.1 Load all needed files of the project

```
const files = `core,storage`.split(',');
```

4.12.2 berofeEach used in this test

```
beforeEach(()
```

4.12.3 afterEach used in this test

```
afterEach(()
```

4.12.4 saltos.storage

```
test('saltos.storage', ()
```

This function performs the test of the storage functions

4.13 Bootstrap unit tests

```
test_tester.js
```

This file contains the bootstrap unit tests

4.13.1 Puppeteer setup

```
const puppeteer = require('puppeteer');
```

This lines contain the needed setup for run puppeteer and take screenshots

4.13.2 Global variables

```
let browser;
```

This variables contains the browser and page links

4.13.3 Before All

```
beforeAll(async ()
```

This function contains all code executed before all tests, in this case the features provided by this function includes the launch of the browser, set the screen size and start the javascript coverage

4.13.4 After All

```
afterAll(async ()
```

This function contains all code executed after all tests, in this case the features provided by this function include the stop of the javsacript coverage recording, the save feature to the desired storage path and the browser close

4.13.5 Workflow variables

```
let testFailed = false;
```

This variables allow to control the workflow of the test, the main idea is to skip all tests when one test fails

4.13.6 Before Each

```
beforeEach(()
```

This function contains all code executed before each test, in this case the features provided by this function includes the control of the workflow

4.13.7 After Each

```
afterEach(()
```

This function contains all code executed after each test, in this case the features provided by this function includes the control of the workflow

4.13.8 App Tester

```
describe('App Tester', ()
```

This test is intended to validate the correctness of the tester application by execute the init, disabled, enabled, all bs_themes and all css_themes and validate with the expected screenshot

4.13.9 Action Init

```
test('Action Init', async ()
```

This part of the test tries to initialize the tester screen by provide a valid credentials and loads the tester application

4.13.10 Action Disabled

```
test('Action Disabled', async ()
```

This part of the test tries to disable all widgets

4.13.11 Action Enabled

```
test('Action Enabled', async ()
```

This part of the test tries to enable all widgets

4.13.12 List of bs_themes

```
const bs_themes = ['light', 'dark', 'auto'];
```

4.13.13 Action Bs Theme

```
test.each(bs_themes)('Action Bs Theme %s', async theme
```

This part of the test tries to set the differents bs_themes

4.13.14 List of css_themes

```
const css_themes = ['default', 'cerulean', 'cyborg', 'darkly', 'flatly', '
    journal', 'litera', 'lumen',
```

4.13.15 Action Css Theme

```
test.each(css_themes)('Action Css Theme %s', async theme
```

This part of the test tries to set the differents css_themes

4.14 Token unit tests

```
test_token.js
```

This file contains the token unit tests

4.14.1 Load all needed files of the project

```
const files = `core,storage,token`.split(',');
```

4.14.2 berofeEach used in this test

```
beforeEach(()
```

4.14.3 afterEach used in this test

```
afterEach(()
```

4.14.4 saltos.token

```
test('saltos.token', ()
```

This function performs the test of the token functions

4.15 Window unit tests

```
test_window.js
```

This file contains unit tests for window management functionality including window opening/closing and cross-tab communication

4.15.1 Load all needed files of the project

```
const files = `core,storage,window`.split(',');
```

4.15.2 Reset mocks before each test

```
beforeEach(()
```

Ensures all Jest mocks are reset before each test case runs

4.15.3 Restore mocks after each test

```
afterEach(()
```

Ensures all Jest mocks are restored to their original implementations after each test case completes

4.15.4 Test suite for window open/close functionality

```
describe('saltos.window.open/close', ()
```

Contains tests for opening new windows with different URL types and closing the current window

4.15.5 Setup before each test in this suite

```
beforeEach(()
```

Mocks window.open and window.close functions

4.15.6 Test opening app URLs

```
test('should call window.open with the app prefix', ()
```

Verifies that app URLs are properly prefixed with .#/ when opening new windows

4.15.7 Test opening HTTP URLs

```
test('should call window.open with the http prefix', ()
```

Verifies that HTTP URLs are passed through unchanged when opening new windows

4.15.8 Test opening HTTPS URLs

```
test('should call window.open with the https prefix', ()
```

Verifies that HTTPS URLs are passed through unchanged when opening new windows

4.15.9 Test unsupported URL protocols

```
test('should throw an error when call window.open with non supported
    protocol', ()
```

Verifies that attempting to open URLs with unsupported protocols throws an error

4.15.10 Test window closing

```
test('should call window.close', ()
```

Verifies that the close function properly calls window.close

4.15.11 Test suite for window event listeners

```
describe('saltos.window.listeners', ()
```

Contains tests for cross-tab communication functionality including setting listeners and sending events between tabs

4.15.12 Test setting event listeners

```
test('set_listener should add a listener for a specific event', ()
```

Verifies that listeners can be registered for specific events

4.15.13 Test removing event listeners

```
test('unset_listener should remove a listener for a specific event', ()
```

Verifies that listeners can be removed for specific events

4.15.14 Test sending events to current tab

```
test('send should trigger the listener in the same tab when scope is "me"',
     ()
```

Verifies that events with "me" scope only trigger callbacks in the current tab

4.15.15 Test sending events to other tabs

```
test('send should trigger the listener in other tabs when scope is "other
   "', ()
```

Verifies that events with "other" scope trigger callbacks in other tabs through localStorage events

4.15.16 Test sending events to all tabs

```
test('send should trigger the listener in all tabs when scope is "all"', ()
```

Verifies that events with "all" scope trigger callbacks in all tabs including the current one

4.15.17 Test sessionStorage event filtering

```
test('storage event listener should not trigger if the event is not from
   localStorage', ()
```

Verifies that events from sessionStorage don't trigger the cross-tab communication callbacks

4.15.18 Test event key filtering

```
test('storage event listener should not trigger if the key does not match',
    ()
```

Verifies that events with incorrect keys don't trigger the cross-tab communication callbacks

4.15.19 Test unregistered event filtering

```
test('storage event listener should not trigger if the event name is not in
    listeners', ()
```

Verifies that events for unregistered event names don't trigger any callbacks