

---

# **LiveObs Selenium Documentation**

***Release 1.10***

**Colin Wren, Nayira Sanchez, Daniel Metcalfe, Stuart Gray**

**Feb 09, 2018**



---

## Contents:

---

<b>1</b>	<b>Installation</b>	<b>1</b>
<b>2</b>	<b>Usage</b>	<b>3</b>
2.1	Page Object Models . . . . .	3
2.2	Utility Functions . . . . .	19
<b>3</b>	<b>Indices and tables</b>	<b>21</b>
	<b>Python Module Index</b>	<b>23</b>



# CHAPTER 1

---

## Installation

---

To get started using the LiveObs Selenium library you first need to include it in your project. You can do this adding the following to your `requirements.txt`:

```
git+git://github.com/BJSS/BJSS_liveobs_selenium.git@OPTIONAL_VERSION_NUMBER
```



Once you've got the library installed you can then import the page object models to interact with the LiveObs GUI.

Example:

```
from liveobs_ui.page_object_models.login_page import LoginPage

def login_into_liveobs(username, password):
    """
    Log into the LiveObs Mobile Frontend
    :param username: Username of user to log in with
    :param password: Password of the user to log in with
    """
    login_page = LoginPage(self.client) # self.client must be a webdriver
    login_page.login(username, password)
```

You shouldn't need to import the element selectors as these are intended for use within the page object models themselves. If you do need to use the element selectors then it's suggested that you create a pull request with the functionality to add this back into the liveobs\_selenium library.

## 2.1 Page Object Models

Page Object Models provide methods to interact with the pages they represent for instance using the Data Entry Page Object Model provides methods to fill out the observation form.

All Mobile Page Object Models are subclasses of BaseMobilePage which provides methods for going to the various mobile pages as well as common interaction patterns such as clicking a button and waiting for a particular element to become visible/invisible to verify the intended action has worked.

### 2.1.1 Login Page

The login page is the first page that mobile users see when they use LiveObs, they are prompted to enter their username and password (either supplied to them by a more privileged user or via LDAP).

For test and development purposes there may be multiple databases running on the one LiveObs instances, if this is the case then there will be a dropdown that allows for the selection of the database to log into.

#### Class Methods:

**class** `liveobs_ui.page_object_models.common.login_page.LoginPage(driver)`

Class that handles interaction with the Login Page

**get\_login\_error\_message()**

Get error message shown after an unsuccessful login

**Returns** Error message if present

**Return type** str or None

**is\_login\_page()**

Verify that we are indeed on the login page

**Returns** if current page is the login page

**login(username, password, database='nhclinical', desktop=False)**

Fill out the login form and press the submit button

#### Parameters

- **username** – Username to login with
- **password** – Password for the username supplied
- **database** – Name of the database to log into if the database selection element is visible
- **desktop** – If the login is via desktop (as different field name)

### 2.1.2 Mobile Pages

There are a few common patterns for the mobile. These involve navigating around the app and verifying the page the user is on.

#### Data Entry Page

The Data Entry Page allows the user to enter data for the observation or escalation task they are doing. It has a floating header that allows the user to access information on the patient while filling out the form.

The page will have some inputs if needed to complete the form but this is not always the case however there will always be at least one button that will allow the user to complete or cancel the task they are doing.

#### Class Methods:

**class** `liveobs_ui.page_object_models.mobile.data_entry_page.DataEntryPage(driver)`

Class that handles interacting with the data entry form

**cancel\_form()**

Press the cancel button on the form (if present)



**close\_full\_patient\_info()**  
Close the full screen patient information popup

**close\_patient\_info()**  
Close the patient information popup, opened by pressing the button in the floating header

**confirm\_submit\_scored\_ob()**  
Confirm the 'Submit Score of X for Y, Z' modal that pops up after first pressing the submit button for scored observations.

**get\_clinical\_risk\_in\_popup(*cli\_risk\_expected*)**  
Locates and returns the risk from an observation in the confirmation popup :param cli\_risk\_expected: value expected to be returned :return: string in specified element

**get\_data\_model\_from\_form()**  
Get the data model for the form so we can use this to determine what to do

**static locate\_attribute\_path(*field\_input*)**  
Identify the class of the input field and return the specific locator for their attributes

**Parameters** *field\_input* – the general locator for the field

**Returns** locator for the attribute in the DOM

**open\_full\_patient\_info()**  
Open the full screen patient information popup

**open\_patient\_info()**  
Open the patient information popup in the floating header

**submit\_cancellation\_reason(*cancel\_reason*)**  
Select the specified cancellation reason and submit the modal to cancel the task

**Parameters** *cancel\_reason* – Name of reason to select

**submit\_form()**  
Try to press the submit button on the form. Otherwise scroll down and try again. (Scroll implemented to deal with EOBS-1864)

**take\_escalation\_task()**  
Take the escalation task presented to the user once an observation that has escalation tasks is submitted

**verify\_obs\_field\_displayed(*obs\_field*)**  
Locates a specific field in an observation form and verifies its visibility

**Parameters** *obs\_field* – the observation field to look for

**verify\_obs\_field\_not\_displayed(*obs\_field*)**  
Locate a specific field in an observation form and verifies it is not visible

**Parameters** *obs\_field* – the observation field to look for

**verify\_obs\_form\_displayed(*obs\_type*)**  
Finds the data-type attribute for the currently open observation form and compares the result to the expected data-type as per the obs\_type selected

**Parameters** *obs\_type* – the observation selected

## List Page

The list page is a base class for the task and patient list as these use the same template to render their respective lists. This class handles selecting items in the list as well as verifying the text in the list items.

### Class Methods:

**class** `liveobs_ui.page_object_models.mobile.list_page.ListPage` (*driver*)  
Class that handles generic list page functionality

**get\_list\_item** (*text\_to\_find*)  
Get a specific item from the list

**Parameters** *text\_to\_find* (*str*) – Text that should be in the list item to be found

**Returns** The list item if found or None

**get\_list\_items** ()  
Get a list of the items in the task / patient list

**Returns** A list of a tags from the list

**Return type** list

**static get\_patient\_from\_item** (*list\_item*)  
Get the patient's name from list item

**Parameters** *list\_item* – WebElement for List Item

**Returns** name of patient the list item is about

**static get\_task\_info\_from\_item** (*list\_item*)  
Get the .taskInfo element from the list item

**Parameters** *list\_item* – WebElement for List Item

**Returns** text inside .taskInfo element of List Item

**open\_item** (*list\_item*)  
Open (click on) the supplied list item and ensure that the page has been successfully loaded

**Parameters** *list\_item* – List Item to click on

**Returns** True if successfully opened item, False if not

**select\_random\_patient** ()  
Finds a random patient in the patient list and selects it

**Returns** selects and opens patient

### Modal Page

As a modal is a change in context there is a separate page object model to cover this. Having a page object model for this also helps with code reuse.

### Class Methods:

**class** `liveobs_ui.page_object_models.mobile.modal_page.ModalPage` (*driver*)  
Class that handles interacting with Modals

**click\_modal\_option** (*modal, option\_title*)  
Locate the option in the modal options and click it

**Parameters**

- **modal** – Modal to find option in

- **option\_title** – Title of the button to click

**close\_modal** (*modal*)

Close supplied modal

**Parameters** **modal** – Modal object (.dialog) to close

**get\_cover\_for\_modal** (*modal*)

Get the cover that is associated with the modal

**Parameters** **modal** – Modal to find cover for

**Returns** cover object

**static get\_modal\_content** (*modal*)

Get the text content of the supplied modal

**Parameters** **modal** – Modal to get content of

**Returns** Text content of modal

**static get\_modal\_options** (*modal*)

Get the option buttons for the supplied modal

**Parameters** **modal** – Modal to find option buttons for

**Returns** List of button elements

**static get\_modal\_title** (*modal*)

Get the title text for the supplied modal

**Parameters** **modal** – Modal to get title for

**Returns** Text content of the title element

**get\_open\_modals** ()

Get the currently open modals on the page

**Returns** list of modal objects

**static select\_reason\_in\_modal** (*modal, value\_to\_select*)

For modals with select boxes (cancel reason, partial reason) select the supplied value

**Parameters**

- **modal** – Modal with select box
- **value\_to\_select** – Value to select in select box

## Patient Page

The Patient Page shows the user the observation information about the patient they have selected. From this screen they can select an observation to carry out ad-hoc, change the observation data they are looking at and change the rapid tranquilisation status of the patient.

### Class Methods:

**class** `liveobs_ui.page_object_models.mobile.patient_page.PatientPage` (*driver*)

Class that handles interacting with patient graphs and ad-hoc observation menu

**close\_adhoc\_menu** ()

Close the ad-hoc observation menu

**get\_observation\_in\_list** (*observation\_name*)

Find the specified observation and open the observation form

**Parameters** **observation\_name** – Name of the observation to open

**Returns** Boolean. Returns if the page is the intended observation

**get\_observation\_in\_menu** (*observation\_name*)

Select the specified observation from the ad-hoc observation menu

**Parameters** **observation\_name** (*str*) – Name of the observation to select

**Returns** WebElement for that observation

**get\_observation\_menu\_items** ()

Get a list of the ad-hoc observations in the ‘Take Observation’ menu :return: list of WebElements

**open\_adhoc\_menu** ()

Press the ‘Take Observation’ button to open the ad-hoc observation list

**open\_observation\_form** (*observation\_name*)

Open the ad-hoc observation menu, find the specified observation and open the observation form

**Parameters** **observation\_name** – Name of the observation to open

**Returns** Boolean. Returns if the page is the intended observation

**class** `liveobs_ui.page_object_models.mobile.mobile_common.BaseMobilePage` (*driver*)

Base class to initialise the base page that will be called from all pages

**find\_button\_to\_select** (*button\_name*)

Find a button in the page by name

**Returns** the button element on the page

**go\_to\_patient\_list** ()

Navigate to the patient list

**go\_to\_task\_list** ()

Navigate to the task list

**is\_patient\_list** ()

Are we on the patient list?

**Returns** If current URL is the patient list URL

**Return type** bool

**is\_task\_list** ()

Are we on the task list?

**Returns** If current URL is the task list URL

**Return type** bool

**logout** ()

Log out of the app

## 2.1.3 Desktop Pages

As the desktop for LiveObs uses Odoo’s webclient there are a number of common page objects that can be interactive. These are the top and left navigations, the search box and the view switch buttons.

## Account Administration Page

The Account Administration page shows the users in the system. This allows users with the correct permissions to change these user's roles in the system.

```
class liveobs_ui.page_object_models.desktop.account_administration.AccountAdministrationPage
    Interaction with the Account Administration page

    filter_on_hca()
        Filter the list to show HCA users

    filter_on_nurse()
        Filter the list to show Nurse users

    filter_on_shift_coordinators()
        Filter the list to show Shift Coordinators

    go_to_account_administration()
        Navigate the user to the Account Administration page

    select_user(user_name)
        Click the select box for the user in the list

        Parameters user_name – Name of the user to select

    select_users(user_names)
        Select multiple users in the list

        Parameters user_names – List of users to select
```

## Acuity Board Page

The Acuity Board Page shows a kanban view where patients are sorted into the columns based on their ward or their clinical risk. Users can also filter the items in the view based on clinical risk or by NEWS trend.

```
class liveobs_ui.page_object_models.desktop.acuity_board.AcuityBoardPage(driver)
    Interact with the Acuity Board

    go_to_the_acuity_board()
        Navigate to the Acuity Board

    wait_for_data_refresh()
        Wait for the kanban data to refresh
```

## Allocation Wizard Interaction

The allocation wizard allows for the assignment of a nurse and multiple HCAs to a bed location.

```
class liveobs_ui.page_object_models.desktop.allocation_wizard.AllocationWizard(driver)
    Interacting with the Allocation Wizard. The Allocation Wizard is presented to the user at the 4th stage of the
    Nursing Shift Change Wizard and the 2nd stage of the Staff Re-allocation Wizard.

    get_currently_open_modal()
        Get the currently open Allocation Modal

    save_wizard()
        Save the allocation in the wizard

    set_allocation(nurse, hca_list)
        Set the allocation for the wizard
```

**Parameters**

- **nurse** – Nurse to set for the allocation
- **hca\_list** – List of HCAs to set for the allocation

**set\_hcas** (*hca\_list*)

Enter the HCAs for the allocation into the HCAs input box

**Parameters** **hca\_list** – List of HCA user names we want to add to the allocation**set\_nurse** (*nurse\_name*)

Enter the nurse for the allocation into the nurse input box

**Parameters** **nurse\_name** – Name of the nurse we want to add to the allocation**Common Form View Interactions**

There are common actions that apply to the form view. These involve getting action buttons and tabs.

**class** `liveobs_ui.page_object_models.desktop.form_view_common.BaseFormViewPage` (*driver*)

Interaction with form view

**get\_actionbar\_button\_by\_name** (*button\_name*)

Get a button that sits in the form's actionbar that has the specified name

**Parameters** **button\_name** – Name of button to get**Returns** Button element**get\_actionbar\_buttons** ()

Get the buttons that sit in the form's actionbar

**Returns** List of button elements**get\_tab\_by\_name** (*tab\_name*)

Get tab in form view with specified name

**Parameters** **tab\_name** – Name of the tab to get**Returns** tab element**get\_tabs** ()

Get tabs on form view

**Returns** List of tab elements**wait\_for\_form\_view\_to\_load** ()

Wait until the form container has loaded, useful for ensuring that nothing is executed until the view has loaded

**Common Kanban View Interaction**

There are a number of common interaction patterns we need to do for kanban views. These involve getting a list of kanban cards and opening them.

**class** `liveobs_ui.page_object_models.desktop.kanban_view_common.BaseKanbanViewPage` (*driver*)

Interaction with Kanban views

**get\_kanban\_card\_by\_name** (*search\_string*)

Get a kanban card that contains the specified search string

**Parameters** **search\_string** – String to find in kanban card content

**Returns** Kanban card element

**get\_kanban\_cards** ()

Get all the kanban cards in the current view

**Returns** List of kanban card elements

**open\_kanban\_card** (*card*)

Open the supplied kanban card

**Parameters** *card* – Element for kanban card to open

**open\_kanban\_card\_with\_name** (*search\_string*)

Locate and click the kanban card that contains the specified search string

**Parameters** *search\_string* – String to find with kanban card content

**wait\_for\_kanban\_view\_to\_load** ()

Wait until the kanban container has loaded, useful for ensuring that nothing is executed until the view has loaded

## Common List View Interaction

There are a number of common interaction patterns we need to do for list views. These involve getting the list of items in the list view and getting a specific list item.

**class** `liveobs_ui.page_object_models.desktop.list_view_common.BaseListViewPage` (*driver*)

Common functionality for list view pages

**get\_list\_item\_by\_name** (*search\_string*)

Get a specific list item for the supplied string

**Parameters** *search\_string* – String to find in row

**Returns** element for list item with string it in

**get\_list\_items** ()

Get a list of items in the list view

**Returns** list of list items

**open\_list\_item** (*list\_item*)

Open the supplied list item

**Parameters** *list\_item* – List Item to open

**open\_list\_item\_by\_name** (*search\_string*)

Find the row in the list for the *search\_string* and open the item

**Parameters** *search\_string* – string to find in the list item

**wait\_for\_list\_view\_to\_load** ()

Wait until the list container has loaded, useful for ensuring that nothing is executed until the view has loaded

## Common Modal View Interactions

There are a number of common interaction patterns with modals such as closing them or clicking the buttons at the bottom of the modal.

**class** `liveobs_ui.page_object_models.desktop.modal_view_common.BaseModalPage` (*driver*)

Common interaction patterns with modal dialogs

**click\_modal\_button\_by\_name** (*modal*, *name*)

Click a button in modal's footer with a specified name

**Parameters**

- **modal** – Modal to click button in
- **name** – Name of button to click

**close\_modal** (*modal*)

Close the supplied modal

**Parameters** **modal** – Modal to close

**get\_currently\_open\_modal** ()

Get the currently open modal on the screen

**Returns** Modal element

**get\_modal\_button\_by\_name** (*modal*, *name*)

Get a button in the modal's footer with a specified name

**Parameters**

- **modal** – Modal to find button in
- **name** – Name of button

**Returns** button element

**static get\_modal\_buttons** (*modal*)

Get the buttons in the supplied modal's footer

**Parameters** **modal** – Modal to get buttons for

**Returns** list of button elements

**static get\_modal\_title** (*modal*)

Get the title of the supplied modal

**Parameters** **modal** – Modal to get title for

**Returns** Title for supplied modal

**Return type** str

## Nursing Shift Change Wizard Interaction

The Nursing Shift Change Wizard is a modal wizard that guides the user through four stages, ward selection, deallocation, roll call and allocation.

**class** liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.**NursingShiftChangeWizard**

Interacting with the Nursing Shift Change Wizard and it's different stages

**allocate** (*allocation\_map*)

Allocate users to beds

**Parameters** **allocation\_map** – dictionary containing mapping of users to beds

**collect\_current\_allocation** ()

Get the current user(s) -> bed allocation from the de-allocation screen

**Returns** list of dictionaries representing beds and the different nurse and HCA users allocated to that bed



**confirm\_stage()**  
Confirm the data in the current stage and move to the next one

**get\_confirm\_button()**  
Get the button to take the user to next stage of the wizard

**Returns** WebElement for button

**go\_to\_allocation()**  
Go to the allocation stage

**go\_to\_deallocate\_stage()**  
Go to the deallocation stage

**go\_to\_end()**  
Go to the end of the wizard

**go\_to\_roll\_call()**  
Go to the roll call stage

**roll\_call(*user\_list*)**  
Add all users that were originally on ward back into ward

**Parameters** *user\_list* – list of names to enter into input so these users can then be allocated to the beds in the ward

**select\_ward(*ward\_name*)**  
Get currently logged in user's ward and select it in dropdown

**Parameters** *ward\_name* – Name of ward to select for Nursing Shift change

**set\_ward\_for\_shift\_change(*ward\_name*)**  
Set ward in ward selection box in Nursing Shift Change Wizard

**Parameters** *ward\_name* – Name of the ward to set

**start\_nursing\_shift\_change()**  
Click the Nursing Shift Change menu item to start the wizard

## Overdue Tasks Page

The Overdue Tasks lists tasks that have a deadline on or before the current time. This page allows users to see what tasks need to be done urgently on the ward(s) they are responsible for.

**class** `liveobs_ui.page_object_models.desktop.overdue_tasks.OverdueTasksPage` (*driver*)  
Interaction with the Overdue Tasks Page

**go\_to\_overdue\_tasks()**  
Navigate the user to the Overdue Tasks page

**group\_by\_activity()**  
Group the list items by activity

**group\_by\_assigned\_to()**  
Group the list items by the user they are assigned to

**group\_by\_parent\_location()**  
Group the list items by parent location

**group\_by\_patient()**  
Group the list items by patient

## Patient Record Page

The Patient Record page shows the demographics for the patient and allows the user to access the observation data for the patient too. There is also functionality to move the patient around the ward, print a PDF of the patients data and stop the observations for the patient.

```
class liveobs_ui.page_object_models.desktop.patient_record.PatientRecordPage (driver)
    Interaction with the patient record

    go_to_previous_spell()
        Go to the patient's previous clinical spell

    open_move_patient_wizard()
        Open the Move Patient wizard

    open_print_report_wizard()
        Open the Print Report wizard

    open_stop_observations_wizard()
        Open the Stop Observations wizard

    open_swap_bed_wizard()
        Open the Swap Bed wizard

    open_wizard_with_name (button_name)
        Open wizard via button name

        Parameters button_name – Name of button to press
```

## Patients By Ward Page

the Patients By Ward Page shows a list of patients grouped by ward. The user can then click on buttons to open the patient's EWS Chart or EWS data and they can open the record for the patient.

```
class liveobs_ui.page_object_models.desktop.patients_by_ward.PatientsByWardPage (driver)
    Interaction with Patients By Ward Page

    go_to_patients_by_ward()
        Navigate the user to the Patients By Ward page

    open_patient_chart_popup (patient_name)
        Find the row in the list for the supplied patient then open the chart popup

        Parameters patient_name – Name of patient who's chart we want to open

    open_patient_list_popup (patient_name)
        Find the row in the list for the supplied patient then open the list popup

        Parameters patient_name – Name of the patient who's list we want to open

    wait_for_data_refresh()
        Wait for the kanban data to refresh

    wait_for_list_view_to_load()
        Override of the function that waits for the list view to load to take into consideration the fact that Patients
        By Ward is a grouped list view
```

## Patients Without Bed

The Patients Without Bed page lists patients who are currently associated with the ward but not placed in a bed. The user will then select a patient and place them into a bed via this page.

**class** `liveobs_ui.page_object_models.desktop.patients_without_bed.PatientsWithoutBedPage` (*driver*)  
Interaction with Patients Without Bed Page

**go\_to\_patients\_without\_bed**()  
Navigate the user to the Patients Without Bed page

**open\_bed\_placement\_by\_name** (*patient\_name*)  
Find the row in the list for the supplied patient then open the bed placement popup

**Parameters** *patient\_name* – Name of patient who we want to place in bed

**open\_bed\_placement\_popup** (*patient*)  
Open the bed placement popup for the supplied patient element

**Parameters** *patient* – Patient to click bed placement button on

## Recently Discharged / Transferred Pages

The Recently Discharged and Recently Transferred pages show the user patients who have been discharged or transferred from the ward they are allocated to in the last 72 (by default) hours.

The number of hours that records are shown for is configurable by the System Administrator.

**class** `liveobs_ui.page_object_models.desktop.recently_discharged_transferred.RecentlyDischarged`  
Interactions for the Recently Discharged and Recently Transferred pages. These pages share a common view so are bundled together.

**go\_to\_recently\_discharged**()  
Navigate the user to the Recently Discharged page

**go\_to\_recently\_transferred**()  
Navigate the user to the Recently Transferred page

## Wardboard Common Functionality

The Wardboard is a view on the patients in the ward the user is allocated to. This can be as a kanban, list or form view.

Common functionality for the page involves filtering and grouping the data in the views by clinical risk, score trend and location.

**class** `liveobs_ui.page_object_models.desktop.wardboard_common.WardBoardPage` (*driver*)  
Interaction patterns for wardboard page

**group\_by\_clinical\_risk**()  
Group the Acuity Board by Clinical Risk

**group\_by\_ward**()  
Group the Acuity Board by Ward

**select\_deteriorating\_trend\_filter**()  
Select the Deteriorating Trend filter

**select\_high\_risk\_filter**()  
Select the High Risk Filter

**select\_improving\_trend\_filter**()  
Select the Improving Trend filter

**select\_low\_risk\_filter**()  
Select the Low Risk Filter

```
select_medium_risk_filter()  
    Select the Medium Risk Filter  
  
select_no_risk_filter()  
    Select the No Risk Filter  
  
select_unchanged_trend_filter()  
    Select the Unchanged Trend filter  
  
wait_for_data_refresh()  
    Wait until the data on the page is refreshed (via JS timeout)
```

## Workload Page

The workload page shows the currently open tasks for patients on the ward the user is allocated to sorted into time ranges and present in a kanban like view.

```
class liveobs_ui.page_object_models.desktop.workload.WorkloadPage (driver)  
    Interaction with the workload page  
  
    filter_by_news()  
        Filter the view by NEWS observations  
  
    filter_by_non_news()  
        Filter the view by non-NEWS tasks  
  
    go_to_workload()  
        Navigate the user to the workload page  
  
class liveobs_ui.page_object_models.desktop.desktop_common.BaseDesktopPage (driver)  
    Base class to initialise the base page that will be called from all pages  
  
    change_to_form_view()  
        Change the Form view mode  
  
    change_to_kanban_view()  
        Change the Kanban view mode  
  
    change_to_list_view()  
        Change the list view mode  
  
    change_view_mode (view_mode)  
        Click the supplied view_mode button  
  
        Parameters view_mode – View mode to select  
  
    click_breadcrumb (breadcrumb)  
        Click on the supplied breadcrumb and verify that it is no longer on the page  
  
        Parameters breadcrumb – Breadcrumb element to click  
  
    close_search_options_draw()  
        Close the search options draw on the current page  
  
    get_breadcrumbs()  
        Get the breadcrumbs that allow to go back up a page in the navigation  
  
        Returns list of breadcrumb elements  
  
    static get_menu_item_text (menu_item)  
        Returns the text attribute of an element on an item in the left menu section on the desktop app  
  
        Parameters menu_item – WebElement from where to get the text element
```

**Returns** The text within the WebElement

**Return type** unicode

**get\_menu\_items\_list** ()

Returns a list of all the items in the left menu section on the desktop app

**Returns** A list of all the WebElement items

**Return type** List of WebElements

**go\_to\_next\_record\_in\_list** ()

Click the next record button in the list of records. This is shown on the form view

**go\_to\_page** (*page\_title*)

Go to the supplied page in the left hand menu based on the title of the page

**Parameters** **page\_title** – Title of the page to go to

**go\_to\_previous\_page** ()

Go back to the page used to get to the patient record in the breadcrumbs

**go\_to\_previous\_record\_in\_list** ()

Click the previous record button in the list of records. This is shown on the form view

**open\_search\_options\_draw** ()

Open the search options draw on the current page

**perform\_search** (*search\_query*, *search\_type=None*)

Using the searchview conduct a search with the supplied search query

**Parameters**

- **search\_query** – Query to input into the search box
- **search\_type** – Type of search (uses Odoo Search autocomplete)

**select\_filter** (*filter\_name*)

Open up the search options draw and select a filter based on filter name

**Parameters** **filter\_name** – Name of the filter to select

**select\_group\_by** (*group\_by\_name*)

Open the search options draw and select a group by based on the supplied name

**Parameters** **group\_by\_name** – Name of the group by option to select

## 2.1.4 Common Methods:

**class** `liveobs_ui.page_object_models.common.base_liveobs_page.BaseLiveObsPage` (*driver*)  
Base Page for all LiveObs Interaction

**click\_and\_verify\_change** (*el\_to\_click*, *el\_to\_verify*, *hidden=False*)

Wrapper around clicking an element and then waiting for a change in the page and verifying said change by ensuring an element is visible

**Parameters**

- **el\_to\_click** – Element to click to induce change
- **el\_to\_verify** – Element to look for to verify change
- **hidden** – Should check for if element is now hidden or not

**static element\_is\_displayed** (*element\_object*)

Verify that an element is visible on the page

**Parameters** **element\_object** – the object or element to verify

**Returns** either True/False for the element being displayed

**enter\_input\_value** (*element, value, autocompleted=False, dropdown=False*)

Enter a value into an input field

**Parameters**

- **element** – webelement we want to add value to
- **value** – Value we want to add
- **autocompleted** – If should wait for autocomplete box to show before continuing
- **dropdown** – If should wait for the different type of dropdown to show before continuing

**enter\_many2one\_tag\_value** (*element, value*)

Add value to many2one tag input then verify that the tag has been added

**Parameters**

- **element** – many2one tag input element
- **value** – value to add to the many2one input

**fill\_input\_field** (*element, value*)

Enter a value into an input field

**Parameters**

- **element** – webelement we want to add value to
- **value** – Value to enter into input

**fill\_out\_form** (*data*)

Using the provided list fill out the form

Expected Data Format:

```
{
  'name': 'respiration_rate',
  'value': '18',
  'type': 'textbox'
}
```

**Parameters** **data** – list of dictionaries that contain field names, value to enter and field type

**fill\_select\_field** (*element, value*)

Select an option in a select field

**Parameters**

- **element** – Select we want to sort out's webelement
- **value** – Value to select in the select field

**get\_element\_by\_lookup** (*element\_name*)

Get an element by the lookup :param element\_name: look up term to match :return: webelement

**static get\_locator** (*element*)

Get locator for element

**Parameters** `element` – webelement we want to generate locator for

**Returns** Locator for supplied element

**static** `get_state_of_el` (*element, attribute, state*)

Return if attribute on element is a particular state :param element: Webelement :param attribute: attribute to find :param state: expected value :return: bool

**verify\_page\_loaded** (*app\_view*)

Waits for a specific element to be visible on the screen the ‘desktop’ or ‘mobile’ app to ensure the page has loaded before progressing

**Parameters** `app_view` – Either ‘mobile’ or ‘desktop’

**wait\_for\_element** (*element\_selector, hidden=False, wait\_time=None*)

Wrapper around WebDriverWait to wait for specified element to become visible

**Parameters**

- **element\_selector** (*tuple*) – Element Selector tuple
- **hidden** (*bool*) – Check if element is hidden or not
- **wait\_time** (*int*) – Custom time to wait

## 2.2 Utility Functions

There are a number of utility functions that help with running selenium against the UI.

### 2.2.1 Converters

Converters for use with Selenium

`liveobs_ui.utils.converters.convert_list_of_data_dicts` (*values\_list*)

Convert a list of dicts that contain data to fill out a form

**Parameters** `values_list` – List of dictionaries that contain form data

**Returns** list of dictionaries with converted values

`liveobs_ui.utils.converters.convert_numbers_in_dict` (*values\_dict*)

Convert any values in dictionary that are values to string

**Parameters** `values_dict` – dictionary of values used to submit

**Returns** dictionary with all numerical values converted to a string





## CHAPTER 3

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



I

`liveobs_ui.utils.converters`, [19](#)



## A

AccountAdministrationPage (class in `liveobs_ui.page_object_models.desktop.account_administration`), 9  
 AcuityBoardPage (class in `liveobs_ui.page_object_models.desktop.acuity_board`), 9  
 allocate() (`liveobs_ui.page_object_models.desktop.nursing_shift_change_wizard.NursingShiftChangeWizard` method), 12  
 AllocationWizard (class in `liveobs_ui.page_object_models.desktop.allocation_wizard`), 9  
 change\_to\_form\_view() (`liveobs_ui.page_object_models.desktop.desktop_common.BaseDesktopPage` method), 16  
 change\_to\_kanban\_view() (`liveobs_ui.page_object_models.desktop.desktop_common.BaseDesktopPage` method), 16  
 change\_to\_list\_view() (`liveobs_ui.page_object_models.desktop.desktop_common.BaseDesktopPage` method), 16  
 change\_view\_model() (`liveobs_ui.page_object_models.desktop.desktop_common.BaseDesktopPage` method), 16  
 click\_and\_verify\_change() (`liveobs_ui.page_object_models.common.base_liveobs_page.BaseLiveObsPage` method), 17  
 click\_breadcrumb() (`liveobs_ui.page_object_models.desktop.desktop_common.BaseDesktopPage` method), 16

## B

BaseDesktopPage (class in `liveobs_ui.page_object_models.desktop.desktop_common`), 16  
 BaseFormViewPage (class in `liveobs_ui.page_object_models.desktop.form_view_common`), 10  
 BaseKanbanViewPage (class in `liveobs_ui.page_object_models.desktop.kanban_view_common`), 10  
 BaseListViewPage (class in `liveobs_ui.page_object_models.desktop.list_view_common`), 11  
 BaseLiveObsPage (class in `liveobs_ui.page_object_models.common.base_liveobs_page`), 17  
 BaseMobilePage (class in `liveobs_ui.page_object_models.mobile.mobile_common`), 8  
 BaseModalPage (class in `liveobs_ui.page_object_models.desktop.modal_view_common`), 11  
 click\_modal\_button\_by\_name() (`liveobs_ui.page_object_models.desktop.modal_view_common.BaseModalPage` method), 11  
 click\_modal\_option() (`liveobs_ui.page_object_models.mobile.modal_page.ModalPage` method), 6  
 close\_adhoc\_menu() (`liveobs_ui.page_object_models.mobile.patient_page.PatientPage` method), 7  
 close\_full\_patient\_info() (`liveobs_ui.page_object_models.mobile.data_entry_page.DataEntryPage` method), 5  
 close\_modal() (`liveobs_ui.page_object_models.desktop.modal_view_common.BaseModalPage` method), 12  
 close\_modal() (`liveobs_ui.page_object_models.mobile.modal_page.ModalPage` method), 7  
 close\_patient\_info() (`liveobs_ui.page_object_models.mobile.data_entry_page.DataEntryPage` method), 5  
 close\_search\_options\_draw() (`liveobs_ui.page_object_models.desktop.desktop_common.BaseDesktopPage` method), 16  
 collect\_current\_allocation() (`liveobs_ui.page_object_models.desktop.nursing_shift_change_wizard.NursingShiftChangeWizard` method), 12  
 confirm\_stage() (`liveobs_ui.page_object_models.desktop.nursing_shift_change_wizard.NursingShiftChangeWizard` method), 12  
 confirm\_submit\_scored\_obs() (`liveobs_ui.page_object_models.mobile.data_entry_page.DataEntryPage` method), 5

## C

cancel\_form() (`liveobs_ui.page_object_models.mobile.data_entry_page.DataEntryPage` method), 4  
 confirm\_submit\_scored\_obs() (`liveobs_ui.page_object_models.mobile.data_entry_page.DataEntryPage` method), 5

convert\_list\_of\_data\_dicts() (in module get\_cover\_for\_modal() (liveobs\_ui.page\_object\_models.mobile.modal\_page.  
 liveobs\_ui.utils.converters), 19 method), 7  
 convert\_numbers\_in\_dict() (in module get\_currently\_open\_modal()  
 liveobs\_ui.utils.converters), 19 (liveobs\_ui.page\_object\_models.desktop.allocation\_wizard.Alloc  
 method), 9  
 get\_currently\_open\_modal()  
 (liveobs\_ui.page\_object\_models.desktop.modal\_view\_common.B  
 method), 12  
 get\_data\_model\_from\_form()  
 (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEnt  
 method), 5  
**D**  
 DataEntryPage (class in (liveobs\_ui.page\_object\_models.desktop.modal\_view\_common.B  
 liveobs\_ui.page\_object\_models.mobile.data\_entry\_page), method), 12  
 4 get\_data\_model\_from\_form()  
 (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEnt  
 method), 5  
**E**  
 element\_is\_displayed() (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
 static method), 17 (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.Bas  
 method), 18  
 enter\_input\_value() (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
 method), 18 get\_kanban\_card\_by\_name()  
 (liveobs\_ui.page\_object\_models.desktop.kanban\_view\_common.I  
 method), 10  
 enter\_many2one\_tag\_value() (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
 method), 18 get\_kanban\_cards() (liveobs\_ui.page\_object\_models.desktop.kanban\_view  
 method), 11  
**F**  
 fill\_input\_field() (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
 method), 18 get\_list\_item() (liveobs\_ui.page\_object\_models.mobile.list\_page.ListPage  
 method), 6  
 fill\_out\_form() (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
 method), 11 (liveobs\_ui.page\_object\_models.desktop.list\_view\_common.Bas  
 method), 11  
 fill\_select\_field() (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
 method), 11 get\_list\_items() (liveobs\_ui.page\_object\_models.desktop.list\_view\_common  
 method), 11  
 filter\_by\_news() (liveobs\_ui.page\_object\_models.desktop.workload.WorkloadPage  
 method), 6 get\_list\_items() (liveobs\_ui.page\_object\_models.mobile.list\_page.ListPage  
 method), 6  
 filter\_by\_non\_news() (liveobs\_ui.page\_object\_models.desktop.workload.WorkloadPage  
 static method), 18  
 filter\_on\_hca() (liveobs\_ui.page\_object\_models.desktop.account\_administration.AccountAdministrationPage  
 method), 9 (liveobs\_ui.page\_object\_models.common.login\_page.LoginPage  
 method), 4  
 filter\_on\_nurse() (liveobs\_ui.page\_object\_models.desktop.account\_administration.AccountAdministrationPage  
 get\_menu\_item\_text() (liveobs\_ui.page\_object\_models.desktop.desktop\_co  
 static method), 16  
 filter\_on\_shift\_coordinators() (liveobs\_ui.page\_object\_models.desktop.account\_administration.AccountAdministrationPage  
 method), 17  
 (liveobs\_ui.page\_object\_models.desktop.account\_administration.AccountAdministrationPage  
 method), 9  
 find\_button\_to\_select() (liveobs\_ui.page\_object\_models.mobile.common.BaseMobilePage  
 method), 8 (liveobs\_ui.page\_object\_models.desktop.modal\_view\_common.B  
 method), 12  
**G**  
 get\_modal\_buttons() (liveobs\_ui.page\_object\_models.desktop.modal\_view  
 static method), 12  
 get\_actionbar\_button\_by\_name() (liveobs\_ui.page\_object\_models.desktop.form\_view\_common.BaseFormViewPage  
 method), 10  
 get\_actionbar\_buttons() (liveobs\_ui.page\_object\_models.desktop.form\_view\_common.BaseFormViewPage  
 static method), 7  
 get\_breadcrumbs() (liveobs\_ui.page\_object\_models.desktop.desktop\_common.BaseDesktopPage  
 static method), 12  
 get\_modal\_title() (liveobs\_ui.page\_object\_models.desktop.modal\_view\_co  
 static method), 12  
 get\_modal\_title() (liveobs\_ui.page\_object\_models.mobile.modal\_page.Mod  
 static method), 7  
 get\_clinical\_risk\_in\_popup() (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage  
 method), 5  
 get\_observation\_in\_hist() (liveobs\_ui.page\_object\_models.mobile.patient\_page.PatientPage  
 method), 7  
 get\_confirm\_button() (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard  
 method), 13  
 get\_observation\_in\_menu()

(liveobs\_ui.page\_object\_models.mobile.patient\_page.PatientPage method), 8

(liveobs\_ui.page\_object\_models.desktop.recently\_discharged\_transferred\_page.RecentlyDischargedTransferredPage method), 15

get\_observation\_menu\_items() (liveobs\_ui.page\_object\_models.mobile.patient\_page.PatientPage method), 13

(liveobs\_ui.page\_object\_models.mobile.patient\_page.PatientPage method), 8

get\_open\_modals() (liveobs\_ui.page\_object\_models.mobile.modal\_page.ModalPage method), 7

get\_patient\_from\_item() (liveobs\_ui.page\_object\_models.mobile.list\_page.ListPage method), 6

(liveobs\_ui.page\_object\_models.desktop.acuity\_board.AcuityBoard static method), 9

get\_state\_of\_el() (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage static method), 19

(liveobs\_ui.page\_object\_models.desktop.workload.Workload static method), 16

get\_tab\_by\_name() (liveobs\_ui.page\_object\_models.desktop.form\_view\_page.FormViewPage method), 10

(liveobs\_ui.page\_object\_models.desktop.overdue\_tasks.OverdueTasks method), 13

get\_tabs() (liveobs\_ui.page\_object\_models.desktop.form\_view\_page.FormViewPage method), 10

(liveobs\_ui.page\_object\_models.desktop.overdue\_tasks.OverdueTasks method), 13

get\_task\_info\_from\_item() (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 15

(liveobs\_ui.page\_object\_models.mobile.list\_page.ListPage static method), 6

group\_by\_clinical\_risk() (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 15

group\_by\_parent\_location() (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 15

go\_to\_account\_administration() (liveobs\_ui.page\_object\_models.desktop.account\_administration\_page.AccountAdministrationPage method), 9

group\_by\_patient() (liveobs\_ui.page\_object\_models.desktop.overdue\_tasks.OverdueTasks method), 13

go\_to\_allocation() (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard method), 13

group\_by\_ward() (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 15

go\_to\_deallocate\_stage() (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard method), 13

go\_to\_end() (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard method), 13

is\_login\_page() (liveobs\_ui.page\_object\_models.common.login\_page.LoginPage method), 4

go\_to\_next\_record\_in\_list() (liveobs\_ui.page\_object\_models.desktop.desktop\_common.BaseDesktopPage method), 17

is\_patient\_list() (liveobs\_ui.page\_object\_models.mobile.mobile\_common.BaseMobilePage method), 8

is\_task\_list() (liveobs\_ui.page\_object\_models.mobile.mobile\_common.BaseMobilePage method), 8

go\_to\_overdue\_tasks() (liveobs\_ui.page\_object\_models.desktop.overdue\_tasks.OverdueTasksPage method), 13

go\_to\_page() (liveobs\_ui.page\_object\_models.desktop.desktop\_common.BaseDesktopPage method), 17

ListPage (class in liveobs\_ui.page\_object\_models.mobile.list\_page), 19

go\_to\_patient\_list() (liveobs\_ui.page\_object\_models.mobile.mobile\_common.BaseMobilePage method), 8

liveobs\_ui.utils.converters (module), 19

go\_to\_patients\_by\_ward() (liveobs\_ui.page\_object\_models.desktop.patients\_by\_ward.PatientsByWardPage static method), 14

locate\_attribute\_path() (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage method), 14

login() (liveobs\_ui.page\_object\_models.common.login\_page.LoginPage method), 4

go\_to\_patients\_without\_bed() (liveobs\_ui.page\_object\_models.desktop.patients\_without\_bed.PatientsWithoutBedPage method), 15

LoginPage (class in liveobs\_ui.page\_object\_models.common.login\_page), 4

go\_to\_previous\_page() (liveobs\_ui.page\_object\_models.desktop.desktop\_common.BaseDesktopPage method), 17

logout() (liveobs\_ui.page\_object\_models.mobile.mobile\_common.BaseMobilePage method), 8

go\_to\_previous\_record\_in\_list() (liveobs\_ui.page\_object\_models.desktop.desktop\_common.BaseDesktopPage method), 17

ModalPage (class in liveobs\_ui.page\_object\_models.mobile.modal\_page), 8

go\_to\_previous\_spell() (liveobs\_ui.page\_object\_models.desktop.patient\_record.PatientRecordPage method), 14

NursingShiftChangeWizard (class in liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard), 12

go\_to\_recently\_discharged() (liveobs\_ui.page\_object\_models.desktop.recently\_discharged\_transferred\_page.RecentlyDischargedTransferredPage method), 15

go\_to\_recently\_tranferred() (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard method), 12

## O

`open_adhoc_menu()` (liveobs\_ui.page\_object\_models.mobile.patient\_page.PatientPage method), 8  
`open_bed_placement_by_name()` (liveobs\_ui.page\_object\_models.desktop.patients\_without\_bed.PatientsWithoutBedPage method), 15  
`open_bed_placement_popup()` (liveobs\_ui.page\_object\_models.desktop.patients\_without\_bed.PatientsWithoutBedPage method), 15  
`open_full_patient_info()` (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage method), 5  
`open_item()` (liveobs\_ui.page\_object\_models.mobile.list\_page.ListPage method), 6  
`open_kanban_card()` (liveobs\_ui.page\_object\_models.desktop.kanban\_view\_common.BaseKanbanViewPage method), 11  
`open_kanban_card_with_name()` (liveobs\_ui.page\_object\_models.desktop.kanban\_view\_common.BaseKanbanViewPage method), 11  
`open_list_item()` (liveobs\_ui.page\_object\_models.desktop.list\_view\_rollcall(BaseObsUiPage method), 11  
`open_list_item_by_name()` (liveobs\_ui.page\_object\_models.desktop.list\_view\_rollcall(BaseObsUiPage method), 11  
`open_move_patient_wizard()` (liveobs\_ui.page\_object\_models.desktop.patient\_record.PatientRecordPage method), 14  
`open_observation_form()` (liveobs\_ui.page\_object\_models.mobile.patient\_page.PatientPage method), 8  
`open_patient_chart_popup()` (liveobs\_ui.page\_object\_models.desktop.patients\_by\_ward.PatientByWardPage method), 14  
`open_patient_info()` (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage method), 5  
`open_patient_list_popup()` (liveobs\_ui.page\_object\_models.desktop.patients\_by\_ward.PatientByWardPage method), 14  
`open_print_report_wizard()` (liveobs\_ui.page\_object\_models.desktop.patient\_record.PatientRecordPage method), 14  
`open_search_options_draw()` (liveobs\_ui.page\_object\_models.desktop.desktop\_common.BaseDesktopPage method), 17  
`open_stop_observations_wizard()` (liveobs\_ui.page\_object\_models.desktop.patient\_record.PatientRecordPage method), 14  
`open_swap_bed_wizard()` (liveobs\_ui.page\_object\_models.desktop.patient\_record.PatientRecordPage method), 14  
`open_wizard_with_name()` (liveobs\_ui.page\_object\_models.desktop.patient\_record.PatientRecordPage method), 14  
`OverdueTasksPage` (class in liveobs\_ui.page\_object\_models.desktop.overdue\_tasks), 16

## P

`PatientPage` (class in liveobs\_ui.page\_object\_models.mobile.patient\_page), 13  
`PatientRecordPage` (class in liveobs\_ui.page\_object\_models.desktop.patient\_record), 14  
`PatientsByWardPage` (class in liveobs\_ui.page\_object\_models.desktop.patients\_by\_ward), 14  
`PatientsWithoutBedPage` (class in liveobs\_ui.page\_object\_models.desktop.patients\_without\_bed), 14  
`perform_search()` (liveobs\_ui.page\_object\_models.desktop.desktop\_common.BaseDesktopPage method), 17  
`RecentlyDischargedBedsPage` (class in liveobs\_ui.page\_object\_models.desktop.recently\_discharged\_transfers), 15  
`rollcall()` (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard method), 13

## S

`save_wizard()` (liveobs\_ui.page\_object\_models.desktop.allocation\_wizard.AllocationWizard method), 9  
`select_high_risk_filter()` (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 15  
`select_improving_trend_filter()` (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 15  
`select_low_risk_filter()` (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 15  
`select_random_patient()` (liveobs\_ui.page\_object\_models.mobile.list\_page.ListPage method), 6  
`select_reason_in_modal()` (liveobs\_ui.page\_object\_models.mobile.modal\_page.ModalPage method), 7  
`select_unchanged_trend_filter()` (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardboardCommon method), 16



[select\\_user\(\)](#) (liveobs\_ui.page\_object\_models.desktop.account\_administration.AccountAdministrationPage  
method), 9  
[select\\_users\(\)](#) (liveobs\_ui.page\_object\_models.desktop.account\_administration.AccountAdministrationPage  
method), 9  
[select\\_ward\(\)](#) (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard  
method), 13  
[set\\_allocation\(\)](#) (liveobs\_ui.page\_object\_models.desktop.allocation\_wizard.AllocationWizard  
method), 9  
[set\\_hcas\(\)](#) (liveobs\_ui.page\_object\_models.desktop.allocation\_wizard.AllocationWizard  
method), 10  
[set\\_nurse\(\)](#) (liveobs\_ui.page\_object\_models.desktop.allocation\_wizard.AllocationWizard  
method), 10  
[set\\_ward\\_for\\_shift\\_change\(\)](#) (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard  
method), 13  
[start\\_nursing\\_shift\\_change\(\)](#) (liveobs\_ui.page\_object\_models.desktop.nursing\_shift\_change\_wizard.NursingShiftChangeWizard  
method), 13  
[submit\\_cancellation\\_reason\(\)](#) (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage  
method), 5  
[submit\\_form\(\)](#) (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage  
method), 5

## T

[take\\_escalation\\_task\(\)](#) (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage  
method), 5

## V

[verify\\_obs\\_field\\_displayed\(\)](#) (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage  
method), 5  
[verify\\_obs\\_field\\_not\\_displayed\(\)](#) (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage  
method), 5  
[verify\\_obs\\_form\\_displayed\(\)](#) (liveobs\_ui.page\_object\_models.mobile.data\_entry\_page.DataEntryPage  
method), 5  
[verify\\_page\\_loaded\(\)](#) (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
method), 19

## W

[wait\\_for\\_data\\_refresh\(\)](#) (liveobs\_ui.page\_object\_models.desktop.acuity\_board.AcuityBoardPage  
method), 9  
[wait\\_for\\_data\\_refresh\(\)](#) (liveobs\_ui.page\_object\_models.desktop.patients\_by\_ward.PatientsByWardPage  
method), 14  
[wait\\_for\\_data\\_refresh\(\)](#) (liveobs\_ui.page\_object\_models.desktop.wardboard\_common.WardBoardPage  
method), 16  
[wait\\_for\\_element\(\)](#) (liveobs\_ui.page\_object\_models.common.base\_liveobs\_page.BaseLiveObsPage  
method), 19  
[wait\\_for\\_form\\_view\\_to\\_load\(\)](#) (liveobs\_ui.page\_object\_models.desktop.form\_view\_common.BaseFormViewPage  
method), 10