runmanager

Release 0.1.0.dev147+g34e1442

labscript suite contributors

Feb 09, 2024

DOCUMENTATION

A graphical and remote interface to parameterized experiments.

CHAPTER

ONE

INTRODUCTION

Runmanager (see Fig. 1.1) is the primary interface for adjusting the behaviour of the experiment and generating shot files. Runmanager allows you to define parameters via a graphical interface and select the experiment logic file to use. The parameters are inserted into the experiment logic (as global variables) when shot generation is started via the 'Engage' button.

The parameters can take the form of any Python data type that can be stored in a hdf5 file. When a parameter is specified as a list or array, runmanager automatically detects that you wish to explore a parameter space. Runmanager will generate a separate shot for each point in the array. If multiple arrays are specified across two or more parameters, runmanager automatically takes the Cartesian product of those parameters in order to generate shots that span the entire parameter space. For more complex experiments, parameters can be linked together so they iterate in lock step by defining one or more zip groups. If multiple zip groups are defined, the parameter space explored will be the Cartesian product of each zip group and any other parameters containing an array or list.

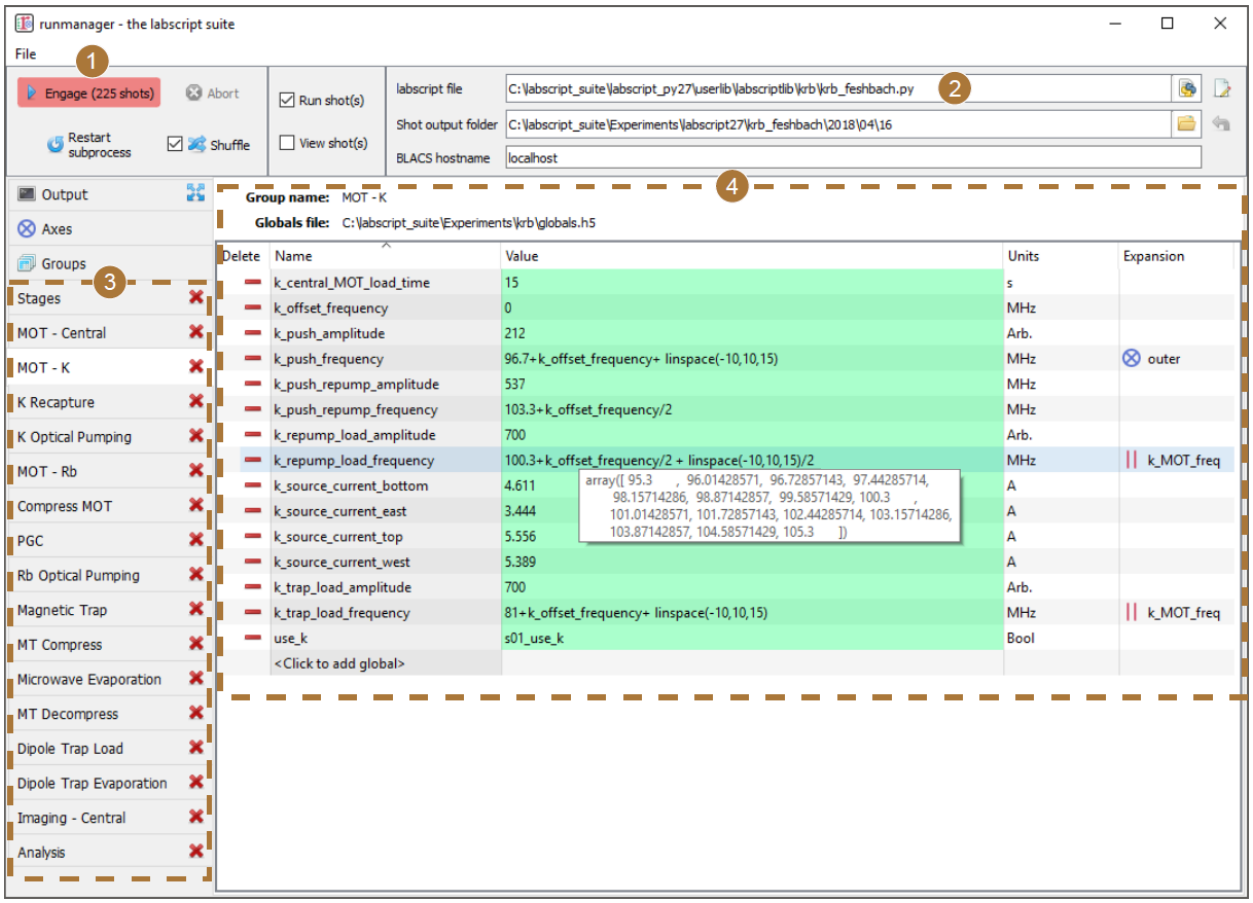

Fig. 1.1: The runmanager interface, used to generate experiment shot files (1) using the experiment logic defined using the labscript API in a Python file (2) and groups of parameters (3), for example those shown for the 'MOT - K' group (4). Further details on the runmanager interface can be found in *Usage*.

CHAPTER

TWO

USAGE

Runmanager is the primary means of defining and managing the set of experiment parameters (global variables - see §5.1.3) used in the labscript experiment logic. Runmanager also handles the creation of each hdf5 shot file, and the invocation of labscript via the execution of a user specified labscript experiment logic file.

2.1 The graphical interface

We believe that the manipulation of parameters, along with controls for producing shots, are best implemented in a graphical interface. Critical information on the current runmanager configuration, along with controls for generating new shots, are located in a always visible toolbar at the top of the runmanager interface. These comprise (as labelled in Fig. 2.1):

1. The engage button: This begins production of the appropriate number of shot files. The number of shot files that will be produced is displayed prominently in the button text so that any mistakes made when defining the parameter space scan can be quickly corrected prior to beginning shot generation. This button can also be 'clicked' via the F5 key on a keyboard.
2. The abort button: This stops the production of shot files prematurely.
3. The restart subprocess button: Primarily for debugging and for use during labscript development, this button restarts the subprocess that manages the execution of the labscript experiment logic file, which in turn generates and stores hardware instructions inside the hdf5 file (see *Shot creation*).
4. The shuffle checkbox: This checkbox controls the global setting for whether parameter space scans are shuffled or not. This is a tri-state checkbox (all-some-none) displaying the current shuffle state on the axes tab. Clicking the checkbox will overwrite the state of each entry on the axes tab with the new state of the global checkbox. For more details, see *Parameter space scans*.
5. The run shots checkbox: If ticked prior to clicking the engage button, shot files will be sent immediately to the BLACS queue once the hardware instructions have been generated by labscript.
6. The view shots checkbox: If ticked prior to clicking the engage button, shots will be sent to runviewer once the hardware instructions have been generated by labscript. Runviewer is assumed to be running locally, and will be launched if it is not already running once the first hdf5 file has been generated.
7. The labscript file: The Python file containing the experiment logic to be compiled into hardware instructions (see labscript).
8. The shot output folder: The location to store the hdf5 shot files. By default, the location in specified by the combination of a value in the laboratory PC configuration file (see labconfig), the name of the experiment logic Python file and the current date. The location automatically updates, at midnight, to a new folder for the day provided the folder location is left as the default.

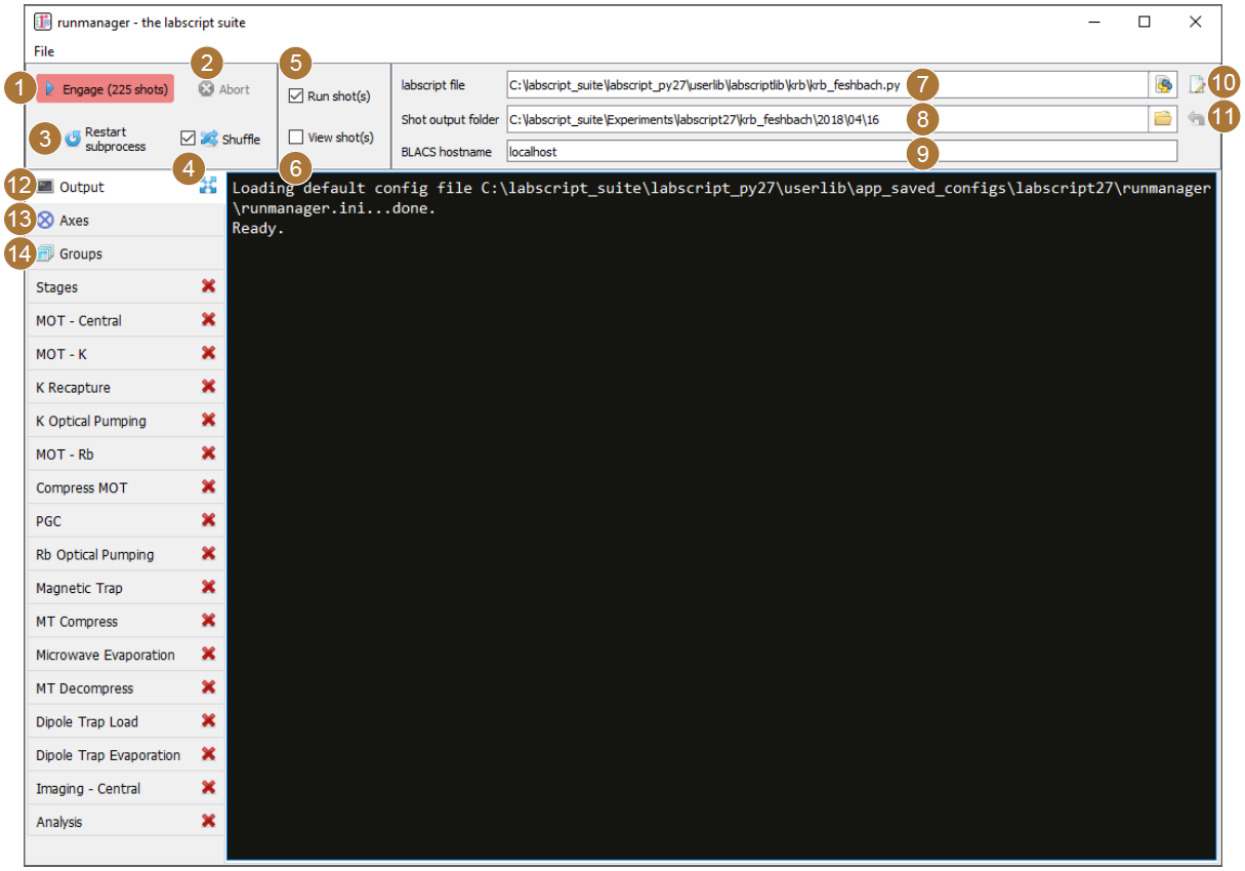

Fig. 2.1: The runmanager graphical interface with the main controls labelled as per the below text. The ‘output’ tab is currently shown.

9. The BLACS hostname: The network hostname of the PC the hdf5 shot files are to be sent to if the 'run shots' checkbox is ticked. It is expected that BLACS is running on the specified PC, and that network access (including firewalls and other network access controls) is configured appropriately.
10. The open in editor button: This button open the specified labscript experiment logic file in the text editor specified in the laboratory PC configuration file (see labconfig in the glossary).
11. The reset shot output folder button: This button resets the shot output folder to the default. This will re-enable the auto incrementation of the folder (based on the current date), which is disabled for custom locations.

These controls provide rapid access to the key functionality of runmanager (creating and distributing shot files) at all times, making for an efficient workflow. The rest of the runmanager interface exists within a set of tabs. The first 3 tabs contain further runmanager specific controls:

12. The output tab: This tab contains the terminal output of the shot creation process including the terminal output produced during the execution of the labscript experiment logic file. For example, Python `print` statements included in the experiment logic code will appear here during shot creation. This makes it easy to debug the experiment logic code using simple methods common to general purpose programming. Warnings and error messages generated by the labscript API also appear here in red text, so that any issues are immediately noticed and can be actioned. As this output is useful for debugging purposes, we allow the tab to be 'popped out' into a separate window so it can be visible at the same time as another tab (to avoid the need to frequently switch between the output and the tab containing the global variable(s) you are currently modifying).
13. The axes tab: This tab allows the user to control the iteration order of the parameters in the defined parameter space (see Fig. 2.2). The length of each axis of the parameter space is displayed, as is a shuffle checkbox for determining whether the points along that axis should be shuffled before the parameter space is expanded into the set of shots to be created. The global shuffle control (see item 4 in this list) is linked to the state of the shuffle checkboxes on the axes tab. This feature, along with the many benefits, is detailed further in *Parameter space scans* (see feature 3 and the paragraphs following).
14. The groups tab: This tab manages the hdf5 files that store the globals (see Fig. 2.3). Further details on managing global variables will be discussed in *Managing global variables*.

These tabs are then followed by an arbitrary number of tabs containing sets of global variables, which will be discussed further in *Managing global variables*.

In addition to this, runmanager can save and restore the entire GUI state via the relevant menu items in the 'File' menu. This allows rapid switching between different types of experiment logic and/or globals files.[2] This is particularly useful for shared experiment apparatuses, where different users want to run different experiments, and for the cases where a user wishes to rapidly switch between one of more diagnostic configurations they have previously saved.

2.2 Managing global variables

Runmanager provides a simple interface for grouping and modifying global variables. As mentioned previously, the 'groups' tab in runmanager handles creating and opening the hdf5 files that store the global variables. There are two levels of organisation for global variables:

- at the file level (globals can be stored across multiple files, the union of which is used to generate shots), and
- groups within each file.

Globals groups are created from the 'groups' tab in runmanager and can have arbitrary names (including spaces and special symbols). The only requirement is that a group name is unique within its file (it can however have the same name as a group in a different file). Globals within a group are then only used in the creation of shots if the 'active' checkbox for the group is checked on the groups tab (see Fig. 2.3). This provides a simple way of switching between

[2] For clarity, the values of the globals are not saved in this configuration file, but simply the location of the hdf5 file containing the globals. This means that any globals in files shared between saved runmanager configurations will share their values. For cases where global values should differ between runmanager configurations, separate globals files should be used.

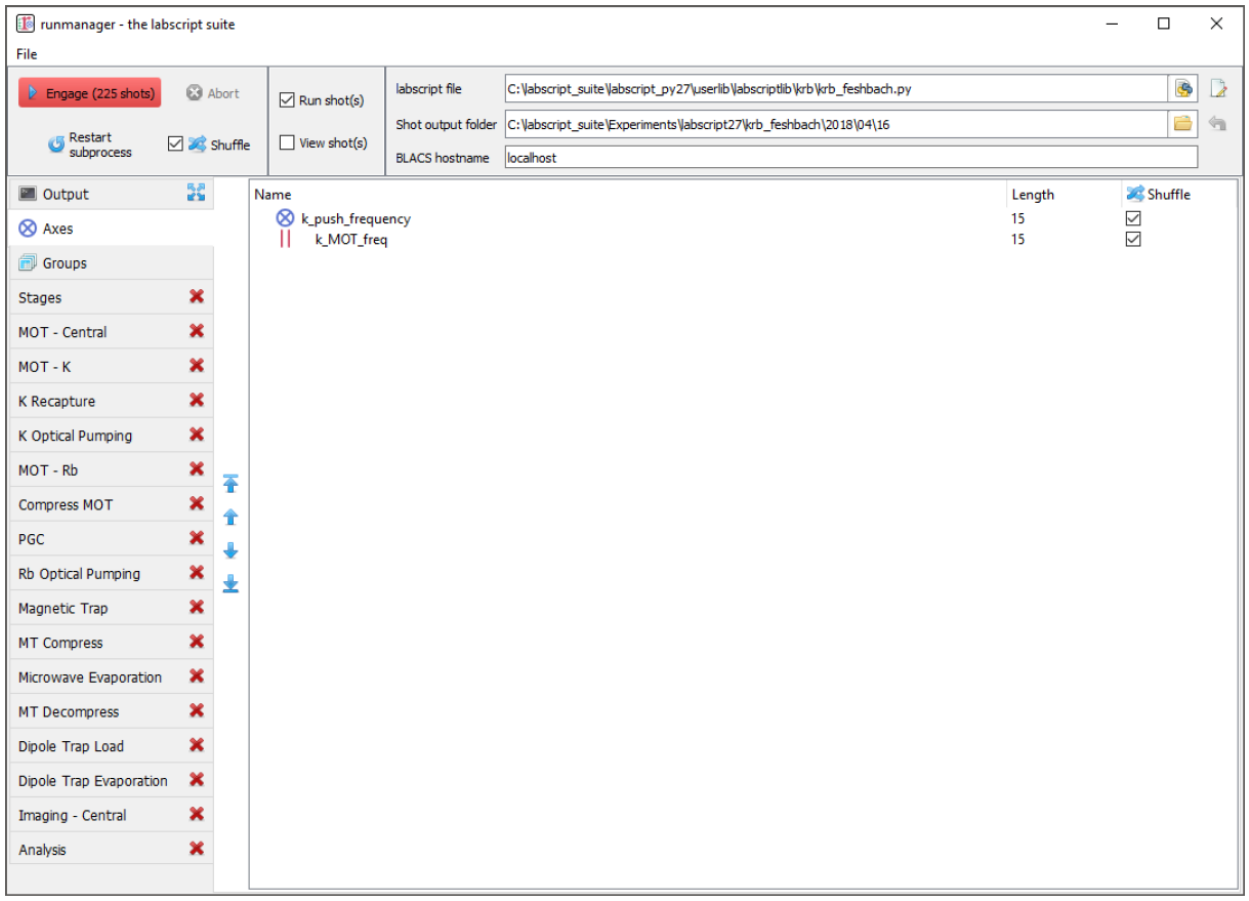

Fig. 2.2: The 'Axes' tab of runmanager. This tab displays a list of all global variables (indicated by the blue outer product icon) or groups of global variables (indicated by the icon with red parallel bars) that form axes of the parameter space that will be scanned over (see item 13 and *Parameter space scans* for further details). The order of the axes can be changed using the controls to the left of the list, which sets the order in which the outer product of the axes is performed (when generating the shot files).

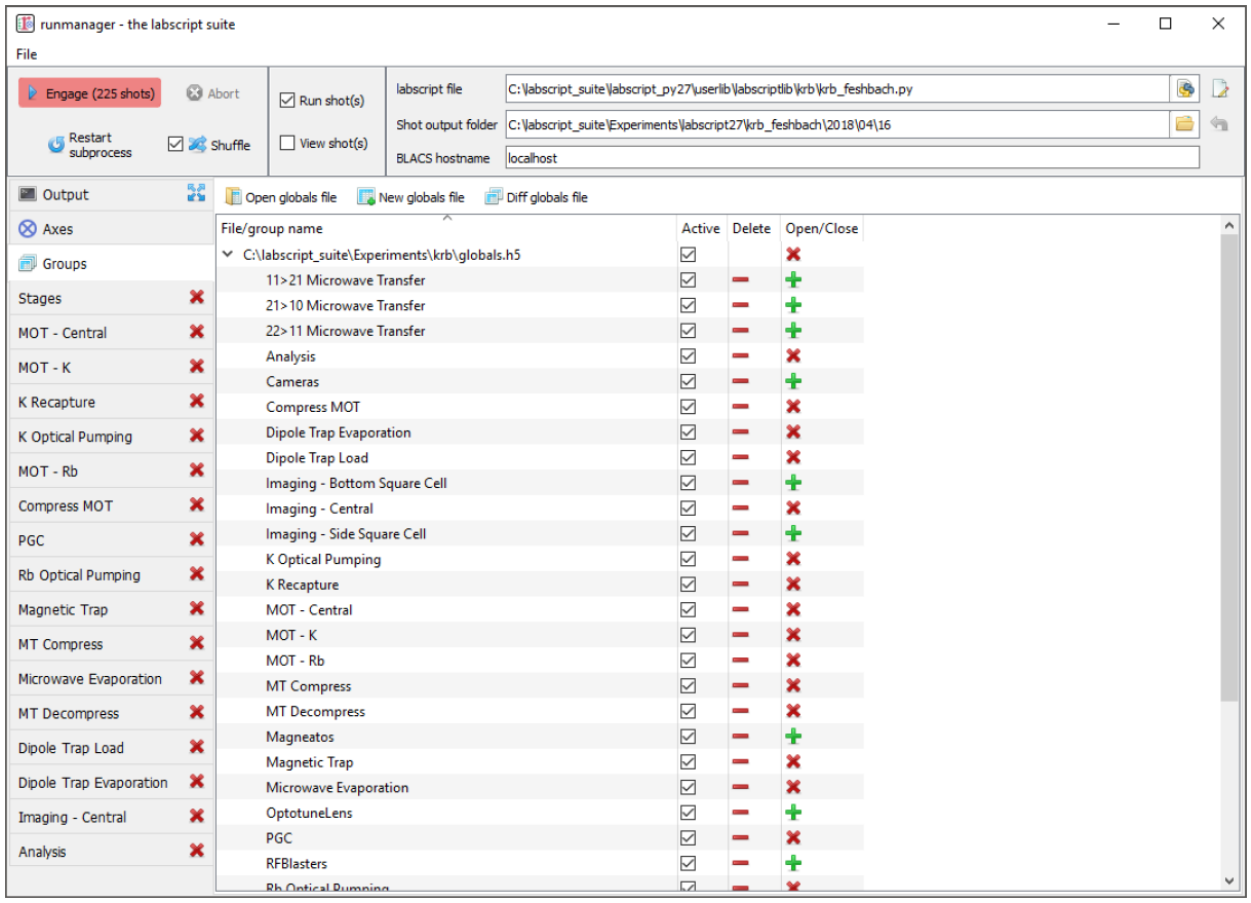

Fig. 2.3: The 'Groups' tab of runmanager. This tab displays the groups of global variables (stored in hdf5 files) that have been loaded into runmanager. From this tab, users can enabled/disable the use of these globals when compiling shots (using the 'active' checkboxes) and open/close an editing tab for each group. The editing tabs, when open, are displayed as additional tabs on the left most edge of the runmanager interface. See *Managing global variables* for further details on managing globals.

different groups of globals, allowing labs to maintain a common set of parameters for their experiments as well as individual parameter sets for specific users and/or experiments. For example, rather than modifying a set of globals in a group, a user could instead deactivate the group containing those globals, and instead ask runmanager to pull those globals from a separate file.

Each group of globals can be opened for editing in a new tab. We provide columns for the global name, value and units. The global name must be a valid Python variable name, and must not conflict with any member of the pylab library, python keywords, or existing items in the Python `__builtin__` module. This ensures that it can be injected into the labscript experiment logic (see labscript) without conflicting with existing Python functionality. The global name must also be unique across all active groups, as global groups are joined into a single set before passing the globals into the labscript experiment logic.

The value of a global can be any Python expression (including the use of functions from the numpy module), that evaluates to a datatype supported by hdf5, such as, but not limited to:

- a number: `1234` or `780e-9`,
- a string: `'N_atoms'`,
- a list or numpy array (which will be treated as an axis of a parameter space to scan, where the global variable will contain only one of the elements of the list or array in each shot): `[1, 2, 3]` or `array([1, 2, 3])`,
- a tuple (which despite being list like, will not be treated as an axis of a parameter space to scan and will instead be passed into labscript as the tuple specified): `(1, 2, 3)`,
- a Boolean: `True` or `False`
- an equation: `1+2`
- a Python inbuilt, or numpy, function call that returns a valid value: `linspace(0, 10, 10)`,
- an expression that references another defined global variable by name (the value of this global variable is used in its place): `2*other_global` or `linspace(0, 10, other_global)`,
- a Boolean expression: `(other_global1 and other_global2)` or `(other_global3 == 7)` or `(other_global4)`, or
- any of the above plus a Python comment: `780e-9 #This was previously 781e-9`.

As these expressions can become quite complex (see Fig. 2.4), the tooltip for the value cells displays the evaluated result of the Python expression. The value cell is also colour coded to the successful evaluation of the expression, so that mistakes can be easily identified (see Fig. 2.5).

Listing 2.1: Full expression for `drop_time` global

```
[ drop_time_rb if x else drop_time_k for x in central_image_order ] if
s11__imaging_both_species else ( drop_time_k if central_image_k == True
else ( drop_time_rb if central_image_rb == True else drop_time_general))
```

The units of the global are not currently passed into the labscript experiment logic code, but are a way to provide context to the user within runmanager. For example, if the labscript experiment logic multiplied a global variable for a frequency by `1e6` everywhere it was used (or the keyword argument `units="MHz"` was used everywhere), then you could type 'MHz' into the units column of runmanager so that a later user would know that the global was expected to be of that magnitude and would not accidentally enter it in kHz or Hz. In addition to this, globals whose values are explicitly specified as either `True` or `False` have their units automatically set to 'Bool', a checkbox is placed in the units column for easy toggling, and the units cell is colour coded to this checkbox for easy observation of the state. We frequently use this functionality to enable/disable various stages of our experiment logic file (see Fig. 2.6).

While we recommend storing globals in a dedicated set of files, the storage format for the globals is identical to that in any shot, which allows a user to easily load in globals from existing shots (even ones that have been executed and analysed). However, once pointed at an existing shot file, any modification to globals will modify that shot file, thus

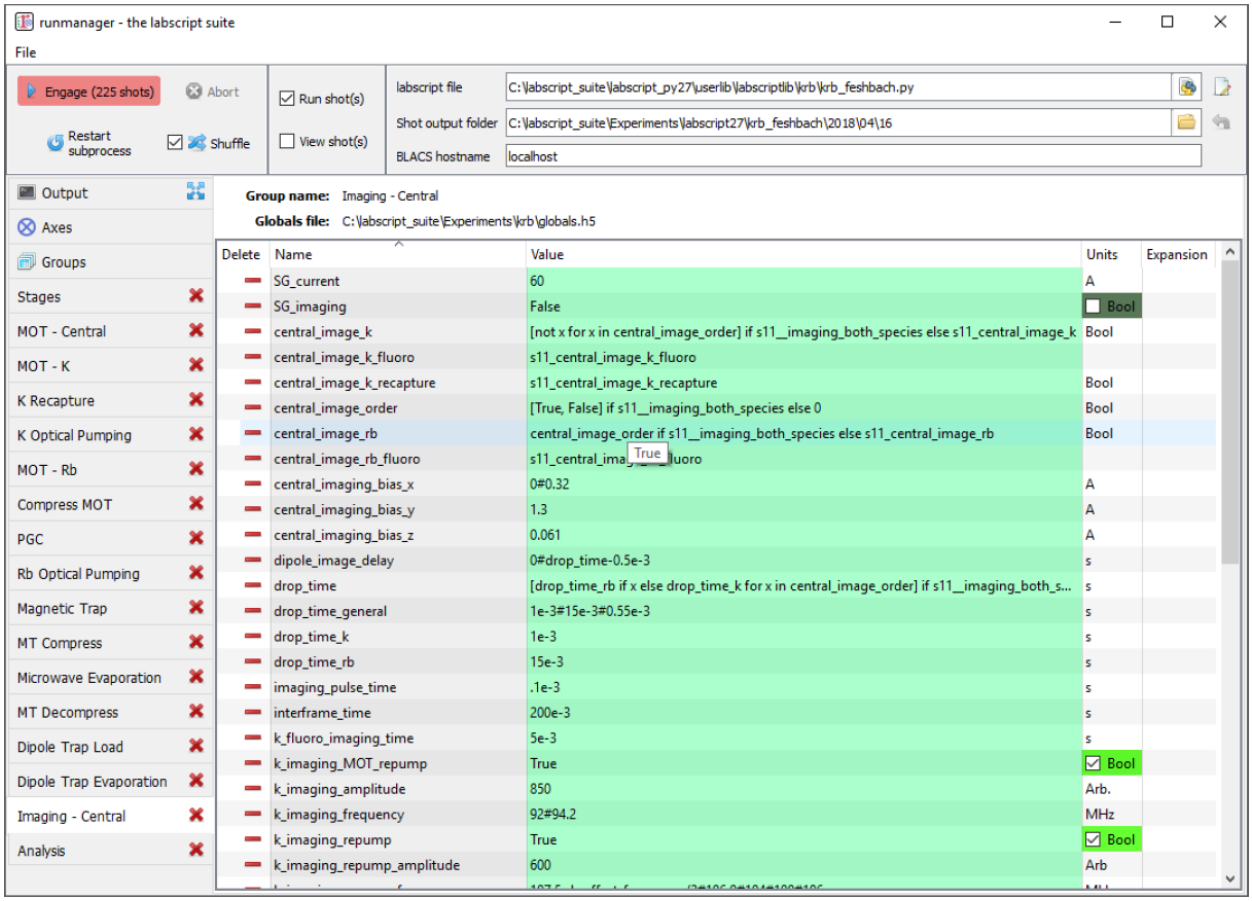

Fig. 2.4: An example of complex global variables that utilise Python expressions to define their value. Note, for example the `drop_time` global variable, whose full expression is shown below. The `drop_time` used is always drawn from one of three global variables, but the global variable selected is determined by a separate global variable (a Boolean) and may contain a list of drop times if the user wishes to image multiple species. In the case of the expression generating a list, this global becomes an axis of a parameter space, running two shots for every other data point in the parameter space (one shot to image each of the two species our experiment supports). Such an expression could not be defined within experiment logic as parameter spaces must be defined within runmanager, not labscript. In order to simplify the view of globals with complex expressions, the tooltip (shown for the `central_image_rb` global) shows the value(s) the global will take in the next compiled shot(s).

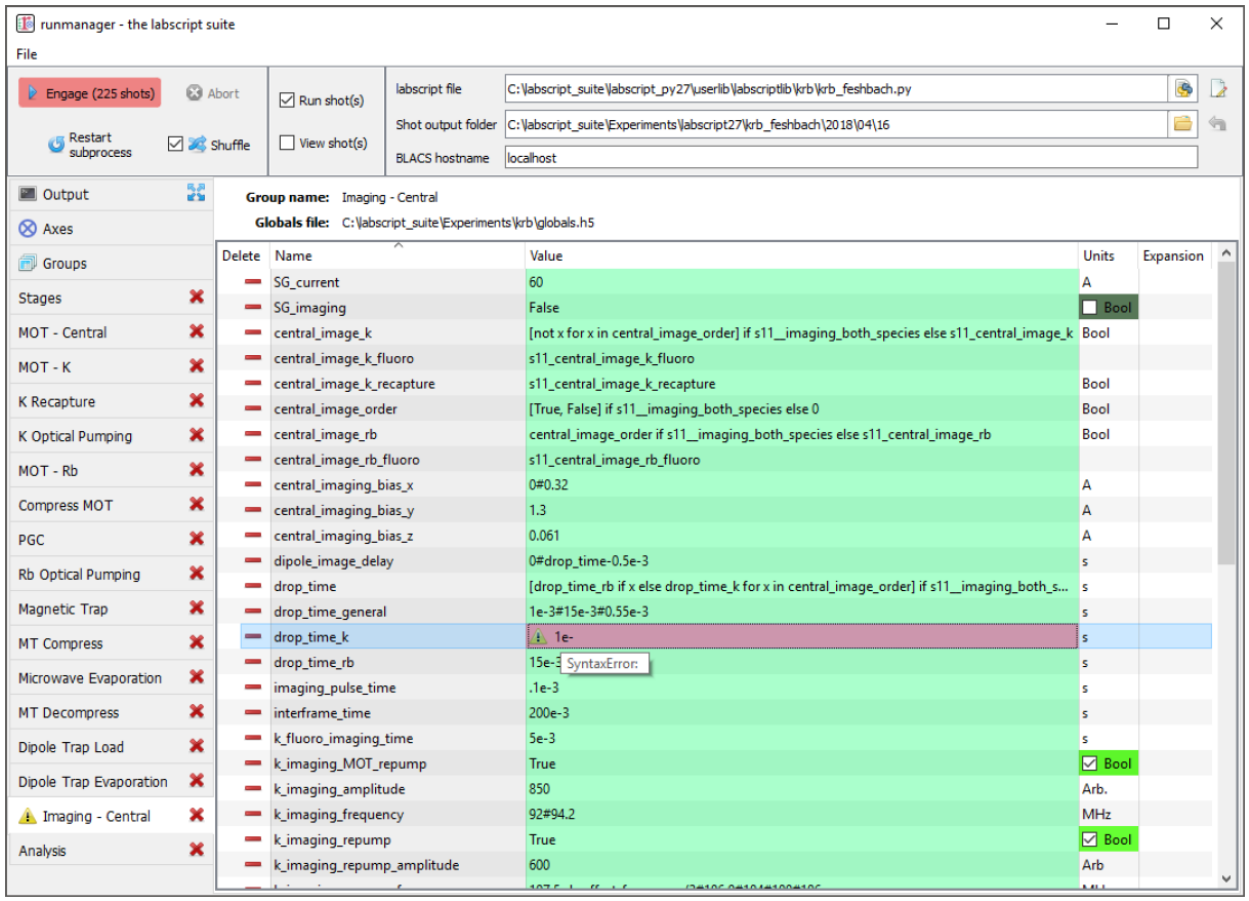

Fig. 2.5: An example of an evaluation error in a global variable. The user is notified of the error in two places: an icon appears next to the tab name and the global in question is highlighted in red. The tooltip displays the cause of the error, in this case a Python syntax error.

partially destroying the complete record of the experiment.[3] Thus, we encourage this feature to only be used for the cases where you wish to look at the globals from an old shot or where you wish to use the globals, without modification, to compile new shots.

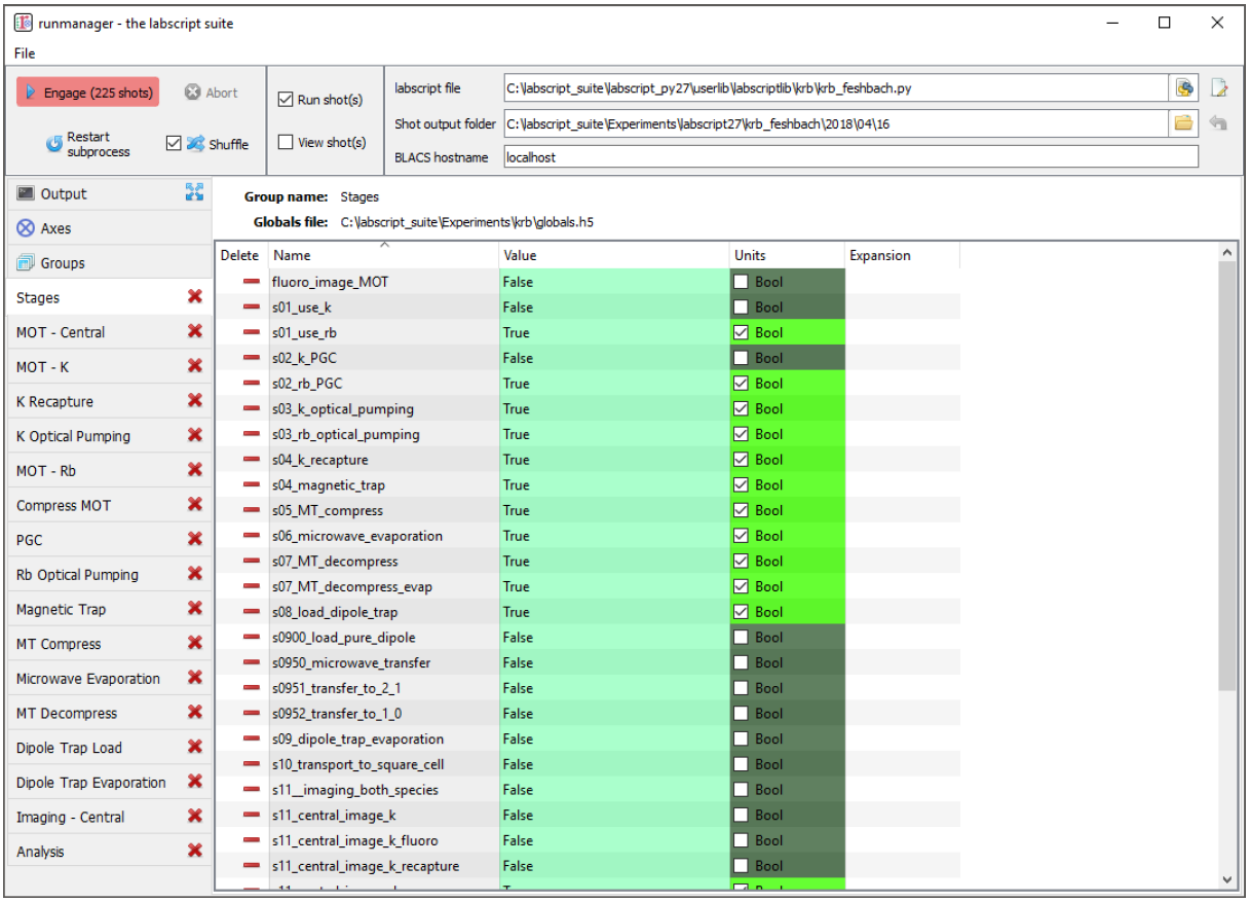

Fig. 2.6: An example of how a labscript experiment can be parameterised by a series of Boolean global variables. Here we split up the production of a BEC into several stages. We name each global with a prefix that increments in order to keep the globals in an appropriate sort order. Runmanager detects the Boolean type of the global, and provides a simple checkbox toggle in the units column, By using these global variables in our labscript experiment logic file, as the Boolean expression for an if statement, we can quickly turn on/off various stages of the BEC production process (which is very useful when debugging or optimising the BEC production process).

2.3 Parameter space scans

One of the key features of runmanager (and critical goals of our scientific control system) is the ability to easily automate the traversal of a large parameter space, an increasingly important requirement for performing modern ultracold atom experiments. Runmanager provides four features for managing parameter space scans:

1. The automatic detection of global variables that are defined as a list.[4] Such globals are labelled 'outer' in the expansions column as all such globals will be combined, via an outer product, into the parameter space to be

[3] Note that in an executed shot file, globals exist in two formats: the evaluated format (one point in the parameter space) used by labscript, and the raw strings as displayed in runmanager. Only the latter would be overwritten if globals were edited in the manor described in the main body text.

[4] Runmanager considers both Python lists and numpy arrays to be what we refer to as 'lists' in this section.

scanned. The number of shots to be generated, which is simply the product of the lengths of all 'outer' product globals, is displayed next to the engage button.

2. The ability to define what we term 'zip groups' after the Python function `zip`. Two (or more) globals (specified as lists) can be grouped together so that they iterate through values in lockstep. In this instance, the zip group is used as a single axis of the outer product rather than one axis for each global.
3. The third feature is the ability to define the order in which the axes of the parameter space are iterated over when producing individual shots (see the 'Axes' tab discussed previously in *The graphical interface*).
4. The ability to randomly shuffle the order of values within each global (or zip group) defined as a list. This can be done on a per global basis or on the entire set of shots that spans the defined parameter space.

These features provide a powerful basis for performing complex experiments.

Consider the following example. Many of the early stages of BEC production (for instance the MOT or magnetic trap stages) should be optimised for best phase-space density. Phase space density is calculated from several parameters; the most important being atom number and atom cloud temperature. While atom number can be easily measured from an absorption image from a single shot, temperature is most commonly determined from analysing the result of multiple shots. In this case, the drop time (the time between releasing the atoms from the trap and taking the absorption image) is varied for each shot and the temperature determined by fitting to the linearised relationship between atom cloud size and drop time. Already, it can be seen that measuring the phase space density for a single set of parameters requires several shots, which can be easily automated via the feature 1 described above.

Now consider the optimisation of MOT or magnetic trap parameters. Many of these are coupled and can not be independently optimised. As such, it is preferable to optimise two or three variables at once, measuring the phase-space density at each point to determine the optimal set of parameters. Such a parameter space typically takes several hours to complete due to the large number of shots that must be run. A BEC apparatus is likely to undergo systematic drifts during this time, which may invalidate the results. However, with careful thought, features 3 and 4 can be used to counteract this. For example, systematic drift will effect the linearity of the data when determining temperature, especially if the acquisition of each data point is separated by a significant period of time. However, by defining the drop time to be the inner most item of the outer product, you ensure that all shots needed to determine the phase-space density for a single set of MOT parameters are executed as close together in time as possible. Shuffling the order of the drop time then eliminates short term systematic drift, as does separately shuffling the order of the values in each remaining axes of the outer product (the MOT parameters). If long term systematic drifts need to be quantified, then an additional axes to the outer product can be added at the outer most layer in order to repeat each of the shots a prescribed number of times (by defining an additional 'dummy' global variable as `range(N)` where `N` is the number of times to repeat each shot).

While the above example may seem complicated, runmanager makes it trivial to implement. A user simply defines the list of values to scan over for each parameter, sets the order in which the outer product should use each axis, and specifies whether the values for each axis should be shuffled. Once done, clicking the engage button generates the sequence of shots and sends them to BLACS to be executed on the experiment.

2.4 Evaluation of globals

All global variable expressions are automatically evaluated after a change to any global variable. This serves to both update the tooltip with the result of the expression, detect axes of a parameter space to scan (and group them into zip groups if appropriate) and warn the user of any errors during the evaluation of the globals. As discussed previously, runmanager allows these global variable expressions to reference other global variables. This allows a user to maintain a record of a set of parameters, and all relevant quantities derived from one or more of those parameters, without ever storing a parameter more than once. This ensures that important quantities need not be derived (from globals) in the labscript experiment logic script, and that they are accessible directly during the analysis stage (see lyse).

To implement this, we take advantage of the Python built-in function exec which not only evaluates a string containing a Python expression, but can do so from within a controlled namespace. This has a two-fold benefit. The first is that

it allows us to provide access to a specific set of functions that can be used from within the Python expressions (such as numpy functions like `linspace`). The second is that it allows us to keep track of the relationship between global variables, which is critical for both descriptive error messages and automatically detecting which globals should be combined into a zip groups.

The Python `exec` function is given access to a namespace to work in via an optional argument in the form of a dictionary. Keys and values in this dictionary correspond to variable names in the namespace and their associated values respectively. Rather than using a native Python dictionary for the namespace, we subclass the Python dictionary and override the built-in dictionary method for looking up entries in the dictionary. When combined with exec, this translates to our `dict` subclasses tracking each time the `exec` function requests the value of a variable in the namespace. This then provides us with a mapping of each global variable, and the names of global variables that it depends on. In order to resolve both the order in which global variable expressions are evaluated in, and detection of any recursive relationships, we begin by evaluating all global expressions and then recursively re-evaluate the set of globals that did not evaluate in the previous iteration. The first iteration will evaluate any (correctly defined) independent globals, and subsequent iterations will then be able to evaluate globals that depend on other globals (once those other globals have been evaluated by a previous iteration).

The hierarchy of global interdependencies is then used to determine automatic zip group names, which are based on the name of the global in the hierarchy that does not depend on any other. If a global depends on multiple other globals, then the zip group name is chosen semi-randomly based on the order of the items in the Python dictionary (which depends on a hash of the dictionary key names and the size of the dictionary). However, it is of course always possible to overwrite the automatic zip group name with something else should our algorithm choose incorrectly.

We believe that this complex evaluation of global variables is only possible due to the use of an interpreted language that has tools for parsing its own syntax. As such, the choice of Python as our programming language has allowed us to implement extremely useful, advanced features that might otherwise be too difficult to produce in more low level languages such as C++.

2.5 Shot creation

The internal process for generating shot files is quite complex. This is primarily motivated by the desire for modularity (for example, to separate shot file generation from hardware instruction generation) and the desire for robustness. As runmanager ultimately initiates the execution of user code (the labscript experiment logic file), there is a risk that problems in the user code could crash runmanager. We mitigate this by using a multi-process architecture.

We originally spawned a new Python process for each shot (in order to guarantee the internal state of labscript was fresh). However the time required to start a Python process (especially on Windows) was a considerable fraction of the entire shot generation time. As such we now use a single, long-lived, Python process and clean-up the internal state of labscript and Python explicitly after each shot.

To generate shot files, runmanager:

1. Re-evaluates all globals (see *Evaluation of globals*). This both determines the number of shots to produce, and generates the evaluated set of global variables for each shot.
2. The globals are then written to hdf5 files, one file for each shot. We also write the unevaluated globals into every hdf5 file, in order to provide a complete record of the experiment (the unevaluated globals contain information about the parameter space that is not available when looking at the single point of parameter space in the evaluated globals of a single shot file).
3. In a thread (in order to keep the GUI responsive), we iterate over the set of files and send their file paths to a long-running subprocess (launched by runmanager at startup) that is used to execute labscript code in an isolated environment. We call this process the 'compilation subprocess'.
4. The subprocess, which has the labscript API imported, calls an initialisation method to inform the labscript API of the hdf5 file to write hardware instructions to.

5. The subprocess loads the global variables from runmanager into the `__builtin__` dictionary.
6. The subprocess then executes the labscript experiment logic file (using the Python function `exec`) in an isolated namespace, which invokes the labscript API via the users experiment logic and generates the required hardware instructions and saves them in the hdf5 file. Terminal output (for example, `print` statements) are sent back to the parent runmanager process and placed in the output tab.
7. The subprocess restores the `__builtin__` dictionary to its original state to prevent globals from polluting subsequent shots. A clean-up method from the labscript API is also called so that the internal state of the labscript Python module is also reset.

Once shot files are created, the file paths are sent to runviewer or BLACS, as determined by the checkboxes in the runmanager GUI, for viewing and/or executing the shots respectively. This architecture also has several unrealised benefits:

1. If the need arose, we could easily parallelise the generation of hardware instructions by instantiating multiple instances of the compilation subprocess.
2. We could use runmanager as a generic parameter (space) management software by replacing the compilation subprocess with something else. For example, runmanager could be used to manage parameters for simulations, producing one shot file per simulation to be run in the same way we do for real experiments. These files could then be sent to a scheduling program (like BLACS) that feeds them to the simulation software.

CHAPTER

THREE

API REFERENCE

runmanager	
runmanager.functions	
runmanager.remote	
runmanager.batch_compiler	
runmanager.globals_diff	Script that runs *runmanager.globals_diff_shots()* between two shot files.
runmanager.__main__	Runmanager GUI and supporting code

3.1 runmanager

Functions

add_expansion_groups(filename)	backward compatability, for globals files which don't have expansion groups.
compile_labscript(labscript_file, run_file)	Compiles labscript_file with the run file, returning the processes return code, stdout and stderr.
compile_labscript_async(labscript_file, ...)	Compiles labscript_file with run_file.
compile_labscript_with_globals_files(...)	Creates a run file output_path, using all the globals from globals_files.
compile_labscript_with_globals_files_async(..	Same as compile_labscript_with_globals_files, except it launches a thread to do
compile_multishot_async(labscript_file, ...)	Compiles labscript_file with run_files.
copy_group(source_globals_file, ...[, ...])	This function copies the group source_groupname from source_globals_file to dest_globals_file and renames the new group so that there is no name collision.
delete_global(filename, groupname, globalname)	
delete_group(filename, groupname)	
dict_diff(dict1, dict2)	Return the difference between two dictionaries as a dictionary of key: [val1, val2] pairs.
evaluate_globals(sequence_globals[, ...])	Takes a dictionary of globals as returned by get_globals.

continues on next page

Table 3.1 – continued from previous page

expand_globals(sequence_globals, evaled_globals)	Expands iterable globals according to their expansion settings.
find_comments(src)	Return a list of start and end indices for where comments are in given Python source.
flatten_globals(sequence_globals[, evaluated])	Flattens the data structure of the globals.
get_all_groups(h5_files)	returns a dictionary of group_name: h5_path pairs from a list of h5_files.
get_expansion(filename, groupname, globalname)	
get_globals(groups)	Takes a dictionary of group_name: h5_file pairs and pulls the globals out of the groups in their files.
get_globalslist(filename, groupname)	
get_grouplist(filename)	
get_shot_globals(filepath)	Returns the evaluated globals for a shot, for use by labscript or lyse.
get_units(filename, groupname, globalname)	
get_value(filename, groupname, globalname)	
globals_diff_groups(active_groups, other_groups)	Given two sets of globals groups, perform a diff of the raw and evaluated globals.
globals_diff_shots(file1, file2[, max_cols])	
guess_expansion_type(value)	
is_valid_hdf5_group_name(name)	Ensure that a string is a valid name for an hdf5 group.
is_valid_python_identifier(name)	
iterator_to_tuple(iterator[, max_length])	
make_run_file_from_globals_files(...[, config])	Creates a run file output_path, using all the globals from globals_files.
make_run_files(output_folder, ...[, shuffle])	Does what it says.
make_single_run_file(filename, ...)	Does what it says.
new_global(filename, groupname, globalname)	
new_globals_file(filename)	
new_group(filename, groupname)	
new_sequence_details(script_path[, config, ...])	Generate the details for a new sequence: the toplevel attrs sequence_date, sequence_index, sequence_id; and the the output directory and filename prefix for the shot files, according to labconfig settings.
next_sequence_index(shot_basedir, dt[, ...])	Return the next sequence index for sequences in the given base directory (i.e. <experiment_shot_storage>/<script_basename>) and the date of the given datetime object, and increment the sequence index atomically on disk if increment=True.

continues on next page

Table 3.1 – continued from previous page

remove_comments_and_tokenify(src)	Removes comments from source code, leaving it otherwise intact, and returns it.
rename_global(filename, groupname, ...)	
rename_group(filename, oldgroupname, ...)	
set_expansion(filename, groupname, ...)	
set_units(filename, groupname, globalname, units)	
set_value(filename, groupname, globalname, value)	

3.1.1 runmanager.add_expansion_groups

runmanager.**add_expansion_groups**(*filename*)

backward compatability, for globals files which don't have expansion groups. Create them if they don't exist. Guess expansion settings based on datatypes, if possible.

3.1.2 runmanager.compile_labscript

runmanager.**compile_labscript**(*labscript_file*, *run_file*)

Compiles labscript_file with the run file, returning the processes return code, stdout and stderr.

3.1.3 runmanager.compile_labscript_async

runmanager.**compile_labscript_async**(*labscript_file*, *run_file*, *stream_port*, *done_callback*)

Compiles labscript_file with run_file. This function is designed to be called in a thread. The stdout and stderr from the compilation will be shovelled into stream_port via zmq push as it spews forth, and when compilation is complete, done_callback will be called with a boolean argument indicating success. Note that the zmq communication will be encrypted, or not, according to security settings in labconfig. If you want to receive the data on a zmq socket, do so using a PULL socket created from a labscript_utils.ls_zprocess.Context, or using a labscript_utils.ls_zprocess.ZMQServer. These subclasses will also be configured with the appropriate security settings and will be able to receive the messages.

3.1.4 runmanager.compile_labscript_with_globals_files

runmanager.**compile_labscript_with_globals_files**(*labscript_file*, *globals_files*, *output_path*)

Creates a run file output_path, using all the globals from globals_files. Compiles labscript_file with the run file, returning the processes return code, stdout and stderr.

3.1.5 runmanager.compile_labscript_with_globals_files_async

runmanager.**compile_labscript_with_globals_files_async**(*labscript_file*, *globals_files*, *output_path*, *stream_port*, *done_callback*)

Same as compile_labscript_with_globals_files, except it launches a thread to do
the work and does not return anything. Instead, stderr and stdout will be put to stream_port via zmq push in the multipart message format ['stdout','hello, world

']

etc. When compilation is finished, the function done_callback will be called a boolean argument indicating success or failure. If you want to receive the data on a zmq socket, do so using a PULL socket created from a labscript_utils.ls_zprocess.Context, or using a labscript_utils.ls_zprocess.ZMQServer. These subclasses will also be configured with the appropriate security settings and will be able to receive the messages.

3.1.6 runmanager.compile_multishot_async

runmanager.**compile_multishot_async**(*labscript_file*, *run_files*, *stream_port*, *done_callback*)

Compiles labscript_file with run_files. This function is designed to be called in a thread. The stdout and stderr from the compilation will be shovelled into stream_port via zmq push as it spews forth, and when each compilation is complete, done_callback will be called with a boolean argument indicating success. Compilation will stop after the first failure. If you want to receive the data on a zmq socket, do so using a PULL socket created from a labscript_utils.ls_zprocess.Context, or using a labscript_utils.ls_zprocess.ZMQServer. These subclasses will also be configured with the appropriate security settings and will be able to receive the messages.

3.1.7 runmanager.copy_group

runmanager.**copy_group**(*source_globals_file*, *source_groupname*, *dest_globals_file*, *delete_source_group=False*)

This function copies the group source_groupname from source_globals_file to dest_globals_file and renames the new group so that there is no name collision. If delete_source_group is False the copyied files have a suffix '_copy'.

3.1.8 runmanager.delete_global

runmanager.**delete_global**(*filename*, *groupname*, *globalname*)

3.1.9 runmanager.delete_group

runmanager.**delete_group**(*filename*, *groupname*)

3.1.10 runmanager.dict_diff

runmanager.**dict_diff**(*dict1*, *dict2*)

Return the difference between two dictionaries as a dictionary of key: [val1, val2] pairs. Keys unique to either dictionary are included as key: [val1, '-'] or key: ['-', val2].

3.1.11 runmanager.evaluate_globals

runmanager.**evaluate_globals**(*sequence_globals*, *raise_exceptions=True*)

Takes a dictionary of globals as returned by get_globals. These globals are unevaluated strings. Evaluates them all in the same namespace so that the expressions can refer to each other. Iterates to allow for NameErrors to be resolved by subsequently defined globals. Throws an exception if this does not result in all errors going away. The exception contains the messages of all exceptions which failed to be resolved. If raise_exceptions is False, any evaluations resulting in an exception will instead return the exception object in the results dictionary

3.1.12 runmanager.expand_globals

runmanager.**expand_globals**(*sequence_globals*, *evaled_globals*, *expansion_config=None*, *return_dimensions=False*)

Expands iterable globals according to their expansion settings. Creates a number of 'axes' which are to be outer product'ed together. Some of these axes have only one element, these are globals that do not vary. Some have a set of globals being zipped together, iterating in lock-step. Others contain a single global varying across its values (the globals set to 'outer' expansion). Returns a list of shots, each element of which is a dictionary for that shot's globals.

3.1.13 runmanager.find_comments

runmanager.**find_comments**(*src*)

Return a list of start and end indices for where comments are in given Python source. Comments on separate lines with only whitespace in between them are coalesced. Whitespace preceding a comment is counted as part of the comment.

3.1.14 runmanager.flatten_globals

runmanager.**flatten_globals**(*sequence_globals*, *evaluated=False*)

Flattens the data structure of the globals. If evaluated=False, saves only the value expression string of the global, not the units or expansion.

3.1.15 runmanager.get_all_groups

runmanager.**get_all_groups**(*h5_files*)

returns a dictionary of group_name: h5_path pairs from a list of h5_files.

3.1.16 runmanager.get_expansion

runmanager.**get_expansion**(*filename*, *groupname*, *globalname*)

3.1.17 runmanager.get_globals

runmanager.**get_globals**(*groups*)

Takes a dictionary of group_name: h5_file pairs and pulls the globals out of the groups in their files. The globals are strings storing python expressions at this point. All these globals are packed into a new dictionary, keyed by group_name, where the values are dictionaries which look like {global_name: (expression, units, expansion), ...}

3.1.18 runmanager.get_globalslist

runmanager.**get_globalslist**(*filename*, *groupname*)

3.1.19 runmanager.get_grouplist

runmanager.**get_grouplist**(*filename*)

3.1.20 runmanager.get_shot_globals

runmanager.**get_shot_globals**(*filepath*)

Returns the evaluated globals for a shot, for use by labscript or lyse. Simple dictionary access as in dict(h5py.File(filepath).attrs) would be fine except we want to apply some hacks, so it's best to do that in one place.

Deprecated: use identical function `labscript_utils.shot_utils.get_shot_globals`

3.1.21 runmanager.get_units

runmanager.**get_units**(*filename*, *groupname*, *globalname*)

3.1.22 runmanager.get_value

runmanager.**get_value**(*filename*, *groupname*, *globalname*)

3.1.23 runmanager.globals_diff_groups

runmanager.**globals_diff_groups**(*active_groups*, *other_groups*, *max_cols=1000*, *return_string=True*)

Given two sets of globals groups, perform a diff of the raw and evaluated globals.

3.1.24 runmanager.globals_diff_shots

runmanager.**globals_diff_shots**(*file1*, *file2*, *max_cols=100*)

3.1.25 runmanager.guess_expansion_type

runmanager.**guess_expansion_type**(*value*)

3.1.26 runmanager.is_valid_hdf5_group_name

runmanager.**is_valid_hdf5_group_name**(*name*)

Ensure that a string is a valid name for an hdf5 group.

The names of hdf5 groups may only contain ASCII characters. Furthermore, the characters "/" and "." are not allowed.

Parameters
name (*str*) – The potential name for an hdf5 group.

Returns

Whether or not name is a valid name for an hdf5 group. This will be `True` if it is a valid name or `False` otherwise.

Return type
bool

3.1.27 runmanager.is_valid_python_identifier

runmanager.**is_valid_python_identifier**(*name*)

3.1.28 runmanager.iterator_to_tuple

runmanager.**iterator_to_tuple**(*iterator*, *max_length=1000000*)

3.1.29 runmanager.make_run_file_from_globals_files

runmanager.**make_run_file_from_globals_files**(*labscript_file*, *globals_files*, *output_path*, *config=None*)

Creates a run file output_path, using all the globals from globals_files. Uses labscript_file to determine the sequence_attrs only

3.1.30 runmanager.make_run_files

runmanager.**make_run_files**(*output_folder*, *sequence_globals*, *shots*, *sequence_attrs*, *filename_prefix*, *shuffle=False*)

Does what it says. sequence_globals and shots are of the datatypes returned by get_globals and get_shots, one is a nested dictionary with string values, and the other a flat dictionary. sequence_attrs is a dict of the attributes pertaining to this sequence to be initially set at the top-level group of the h5 file, as returned by new_sequence_details. output_folder and filename_prefix determine the directory shot files will be output to, as well as their filenames (this function will generate filenames with the shot number and .h5 extension appended to filename_prefix). Sensible defaults for these are also returned by new_sequence_details(), so preferably these should be used.

Shuffle will randomise the order that the run files are generated in with respect to which element of shots they come from. This function returns a *generator*. The run files are not actually created until you loop over this generator (which gives you the filepaths). This is useful for not having to clean up as many unused files in the event of failed compilation of labscripts. If you want all the run files to be created at some point, simply convert the returned generator to a list. The filenames the run files are given is simply the sequence_id with increasing integers appended.

3.1.31 runmanager.make_single_run_file

runmanager.**make_single_run_file**(*filename*, *sequenceglobals*, *runglobals*, *sequence_attrs*, *run_no*, *n_runs*)

Does what it says. runglobals is a dict of this run's globals, the format being the same as that of one element of the list returned by expand_globals. sequence_globals is a nested dictionary of the type returned by get_globals. sequence_attrs is a dict of attributes pertaining to this sequence, as returned by new_sequence_details. run_no and n_runs must be provided, if this run file is part of a sequence, then they should reflect how many run files are being generated in this sequence, all of which must have identical sequence_attrs.

3.1.32 runmanager.new_global

runmanager.**new_global**(*filename*, *groupname*, *globalname*)

3.1.33 runmanager.new_globals_file

runmanager.**new_globals_file**(*filename*)

3.1.34 runmanager.new_group

runmanager.**new_group**(*filename*, *groupname*)

3.1.35 runmanager.new_sequence_details

runmanager.**new_sequence_details**(*script_path*, *config=None*, *increment_sequence_index=True*)

Generate the details for a new sequence: the toplevel attrs sequence_date, sequence_index, sequence_id; and the the output directory and filename prefix for the shot files, according to labconfig settings. If increment_sequence_index=True, then we are claiming the resulting sequence index for use such that it cannot be used by anyone else. This should be done if the sequence details are immediately about to be used to compile a sequence. Otherwise, set increment_sequence_index to False, but in that case the results are indicative only and one should call this function again with increment_sequence_index=True before compiling the sequence, as otherwise the sequence_index may be used by other code in the meantime.

3.1.36 runmanager.next_sequence_index

runmanager.**next_sequence_index**(*shot_basedir*, *dt*, *increment=True*)

Return the next sequence index for sequences in the given base directory (i.e. <experiment_shot_storage>/<script_basename>) and the date of the given datetime object, and increment the sequence index atomically on disk if increment=True. If not setting increment=True, then the result is indicative only and may be used by other code at any time. One must increment the sequence index prior to use.

3.1.37 runmanager.remove_comments_and_tokenify

runmanager.**remove_comments_and_tokenify**(*src*)

Removes comments from source code, leaving it otherwise intact, and returns it. Also returns the raw tokens for the code, allowing comparisons between source to be made without being sensitive to whitespace.

3.1.38 runmanager.rename_global

runmanager.**rename_global**(*filename*, *groupname*, *oldglobalname*, *newglobalname*)

3.1.39 runmanager.rename_group

runmanager.**rename_group**(*filename*, *oldgroupname*, *newgroupname*)

3.1.40 runmanager.set_expansion

runmanager.**set_expansion**(*filename*, *groupname*, *globalname*, *expansion*)

3.1.41 runmanager.set_units

runmanager.**set_units**(*filename*, *groupname*, *globalname*, *units*)

3.1.42 runmanager.set_value

runmanager.**set_value**(*filename*, *groupname*, *globalname*, *value*)

Classes

TraceDictionary(*args, **kwargs)

3.1.43 runmanager.TraceDictionary

class runmanager.**TraceDictionary**(**args*, ***kwargs*)

Bases: dict

__init__(**args*, ***kwargs*)

Methods

__init__(*args, **kwargs)	
clear()	
copy()	
fromkeys([value])	Create a new dictionary with keys from iterable and values set to value.
get(key[, default])	Return the value for key if key is in the dictionary, else default.
items()	
keys()	
pop(k[,d])	If the key is not found, return the default if given; otherwise, raise a KeyError.
popitem()	Remove and return a (key, value) pair as a 2-tuple.
setdefault(key[, default])	Insert key with a value of default if key is not in the dictionary.
start_trace()	
stop_trace()	
update([E,]**F)	If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]
values()	

start_trace()

stop_trace()

Exceptions

ExpansionError	An exception class so that error handling code can tell when a parsing exception was caused by a mismatch with the expansion mode

3.1.44 runmanager.ExpansionError

exception runmanager.**ExpansionError**

An exception class so that error handling code can tell when a parsing exception was caused by a mismatch with the expansion mode

3.2 runmanager.functions

Functions

drop_times(t_min, t_max, n_points[, ...])	
first()	Infinite iterator.
quadspace(t_min, t_max, n_points[, ...])	

3.2.1 runmanager.functions.drop_times

runmanager.functions.**drop_times**(*t_min*, *t_max*, *n_points*, *randomise=False*, *repeats=1*)

3.2.2 runmanager.functions.first

runmanager.functions.**first**()

Infinite iterator. Its first return value is true, subsequent return values are False

3.2.3 runmanager.functions.quadspace

runmanager.functions.**quadspace**(*t_min*, *t_max*, *n_points*, *randomise=False*, *repeats=1*)

3.3 runmanager.remote

Functions

abort()	Trigger abort compilation/submission
engage()	Trigger shot compilation/submission
error_in_globals()	True if any tab of an active group contains error(s)
get_globals([raw])	Return all active globals as a dict of the form: {'<global_name>': value}.
get_labscript_file()	Get the path of the current experiment script
get_run_shots()	Get boolean state of 'Run shot(s)' checkbox
get_shot_output_folder()	Get the current shot output folder
get_shuffle()	Get boolean state of 'Shuffle' checkbox
get_version()	Return the version of runmanager the server is running in
get_view_shots()	Get boolean state of 'View shot(s)' checkbox
is_output_folder_default()	True if shot output folder is not the default path
n_shots()	Get the number of prospective shots from pressing 'Engage'
reset_shot_output_folder()	Reset the shot output folder to the default path
say_hello()	Ping the runmanager server for a response
set_globals(globals[, raw])	For a dict of the form {'<global_name>': value}, set the given globals to the given values.
set_labscript_file(value)	Set the current experiment script
set_run_shots(value)	Set boolean state of 'Run shot(s)' checkbox
set_shot_output_folder(value)	Set the shot output folder
set_shuffle(value)	Set boolean state of 'Shuffle' checkbox
set_view_shots(value)	Set boolean state of 'View shot(s)' checkbox

3.3.1 runmanager.remote.abort

runmanager.remote.**abort**()

Trigger abort compilation/submission

3.3.2 runmanager.remote.engage

runmanager.remote.**engage**()

Trigger shot compilation/submission

3.3.3 runmanager.remote.error_in_globals

runmanager.remote.**error_in_globals**()

True if any tab of an active group contains error(s)

3.3.4 runmanager.remote.get_globals

runmanager.remote.**get_globals**(*raw=False*)

Return all active globals as a dict of the form: {'<global_name>': value}. If raw=True, then the global values are returned as their string representations, as stored in the runmanager GUI and globals HDF5 file, otherwise they are evaluated as python objects and then returned.

3.3.5 runmanager.remote.get_labscript_file

runmanager.remote.**get_labscript_file**()

Get the path of the current experiment script

3.3.6 runmanager.remote.get_run_shots

runmanager.remote.**get_run_shots**()

Get boolean state of 'Run shot(s)' checkbox

3.3.7 runmanager.remote.get_shot_output_folder

runmanager.remote.**get_shot_output_folder**()

Get the current shot output folder

3.3.8 runmanager.remote.get_shuffle

runmanager.remote.**get_shuffle**()

Get boolean state of 'Shuffle' checkbox

3.3.9 runmanager.remote.get_version

runmanager.remote.**get_version**()

Return the version of runmanager the server is running in

3.3.10 runmanager.remote.get_view_shots

runmanager.remote.**get_view_shots**()

Get boolean state of 'View shot(s)' checkbox

3.3.11 runmanager.remote.is_output_folder_default

runmanager.remote.**is_output_folder_default**()

True if shot output folder is not the default path

3.3.12 runmanager.remote.n_shots

runmanager.remote.**n_shots**()

Get the number of prospective shots from pressing 'Engage'

3.3.13 runmanager.remote.reset_shot_output_folder

runmanager.remote.**reset_shot_output_folder**()

Reset the shot output folder to the default path

3.3.14 runmanager.remote.say_hello

runmanager.remote.**say_hello**()

Ping the runmanager server for a response

3.3.15 runmanager.remote.set_globals

runmanager.remote.**set_globals**(*globals*, *raw=False*)

For a dict of the form {'<global_name>': value}, set the given globals to the given values. If raw=True, then global values will be treated as the string representations of Python objects rather than the objects themselves, and written directly to the HDF5 file and runmanager GUI without calling repr() on them first.

3.3.16 runmanager.remote.set_labscript_file

runmanager.remote.**set_labscript_file**(*value*)

Set the current experiment script

3.3.17 runmanager.remote.set_run_shots

runmanager.remote.**set_run_shots**(*value*)

Set boolean state of 'Run shot(s)' checkbox

3.3.18 runmanager.remote.set_shot_output_folder

runmanager.remote.**set_shot_output_folder**(*value*)

Set the shot output folder

3.3.19 runmanager.remote.set_shuffle

runmanager.remote.**set_shuffle**(*value*)

Set boolean state of 'Shuffle' checkbox

3.3.20 runmanager.remote.set_view_shots

runmanager.remote.**set_view_shots**(*value*)

Set boolean state of 'View shot(s)' checkbox

Classes

Client([host, port, timeout])	A ZMQClient for communication with runmanager

3.3.21 runmanager.remote.Client

class runmanager.remote.**Client**(*host=None*, *port=None*, *timeout=None*)

Bases: ZMQClient

A ZMQClient for communication with runmanager

__init__(*host=None*, *port=None*, *timeout=None*)

Methods

__init__([host, port, timeout])	
abort()	Trigger abort compilation/submission
clear_interrupt()	Clear our internal Interruptor object so that future get*()/push*() calls can proceed as normal.
engage()	Trigger shot compilation/submission
error_in_globals()	True if any tab of an active group contains error(s)
get_globals([raw])	Return all active globals as a dict of the form: {'<global_name>': value}.
get_labscript_file()	Get the path of the current experiment script
get_run_shots()	Get boolean state of 'Run shot(s)' checkbox
get_shot_output_folder()	Get the current shot output folder
get_shuffle()	Get boolean state of 'Shuffle' checkbox
get_version()	Return the version of runmanager the server is running in
get_view_shots()	Get boolean state of 'View shot(s)' checkbox
instance()	
interrupt([reason])	Interrupt any current and future get*()/push*() calls, causing them to raise Interrupted(reason) until clear_interrupt() is called.
is_output_folder_default()	True if shot output folder is not the default path
n_shots()	Get the number of prospective shots from pressing 'Engage'
request(command, *args, **kwargs)	
reset_shot_output_folder()	Reset the shot output folder to the default path
say_hello()	Ping the runmanager server for a response
set_globals(globals[, raw])	For a dict of the form {'<global_name>': value}, set the given globals to the given values.
set_labscript_file(value)	Set the current experiment script
set_run_shots(value)	Set boolean state of 'Run shot(s)' checkbox
set_shot_output_folder(value)	Set the shot output folder
set_shuffle(value)	Set boolean state of 'Shuffle' checkbox
set_view_shots(value)	Set boolean state of 'View shot(s)' checkbox

abort()

Trigger abort compilation/submission

engage()

Trigger shot compilation/submission

error_in_globals()

True if any tab of an active group contains error(s)

get_globals(*raw=False*)

Return all active globals as a dict of the form: {'<global_name>': value}. If raw=True, then the global values are returned as their string representations, as stored in the runmanager GUI and globals HDF5 file, otherwise they are evaluated as python objects and then returned.

get_labscript_file()

Get the path of the current experiment script

get_run_shots()

Get boolean state of 'Run shot(s)' checkbox

get_shot_output_folder()

Get the current shot output folder

get_shuffle()

Get boolean state of 'Shuffle' checkbox

get_version()

Return the version of runmanager the server is running in

get_view_shots()

Get boolean state of 'View shot(s)' checkbox

is_output_folder_default()

True if shot output folder is not the default path

n_shots()

Get the number of prospective shots from pressing 'Engage'

request(*command*, **args*, ***kwargs*)

reset_shot_output_folder()

Reset the shot output folder to the default path

say_hello()

Ping the runmanager server for a response

set_globals(*globals*, *raw=False*)

For a dict of the form {'<global_name>': value}, set the given globals to the given values. If raw=True, then global values will be treated as the string representations of Python objects rather than the objects themselves, and written directly to the HDF5 file and runmanager GUI without calling repr() on them first.

set_labscript_file(*value*)

Set the current experiment script

set_run_shots(*value*)

Set boolean state of 'Run shot(s)' checkbox

set_shot_output_folder(*value*)

Set the shot output folder

set_shuffle(*value*)

Set boolean state of 'Shuffle' checkbox

set_view_shots(*value*)

Set boolean state of 'View shot(s)' checkbox

3.4 runmanager.batch_compiler

Classes

BatchProcessor(to_parent, from_parent, kill_lock)	

3.4.1 runmanager.batch_compiler.BatchProcessor

class runmanager.batch_compiler.**BatchProcessor**(*to_parent*, *from_parent*, *kill_lock*)

Bases: object

__init__(*to_parent*, *from_parent*, *kill_lock*)

Methods

__init__(to_parent, from_parent, kill_lock)	
compile(labscript_file, run_file)	
mainloop()	

compile(*labscript_file*, *run_file*)

mainloop()

3.5 runmanager.globals_diff

Script that runs *runmanager.globals_diff_shots()* between two shot files.

It is run from the command prompt:

```
$ python runmanager.global_diffs(shot1,shot2)
```

3.6 runmanager.__main__

Runmanager GUI and supporting code

Functions

composite_colors(r0, g0, b0, a0, r1, g1, b1, a1)	composite a second colour over a first with given alpha values and return the result
error_dialog(message)	
log_if_global(g, g_list, message)	logs a message if the global name "g" is in "g_list"
nested(*contextmanagers)	
question_dialog(message)	
scroll_view_to_row_if_current(view, item)	Checks to see if the item is in the row of the current item.

3.6.1 runmanager.__main__.composite_colors

runmanager.__main__.**composite_colors**(*r0*, *g0*, *b0*, *a0*, *r1*, *g1*, *b1*, *a1*)

composite a second colour over a first with given alpha values and return the result

3.6.2 runmanager.__main__.error_dialog

runmanager.__main__.**error_dialog**(*message*)

3.6.3 runmanager.__main__.log_if_global

runmanager.__main__.**log_if_global**(*g*, *g_list*, *message*)

logs a message if the global name “g” is in “g_list”

useful if you want to print out a message inside a loop over globals, but only for a particular global (or set of globals).

If g_list is empty, then it will use the hardcoded list below (useful if you want to change the behaviour globally)

3.6.4 runmanager.__main__.nested

runmanager.__main__.**nested**(**contextmanagers*)

3.6.5 runmanager.__main__.question_dialog

runmanager.__main__.**question_dialog**(*message*)

3.6.6 runmanager.__main__.scroll_view_to_row_if_current

runmanager.__main__.**scroll_view_to_row_if_current**(*view*, *item*)

Checks to see if the item is in the row of the current item. If it is, scrolls the treeview/tableview vertically to ensure that row is visible. This is done by recording the horizontal scroll position, then using view.scrollTo(), and then restoring the horizontal position

Classes

AlternatingColorModel(view)	
Editor(parent)	Popup editor with word wrapping and automatic resizing.
FingerTabBarWidget([parent, minwidth, minheight])	A TabBar with the tabs on the left and the text horizontal.
FingerTabWidget(parent, *args)	A QTabWidget equivalent which uses our FingerTabBarWidget
GroupTab(tabWidget, globals_file, group_name)	
ItemDelegate(*args, **kwargs)	An item delegate with a larger row height and column width, faint grey vertical lines between columns, and a custom editor for handling multi-line data
ItemView(*args)	Mixin for QTableView and QTreeView that emits a custom signal leftClicked(index) after a left click on a valid index, and doubleLeftClicked(index) (in addition) on double click.
PoppedOutOutputBoxWindow	
RemoteServer()	
RunManager()	
RunmanagerMainWindow(*args, **kwargs)	
TabToolButton(*args, **kwargs)	
TableView([parent])	TableView version of our customised ItemView
TreeView([parent])	Treeview version of our customised ItemView

3.6.7 runmanager.__main__.AlternatingColorModel

class runmanager.__main__.**AlternatingColorModel**(*view*)

Bases: QStandardItemModel

__init__(*view*)

Methods

__init__(view)
appendColumn(self, items)
appendRow()
beginInsertColumns(self, parent, first, last)
beginInsertRows(self, parent, first, last)
beginMoveColumns(self, sourceParent, ...)
beginMoveRows(self, sourceParent, ...)
beginRemoveColumns(self, parent, first, last)
beginRemoveRows(self, parent, first, last)
beginResetModel(self)
blockSignals(self, b)
buddy(self, index)
canDropMimeData(self, data, action, row, ...)
canFetchMore(self, parent)
changePersistentIndex(self, from_, to)
changePersistentIndexList(self, from_, to)
checkIndex(self, index[, options])
childEvent(self, a0)
children(self)
clear(self)
clearItemData(self, index)
columnCount(self[, parent])
connectNotify(self, signal)
createIndex(self, row, column[, object])
customEvent(self, a0)

continues on next page

Table 3.2 – continued from previous page

data(index, role)	When background color data is being requested, returns modified colours for every second row, according to the palette of the view.
decodeData(self, row, column, parent, stream)	
deleteLater(self)	
disconnect(-> bool)	
disconnectNotify(self, signal)	
dropMimeData(self, data, action, row, ...)	
dumpObjectInfo(self)	
dumpObjectTree(self)	
dynamicPropertyNames(self)	
encodeData(self, indexes, stream)	
endInsertColumns(self)	
endInsertRows(self)	
endMoveColumns(self)	
endMoveRows(self)	
endRemoveColumns(self)	
endRemoveRows(self)	
endResetModel(self)	
event(self, a0)	
eventFilter(self, a0, a1)	
fetchMore(self, parent)	
findChild(-> QObjectT)	
findChildren(...)	
findItems(self, text[, flags, column])	
flags(self, index)	
get_bgbrush(normal_brush, alternate, selected)	Get cell colour as a function of its ordinary colour, whether it is on an odd row, and whether it is selected.

continues on next page

Table 3.2 – continued from previous page

hasChildren(self[, parent])
hasIndex(self, row, column[, parent])
headerData(self, section, orientation[, role])
horizontalHeaderItem(self, column)
index(self, row, column[, parent])
indexFromItem(self, item)
inherits(self, classname)
insertColumn())
insertColumns(self, column, count[, parent])
insertRow(-> None))
insertRows(self, row, count[, parent])
installEventFilter(self, a0)
invisibleRootItem(self)
isSignalConnected(self, signal)
isWidgetType(self)
isWindowType(self)
item(self, row[, column])
itemData(self, index)
itemFromIndex(self, index)
itemPrototype(self)
killTimer(self, id)
match(self, start, role, value[, hits, flags])
metaObject(self)
mimeData(self, indexes)
mimeTypes(self)
moveColumn(self, sourceParent, sourceColumn, ...)

continues on next page

Table 3.2 – continued from previous page

moveColumns(self, sourceParent, ...)	
moveRow(self, sourceParent, sourceRow, ...)	
moveRows(self, sourceParent, sourceRow, ...)	
moveToThread(self, thread)	
objectName(self)	
parent(-> QModelIndex)	
persistentIndexList(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
receivers(self, signal)	
removeColumn(self, column[, parent])	
removeColumns(self, column, count[, parent])	
removeEventFilter(self, a0)	
removeRow(self, row[, parent])	
removeRows(self, row, count[, parent])	
resetInternalData(self)	
revert(self)	
roleNames(self)	
rowCount(self[, parent])	
sender(self)	
senderSignalIndex(self)	
setColumnCount(self, columns)	
setData(self, index, value[, role])	
setHeaderData(self, section, orientation, value)	
setHorizontalHeaderItem(self, column, item)	
setHorizontalHeaderLabels(self, labels)	

continues on next page

Table 3.2 – continued from previous page

setItem()
setItemData(self, index, roles)
setItemPrototype(self, item)
setItemRoleNames(self, roleNames)
setObjectName(self, name)
setParent(self, a0)
setProperty(self, name, value)
setRowCount(self, rows)
setSortRole(self, role)
setVerticalHeaderItem(self, row, item)
setVerticalHeaderLabels(self, labels)
sibling(self, row, column, idx)
signalsBlocked(self)
sort(self, column[, order])
sortRole(self)
span(self, index)
startTimer(self, interval[, timerType])
submit(self)
supportedDragActions(self)
supportedDropActions(self)
takeColumn(self, column)
takeHorizontalHeaderItem(self, column)
takeItem(self, row[, column])
takeRow(self, row)
takeVerticalHeaderItem(self, row)
thread(self)

continues on next page

Table 3.2 – continued from previous page

timerEvent(self, a0)
tr(self, sourceText[, disambiguation, n])
verticalHeaderItem(self, row)

Attributes

HorizontalSortHint	
NoLayoutChangeHint	
VerticalSortHint	
columnsAboutToBeInserted	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
columnsAboutToBeMoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
columnsAboutToBeRemoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
columnsInserted	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
columnsMoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
columnsRemoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
dataChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
headerDataChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
itemChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
layoutAboutToBeChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
layoutChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
modelAboutToBeReset	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
modelReset	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
rowsAboutToBeInserted	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
rowsAboutToBeMoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
rowsAboutToBeRemoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
rowsInserted	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
rowsMoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
rowsRemoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	

data(*index*, *role*)

When background color data is being requested, returns modified colours for every second row, according to the palette of the view. This has the effect of making the alternate colours visible even when custom colors have been set - the same shading will be applied to the custom colours. Only really looks sensible when the normal and alternate colors are similar. Also applies selection highlight colour (using ItemView.COLOR_HIGHLIGHT), similarly with alternate-row shading, for the case of a QTableView.

get_bgbrush(*normal_brush*, *alternate*, *selected*)

Get cell colour as a function of its ordinary colour, whether it is on an odd row, and whether it is selected.

3.6.8 runmanager.__main__.Editor

class runmanager.__main__.**Editor**(*parent*)

Bases: QTextEdit

Popup editor with word wrapping and automatic resizing.

__init__(*parent*)

Methods

__init__(parent)
acceptDrops(self)
acceptRichText(self)
accessibleDescription(self)
accessibleName(self)
actionEvent(self, a0)
actions(self)
activateWindow(self)
addAction(self, action)
addActions(self, actions)
addScrollBarWidget(self, widget, alignment)
adjustSize(self)
alignment(self)
anchorAt(self, pos)
append(self, text)

continues on next page

Table 3.3 – continued from previous page

autoFillBackground(self)
autoFormatting(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)
canInsertFromMimeData(self, source)
canPaste(self)
changeEvent(self, e)
childAt(-> Optional[QWidget])
childEvent(self, a0)
children(self)
childrenRect(self)
childrenRegion(self)
clear(self)
clearFocus(self)
clearMask(self)
close(self)
closeEvent(self, a0)
colorCount(self)
connectNotify(self, signal)
contentsMargins(self)
contentsRect(self)
contextMenuEvent(self, e)
contextMenuPolicy(self)
copy(self)
cornerWidget(self)

continues on next page

Table 3.3 – continued from previous page

create(self[, window, initializeWindow, ...])	
createMimeDataFromSelection(self)	
createStandardContextMenu(-> Optional[QMenu])	
createWindowContainer(window[, parent, flags])	
currentCharFormat(self)	
currentFont(self)	
cursor(self)	
cursorForPosition(self, pos)	
cursorRect(-> QRect)	
cursorWidth(self)	
customEvent(self, a0)	
cut(self)	
deleteLater(self)	
depth(self)	
destroy(self[, destroyWindow, destroySubWindows])	
devType(self)	
devicePixelRatio(self)	
devicePixelRatioF(self)	
devicePixelRatioFScale()	
disconnect(-> bool)	
disconnectNotify(self, signal)	
document(self)	
documentTitle(self)	
dragEnterEvent(self, e)	
dragLeaveEvent(self, e)	
dragMoveEvent(self, e)	

continues on next page

Table 3.3 – continued from previous page

drawFrame(self, a0)
dropEvent(self, e)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
effectiveWinId(self)
ensureCursorVisible(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, e)
eventFilter(self, a0, a1)
extraSelections(self)
find() -> bool) -> bool))
findChild(-> QObjectT)
findChildren(...)
focusInEvent(self, e)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, e)
focusPolicy(self)
focusPreviousChild(self)
focusProxy(self)
focusWidget(self)
font(self)
fontFamily(self)
fontInfo(self)

continues on next page

Table 3.3 – continued from previous page

fontItalic(self)
fontMetrics(self)
fontPointSize(self)
fontUnderline(self)
fontWeight(self)
foregroundRole(self)
frameGeometry(self)
frameRect(self)
frameShadow(self)
frameShape(self)
frameSize(self)
frameStyle(self)
frameWidth(self)
geometry(self)
getContentsMargins(self)
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])
graphicsEffect(self)
graphicsProxyWidget(self)
hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)
hasTabletTracking(self)

continues on next page

Table 3.3 – continued from previous page

height(self)
heightForWidth(self, a0)
heightMM(self)
hide(self)
hideEvent(self, a0)
horizontalScrollBar(self)
horizontalScrollBarPolicy(self)
inherits(self, classname)
initPainter(self, painter)
initStyleOption(self, option)
inputMethodEvent(self, a0)
inputMethodHints(self)
inputMethodQuery(-> Any)
insertAction(self, before, action)
insertActions(self, before, actions)
insertFromMimeData(self, source)
insertHtml(self, text)
insertPlainText(self, text)
installEventFilter(self, a0)
isActiveWindow(self)
isAncestorOf(self, child)
isEnabled(self)
isEnabledTo(self, a0)
isFullScreen(self)
isHidden(self)
isLeftToRight(self)

continues on next page

Table 3.3 – continued from previous page

isMaximized(self)
isMinimized(self)
isModal(self)
isReadOnly(self)
isRightToLeft(self)
isSignalConnected(self, signal)
isUndoRedoEnabled(self)
isVisible(self)
isVisibleTo(self, a0)
isWidgetType(self)
isWindow(self)
isWindowModified(self)
isWindowType(self)
keyPressEvent(self, e)
keyReleaseEvent(self, e)
keyboardGrabber()
killTimer(self, id)
layout(self)
layoutDirection(self)
leaveEvent(self, a0)
lineWidth(self)
lineWrapColumnOrWidth(self)
lineWrapMode(self)
loadResource(self, type, name)
locale(self)
logicalDpiX(self)

continues on next page

Table 3.3 – continued from previous page

logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)
mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)
maximumViewportSize(self)
maximumWidth(self)
mergeCurrentCharFormat(self, modifier)
metaObject(self)
metric(self, a0)
midLineWidth(self)
minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumWidth(self)
mouseDoubleClickEvent(self, e)
mouseGrabber()
mouseMoveEvent(self, e)
mousePressEvent(self, e)
mouseReleaseEvent(self, e)

continues on next page

Table 3.3 – continued from previous page

move()
moveCursor(self, operation[, mode])
moveEvent(self, a0)
moveToThread(self, thread)
nativeEvent(self, eventType, message)
nativeParentWidget(self)
nextInFocusChain(self)
normalGeometry(self)
objectName(self)
overrideWindowFlags(self, type)
overrideWindowState(self, state)
overwriteMode(self)
paintEngine(self)
paintEvent(self, e)
paintingActive(self)
palette(self)
parent(self)
parentWidget(self)
paste(self)
physicalDpiX(self)
physicalDpiY(self)
placeholderText(self)
pos(self)
previousInFocusChain(self)
print(self, printer)
print_(self, printer)

continues on next page

Table 3.3 – continued from previous page

property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
raise_(self)	
receivers(self, signal)	
rect(self)	
redo(self)	
releaseKeyboard(self)	
releaseMouse(self)	
releaseShortcut(self, id)	
removeAction(self, action)	
removeEventFilter(self, a0)	
render(, sourceRegion, flags, ...)	
repaint(-> None -> None)	
resize()	
resizeEvent(self, a0)	
restoreGeometry(self, geometry)	
saveGeometry(self)	
screen(self)	
scroll()	
scrollBarWidgets(self, alignment)	
scrollContentsBy(self, dx, dy)	
scrollToAnchor(self, name)	
selectAll(self)	
sender(self)	
senderSignalIndex(self)	
setAcceptDrops(self, on)	

continues on next page

Table 3.3 – continued from previous page

setAcceptRichText(self, accept)
setAccessibleDescription(self, description)
setAccessibleName(self, name)
setAlignment(self, a)
setAttribute(self, attribute[, on])
setAutoFillBackground(self, enabled)
setAutoFormatting(self, features)
setBackgroundRole(self, a0)
setBaseSize()
setContentsMargins()
setContextMenuPolicy(self, policy)
setCornerWidget(self, widget)
setCurrentCharFormat(self, format)
setCurrentFont(self, f)
setCursor(self, a0)
setCursorWidth(self, width)
setDisabled(self, a0)
setDocument(self, document)
setDocumentTitle(self, title)
setEnabled(self, a0)
setExtraSelections(self, selections)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()
setFocusPolicy(self, policy)

continues on next page

Table 3.3 – continued from previous page

setFocusProxy(self, a0)
setFont(self, a0)
setFontFamily(self, fontFamily)
setFontItalic(self, b)
setFontPointSize(self, s)
setFontUnderline(self, b)
setFontWeight(self, w)
setForegroundRole(self, a0)
setFrameRect(self, a0)
setFrameShadow(self, a0)
setFrameShape(self, a0)
setFrameStyle(self, a0)
setGeometry()
setGraphicsEffect(self, effect)
setHidden(self, hidden)
setHorizontalScrollBar(self, scrollbar)
setHorizontalScrollBarPolicy(self, a0)
setHtml(self, text)
setInputMethodHints(self, hints)
setLayout(self, a0)
setLayoutDirection(self, direction)
setLineWidth(self, a0)
setLineWrapColumnOrWidth(self, w)
setLineWrapMode(self, mode)
setLocale(self, locale)
setMarkdown(self, markdown)

continues on next page

Table 3.3 – continued from previous page

setMask()
setMaximumHeight(self, maxh)
setMaximumSize()
setMaximumWidth(self, maxw)
setMidLineWidth(self, a0)
setMinimumHeight(self, minh)
setMinimumSize()
setMinimumWidth(self, minw)
setMouseTracking(self, enable)
setObjectName(self, name)
setOverwriteMode(self, overwrite)
setPalette(self, a0)
setParent()
setPlaceholderText(self, placeholderText)
setPlainText(self, text)
setProperty(self, name, value)
setReadOnly(self, ro)
setShortcutAutoRepeat(self, id[, enabled])
setShortcutEnabled(self, id[, enabled])
setSizeAdjustPolicy(self, policy)
setSizeIncrement()
setSizePolicy()
setStatusTip(self, a0)
setStyle(self, a0)
setStyleSheet(self, styleSheet)
setTabChangesFocus(self, b)

continues on next page

Table 3.3 – continued from previous page

setTabOrder(a0, a1)
setTabStopDistance(self, distance)
setTabStopWidth(self, width)
setTabletTracking(self, enable)
setText(self, text)
setTextBackgroundColor(self, c)
setTextColor(self, c)
setTextCursor(self, cursor)
setTextInteractionFlags(self, flags)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setUndoRedoEnabled(self, enable)
setUpdatesEnabled(self, enable)
setVerticalScrollBar(self, scrollbar)
setVerticalScrollBarPolicy(self, a0)
setViewport(self, widget)
setViewportMargins()
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])
setWindowFlags(self, type)
setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)

continues on next page

Table 3.3 – continued from previous page

setWindowOpacity(self, level)
setWindowRole(self, a0)
setWindowState(self, state)
setWindowTitle(self, a0)
setWordWrapMode(self, policy)
setupViewport(self, viewport)
sharedPainter(self)
show(self)
showEvent(self, a0)
showFullScreen(self)
showMaximized(self)
showMinimized(self)
showNormal(self)
signalsBlocked(self)
size(self)
sizeAdjustPolicy(self)
sizeHint(self)
sizeIncrement(self)
sizePolicy(self)
stackUnder(self, a0)
startTimer(self, interval[, timerType])
statusTip(self)
style(self)
styleSheet(self)
tabChangesFocus(self)
tabStopDistance(self)

continues on next page

Table 3.3 – continued from previous page

tabStopWidth(self)
tabletEvent(self, a0)
testAttribute(self, attribute)
textBackgroundColor(self)
textColor(self)
textCursor(self)
textInteractionFlags(self)
thread(self)
timerEvent(self, e)
toHtml(self)
toMarkdown(self[, features])
toPlainText(self)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
underMouse(self)
undo(self)
ungrabGesture(self, type)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)
update(-> None -> None)
updateGeometry(self)
updateMicroFocus(self)
update_size()
updatesEnabled(self)

continues on next page

Table 3.3 – continued from previous page

verticalScrollBar(self)
verticalScrollBarPolicy(self)
viewport(self)
viewportEvent(self, a0)
viewportMargins(self)
viewportSizeHint(self)
visibleRegion(self)
whatsThis(self)
wheelEvent(self, e)
width(self)
widthMM(self)
winId(self)
window(self)
windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)
windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)
windowTitle(self)
windowType(self)
wordWrapMode(self)
x(self)

continues on next page

Table 3.3 – continued from previous page

y(self)
zoomIn(self[, range])
zoomOut(self[, range])

Attributes

AdjustIgnored
AdjustToContents
AdjustToContentsOnFirstShow
AutoAll
AutoBulletList
AutoNone
Box
DrawChildren
DrawWindowBackground
FixedColumnWidth
FixedPixelWidth
HLine
IgnoreMask
NoFrame
NoWrap
Panel
PdmDepth
PdmDevicePixelRatio
PdmDevicePixelRatioScaled
PdmDpiX

continues on next page

Table 3.4 – continued from previous page

PdmDpiY	
PdmHeight	
PdmHeightMM	
PdmNumColors	
PdmPhysicalDpiX	
PdmPhysicalDpiY	
PdmWidth	
PdmWidthMM	
Plain	
Raised	
Shadow_Mask	
Shape_Mask	
StyledPanel	
Sunken	
VLine	
WidgetWidth	
WinPanel	
copyAvailable	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
currentCharFormatChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
cursorPositionChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
customContextMenuRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
redoAvailable	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
selectionChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	

continues on next page

Table 3.4 – continued from previous page

textChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
undoAvailable	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowTitleChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

resizeEvent(*self*, *a0: QResizeEvent | None*)

update_size()

3.6.9 runmanager.__main__.FingerTabBarWidget

class runmanager.__main__.**FingerTabBarWidget**(*parent=None*, *minwidth=180*, *minheight=30*, ***kwargs*)

Bases: QTabBar

A TabBar with the tabs on the left and the text horizontal. Credit to @LegoStormtroopr, https://gist.github.com/LegoStormtroopr/5075267. We will promote the TabBar from the ui file to one of these.

__init__(*parent=None*, *minwidth=180*, *minheight=30*, ***kwargs*)

Methods

__init__([parent, minwidth, minheight])
acceptDrops(self)
accessibleDescription(self)
accessibleName(self)
accessibleTabName(self, index)
actionEvent(self, a0)
actions(self)
activateWindow(self)
addAction(self, action)
addActions(self, actions)
addTab(-> int)

continues on next page

Table 3.5 – continued from previous page

adjustSize(self)
autoFillBackground(self)
autoHide(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)
changeCurrentOnDrag(self)
changeEvent(self, a0)
childAt(-> Optional[QWidget])
childEvent(self, a0)
children(self)
childrenRect(self)
childrenRegion(self)
clearFocus(self)
clearMask(self)
close(self)
closeEvent(self, a0)
colorCount(self)
connectNotify(self, signal)
contentsMargins(self)
contentsRect(self)
contextMenuEvent(self, a0)
contextMenuPolicy(self)
count(self)
create(self[, window, initializeWindow, ...])
createWindowContainer(window[, parent, flags])

continues on next page

Table 3.5 – continued from previous page

currentIndex(self)
cursor(self)
customEvent(self, a0)
deleteLater(self)
depth(self)
destroy(self[, destroyWindow, destroySubWindows])
devType(self)
devicePixelRatio(self)
devicePixelRatioF(self)
devicePixelRatioFScale()
disconnect(-> bool)
disconnectNotify(self, signal)
documentMode(self)
dragEnterEvent(self, a0)
dragLeaveEvent(self, a0)
dragMoveEvent(self, a0)
drawBase(self)
dropEvent(self, a0)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
effectiveWinId(self)
elideMode(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, a0)

continues on next page

Table 3.5 – continued from previous page

eventFilter(self, a0, a1)
expanding(self)
find(a0)
findChild(-> QObjectT)
findChildren(...)
focusInEvent(self, a0)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, a0)
focusPolicy(self)
focusPreviousChild(self)
focusProxy(self)
focusWidget(self)
font(self)
fontInfo(self)
fontMetrics(self)
foregroundRole(self)
frameGeometry(self)
frameSize(self)
geometry(self)
getContentsMargins(self)
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])

continues on next page

Table 3.5 – continued from previous page

graphicsEffect(self)
graphicsProxyWidget(self)
hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)
hasTabletTracking(self)
height(self)
heightForWidth(self, a0)
heightMM(self)
hide(self)
hideEvent(self, a0)
iconSize(self)
indexAtPos(point)
inherits(self, classname)
initPainter(self, painter)
initStyleOption(self, option, tabIndex)
inputMethodEvent(self, a0)
inputMethodHints(self)
inputMethodQuery(self, a0)
insertAction(self, before, action)
insertActions(self, before, actions)
insertTab(-> int)
installEventFilter(self, a0)
isActiveWindow(self)
isAncestorOf(self, child)
isEnabled(self)

continues on next page

Table 3.5 – continued from previous page

isEnabledTo(self, a0)
isFullScreen(self)
isHidden(self)
isLeftToRight(self)
isMaximized(self)
isMinimized(self)
isModal(self)
isMovable(self)
isRightToLeft(self)
isSignalConnected(self, signal)
isTabEnabled(self, index)
isTabVisible(self, index)
isVisible(self)
isVisibleTo(self, a0)
isWidgetType(self)
isWindow(self)
isWindowModified(self)
isWindowType(self)
keyPressEvent(self, a0)
keyReleaseEvent(self, a0)
keyboardGrabber()
killTimer(self, id)
layout(self)
layoutDirection(self)
leaveEvent(self, a0)
locale(self)

continues on next page

Table 3.5 – continued from previous page

logicalDpiX(self)
logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)
mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)
maximumWidth(self)
metaObject(self)
metric(self, a0)
minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumTabSizeHint(self, index)
minimumWidth(self)
mouseDoubleClickEvent(self, a0)
mouseGrabber()
mouseMoveEvent(self, a0)
mousePressEvent(self, a0)
mouseReleaseEvent(self, a0)
move()

continues on next page

Table 3.5 – continued from previous page

moveEvent(self, a0)	
moveTab(self, from_, to)	
moveToThread(self, thread)	
nativeEvent(self, eventType, message)	
nativeParentWidget(self)	
nextInFocusChain(self)	
normalGeometry(self)	
objectName(self)	
overrideWindowFlags(self, type)	
overrideWindowState(self, state)	
paintEngine(self)	
paintEvent(self, a0)	
paintingActive(self)	
palette(self)	
parent(self)	
parentWidget(self)	
physicalDpiX(self)	
physicalDpiY(self)	
pos(self)	
previousInFocusChain(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
raise_(self)	
receivers(self, signal)	
rect(self)	
releaseKeyboard(self)	

continues on next page

Table 3.5 – continued from previous page

releaseMouse(self)
releaseShortcut(self, id)
removeAction(self, action)
removeEventFilter(self, a0)
removeTab(self, index)
render(, sourceRegion, flags, ...)
repaint(-> None -> None)
resize()
resizeEvent(self, a0)
restoreGeometry(self, geometry)
saveGeometry(self)
screen(self)
scroll()
selectionBehaviorOnRemove(self)
sender(self)
senderSignalIndex(self)
setAcceptDrops(self, on)
setAccessibleDescription(self, description)
setAccessibleName(self, name)
setAccessibleTabName(self, index, name)
setAttribute(self, attribute[, on])
setAutoFillBackground(self, enabled)
setAutoHide(self, hide)
setBackgroundRole(self, a0)
setBaseSize()
setChangeCurrentOnDrag(self, change)

continues on next page

Table 3.5 – continued from previous page

setContentsMargins()
setContextMenuPolicy(self, policy)
setCurrentIndex(self, index)
setCursor(self, a0)
setDisabled(self, a0)
setDocumentMode(self, set)
setDrawBase(self, drawTheBase)
setElideMode(self, a0)
setEnabled(self, a0)
setExpanding(self, enabled)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()
setFocusPolicy(self, policy)
setFocusProxy(self, a0)
setFont(self, a0)
setForegroundRole(self, a0)
setGeometry()
setGraphicsEffect(self, effect)
setHidden(self, hidden)
setIconSize(self, size)
setInputMethodHints(self, hints)
setLayout(self, a0)
setLayoutDirection(self, direction)
setLocale(self, locale)

continues on next page

Table 3.5 – continued from previous page

setMask()	
setMaximumHeight(self, maxh)	
setMaximumSize()	
setMaximumWidth(self, maxw)	
setMinimumHeight(self, minh)	
setMinimumSize()	
setMinimumWidth(self, minw)	
setMouseTracking(self, enable)	
setMovable(movable[, index])	Set tabs movable on an individual basis, or set for all tabs if no index specified
setObjectName(self, name)	
setPalette(self, a0)	
setParent()	
setProperty(self, name, value)	
setSelectionBehaviorOnRemove(self, behavior)	
setShape(self, shape)	
setShortcutAutoRepeat(self, id[, enabled])	
setShortcutEnabled(self, id[, enabled])	
setSizeIncrement()	
setSizePolicy()	
setStatusTip(self, a0)	
setStyle(self, a0)	
setStyleSheet(self, styleSheet)	
setTabButton(self, index, position, widget)	
setTabData(self, index, data)	
setTabEnabled(self, index, a1)	
setTabIcon(self, index, icon)	

continues on next page

Table 3.5 – continued from previous page

setTabOrder(a0, a1)
setTabText(self, index, text)
setTabTextColor(self, index, color)
setTabToolTip(self, index, tip)
setTabVisible(self, index, visible)
setTabWhatsThis(self, index, text)
setTabletTracking(self, enable)
setTabsClosable(self, closable)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setUpdatesEnabled(self, enable)
setUsesScrollButtons(self, useButtons)
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])
setWindowFlags(self, type)
setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)
setWindowOpacity(self, level)
setWindowRole(self, a0)
setWindowState(self, state)
setWindowTitle(self, a0)
shape(self)

continues on next page

Table 3.5 – continued from previous page

sharedPainter(self)
show(self)
showEvent(self, a0)
showFullScreen(self)
showMaximized(self)
showMinimized(self)
showNormal(self)
signalsBlocked(self)
size(self)
sizeHint(self)
sizeIncrement(self)
sizePolicy(self)
stackUnder(self, a0)
startTimer(self, interval[, timerType])
statusTip(self)
style(self)
styleSheet(self)
tabAt(self, pos)
tabButton(self, index, position)
tabData(self, index)
tabIcon(self, index)
tabInserted(self, index)
tabLayoutChange(self)
tabRect(self, index)
tabRemoved(self, index)
tabSizeHint(self, index)

continues on next page

Table 3.5 – continued from previous page

tabText(self, index)
tabTextColor(self, index)
tabToolTip(self, index)
tabWhatsThis(self, index)
tabletEvent(self, a0)
tabsClosable(self)
testAttribute(self, attribute)
thread(self)
timerEvent(self, event)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
underMouse(self)
ungrabGesture(self, type)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)
update(-> None -> None)
updateGeometry(self)
updateMicroFocus(self)
updatesEnabled(self)
usesScrollButtons(self)
visibleRegion(self)
whatsThis(self)
wheelEvent(self, event)
width(self)

continues on next page

Table 3.5 – continued from previous page

widthMM(self)
winId(self)
window(self)
windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)
windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)
windowTitle(self)
windowType(self)
x(self)
y(self)

Attributes

DrawChildren
DrawWindowBackground
IgnoreMask
LeftSide
PdmDepth
PdmDevicePixelRatio
PdmDevicePixelRatioScaled

continues on next page

Table 3.6 – continued from previous page

PdmDpiX	
PdmDpiY	
PdmHeight	
PdmHeightMM	
PdmNumColors	
PdmPhysicalDpiX	
PdmPhysicalDpiY	
PdmWidth	
PdmWidthMM	
RightSide	
RoundedEast	
RoundedNorth	
RoundedSouth	
RoundedWest	
SelectLeftTab	
SelectPreviousTab	
SelectRightTab	
TriangularEast	
TriangularNorth	
TriangularSouth	
TriangularWest	
currentChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
customContextMenuRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	

continues on next page

Table 3.6 – continued from previous page

tabBarClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
tabBarDoubleClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
tabCloseRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
tabMoved	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowTitleChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

indexAtPos(*point*)

isMovable(*self*) → bool

mousePressEvent(*self*, *a0: QMouseEvent* | *None*)

mouseReleaseEvent(*self*, *a0: QMouseEvent* | *None*)

paintEvent(*self*, *a0: QPaintEvent* | *None*)

setMovable(*movable*, *index=None*)

Set tabs movable on an individual basis, or set for all tabs if no index specified

setTabButton(*self*, *index: int*, *position: QTabBar.ButtonPosition*, *widget: QWidget* | *None*)

tabLayoutChange(*self*)

tabSizeHint(*self*, *index: int*) → QSize

3.6.10 runmanager.__main__.FingerTabWidget

class runmanager.__main__.**FingerTabWidget**(*parent*, **args*)

Bases: QTabWidget

A QTabWidget equivalent which uses our FingerTabBarWidget

__init__(*parent*, **args*)

Methods

__init__(parent, *args)
acceptDrops(self)
accessibleDescription(self)

continues on next page

Table 3.7 – continued from previous page

accessibleName(self)
actionEvent(self, a0)
actions(self)
activateWindow(self)
addAction(self, action)
addActions(self, actions)
addTab(-> int)
adjustSize(self)
autoFillBackground(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)
changeEvent(self, a0)
childAt(-> Optional[QWidget])
childEvent(self, a0)
children(self)
childrenRect(self)
childrenRegion(self)
clear(self)
clearFocus(self)
clearMask(self)
close(self)
closeEvent(self, a0)
colorCount(self)
connectNotify(self, signal)
contentsMargins(self)

continues on next page

Table 3.7 – continued from previous page

contentsRect(self)	
contextMenuEvent(self, a0)	
contextMenuPolicy(self)	
cornerWidget(self[, corner])	
count(self)	
create(self[, window, initializeWindow, ...])	
createWindowContainer(window[, parent, flags])	
currentIndex(self)	
currentWidget(self)	
cursor(self)	
customEvent(self, a0)	
deleteLater(self)	
depth(self)	
destroy(self[, destroyWindow, destroySubWindows])	
devType(self)	
devicePixelRatio(self)	
devicePixelRatioF(self)	
devicePixelRatioFScale()	
disconnect(-> bool)	
disconnectNotify(self, signal)	
documentMode(self)	
dragEnterEvent(self, a0)	
dragLeaveEvent(self, a0)	
dragMoveEvent(self, a0)	
dropEvent(self, a0)	
dumpObjectInfo(self)	

continues on next page

Table 3.7 – continued from previous page

dumpObjectTree(self)
dynamicPropertyNames(self)
effectiveWinId(self)
elideMode(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, a0)
eventFilter(self, a0, a1)
find(a0)
findChild(-> QObjectT)
findChildren(...)
focusInEvent(self, a0)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, a0)
focusPolicy(self)
focusPreviousChild(self)
focusProxy(self)
focusWidget(self)
font(self)
fontInfo(self)
fontMetrics(self)
foregroundRole(self)
frameGeometry(self)
frameSize(self)
geometry(self)

continues on next page

Table 3.7 – continued from previous page

getContentsMargins(self)
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])
graphicsEffect(self)
graphicsProxyWidget(self)
hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)
hasTabletTracking(self)
height(self)
heightForWidth(self, width)
heightMM(self)
hide(self)
hideEvent(self, a0)
iconSize(self)
indexOf(self, widget)
inherits(self, classname)
initPainter(self, painter)
initStyleOption(self, option)
inputMethodEvent(self, a0)
inputMethodHints(self)
inputMethodQuery(self, a0)
insertAction(self, before, action)

continues on next page

Table 3.7 – continued from previous page

insertActions(self, before, actions)
insertTab(-> int)
installEventFilter(self, a0)
isActiveWindow(self)
isAncestorOf(self, child)
isEnabled(self)
isEnabledTo(self, a0)
isFullScreen(self)
isHidden(self)
isLeftToRight(self)
isMaximized(self)
isMinimized(self)
isModal(self)
isMovable(self)
isRightToLeft(self)
isSignalConnected(self, signal)
isTabEnabled(self, index)
isTabVisible(self, index)
isVisible(self)
isVisibleTo(self, a0)
isWidgetType(self)
isWindow(self)
isWindowModified(self)
isWindowType(self)
keyPressEvent(self, a0)
keyReleaseEvent(self, a0)

continues on next page

Table 3.7 – continued from previous page

keyboardGrabber()
killTimer(self, id)
layout(self)
layoutDirection(self)
leaveEvent(self, a0)
locale(self)
logicalDpiX(self)
logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)
mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)
maximumWidth(self)
metaObject(self)
metric(self, a0)
minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumWidth(self)
mouseDoubleClickEvent(self, a0)

continues on next page

Table 3.7 – continued from previous page

mouseGrabber()	
mouseMoveEvent(self, a0)	
mousePressEvent(self, a0)	
mouseReleaseEvent(self, a0)	
move()	
moveEvent(self, a0)	
moveToThread(self, thread)	
nativeEvent(self, eventType, message)	
nativeParentWidget(self)	
nextInFocusChain(self)	
normalGeometry(self)	
objectName(self)	
overrideWindowFlags(self, type)	
overrideWindowState(self, state)	
paintEngine(self)	
paintEvent(self, a0)	
paintingActive(self)	
palette(self)	
parent(self)	
parentWidget(self)	
physicalDpiX(self)	
physicalDpiY(self)	
pos(self)	
previousInFocusChain(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.

continues on next page

Table 3.7 – continued from previous page

raise_(self)
receivers(self, signal)
rect(self)
releaseKeyboard(self)
releaseMouse(self)
releaseShortcut(self, id)
removeAction(self, action)
removeEventFilter(self, a0)
removeTab(self, index)
render(, sourceRegion, flags, ...)
repaint(-> None -> None)
resize()
resizeEvent(self, a0)
restoreGeometry(self, geometry)
saveGeometry(self)
screen(self)
scroll()
sender(self)
senderSignalIndex(self)
setAcceptDrops(self, on)
setAccessibleDescription(self, description)
setAccessibleName(self, name)
setAttribute(self, attribute[, on])
setAutoFillBackground(self, enabled)
setBackgroundRole(self, a0)
setBaseSize()

continues on next page

Table 3.7 – continued from previous page

setContentsMargins()
setContextMenuPolicy(self, policy)
setCornerWidget(self, widget[, corner])
setCurrentIndex(self, index)
setCurrentWidget(self, widget)
setCursor(self, a0)
setDisabled(self, a0)
setDocumentMode(self, set)
setElideMode(self, a0)
setEnabled(self, a0)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()
setFocusPolicy(self, policy)
setFocusProxy(self, a0)
setFont(self, a0)
setForegroundRole(self, a0)
setGeometry()
setGraphicsEffect(self, effect)
setHidden(self, hidden)
setIconSize(self, size)
setInputMethodHints(self, hints)
setLayout(self, a0)
setLayoutDirection(self, direction)
setLocale(self, locale)

continues on next page

Table 3.7 – continued from previous page

setMask()
setMaximumHeight(self, maxh)
setMaximumSize()
setMaximumWidth(self, maxw)
setMinimumHeight(self, minh)
setMinimumSize()
setMinimumWidth(self, minw)
setMouseTracking(self, enable)
setMovable(self, movable)
setObjectName(self, name)
setPalette(self, a0)
setParent()
setProperty(self, name, value)
setShortcutAutoRepeat(self, id[, enabled])
setShortcutEnabled(self, id[, enabled])
setSizeIncrement()
setSizePolicy()
setStatusTip(self, a0)
setStyle(self, a0)
setStyleSheet(self, styleSheet)
setTabBar(self, a0)
setTabBarAutoHide(self, enabled)
setTabClosable(index, closable)
setTabEnabled(self, index, a1)
setTabIcon(self, index, icon)
setTabOrder(a0, a1)

continues on next page

Table 3.7 – continued from previous page

setTabPosition(self, a0)
setTabShape(self, s)
setTabText(self, index, a1)
setTabToolTip(self, index, tip)
setTabVisible(self, index, visible)
setTabWhatsThis(self, index, text)
setTabletTracking(self, enable)
setTabsClosable(self, closeable)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setUpdatesEnabled(self, enable)
setUsesScrollButtons(self, useButtons)
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])
setWindowFlags(self, type)
setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)
setWindowOpacity(self, level)
setWindowRole(self, a0)
setWindowState(self, state)
setWindowTitle(self, a0)
sharedPainter(self)

continues on next page

Table 3.7 – continued from previous page

show(self)
showEvent(self, a0)
showFullScreen(self)
showMaximized(self)
showMinimized(self)
showNormal(self)
signalsBlocked(self)
size(self)
sizeHint(self)
sizeIncrement(self)
sizePolicy(self)
stackUnder(self, a0)
startTimer(self, interval[, timerType])
statusTip(self)
style(self)
styleSheet(self)
tabBar(self)
tabBarAutoHide(self)
tabIcon(self, index)
tabInserted(self, index)
tabPosition(self)
tabRemoved(self, index)
tabShape(self)
tabText(self, index)
tabToolTip(self, index)
tabWhatsThis(self, index)

continues on next page

Table 3.7 – continued from previous page

tabletEvent(self, a0)
tabsClosable(self)
testAttribute(self, attribute)
thread(self)
timerEvent(self, a0)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
underMouse(self)
ungrabGesture(self, type)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)
update(-> None -> None)
updateGeometry(self)
updateMicroFocus(self)
updatesEnabled(self)
usesScrollButtons(self)
visibleRegion(self)
whatsThis(self)
wheelEvent(self, a0)
widget(self, index)
width(self)
widthMM(self)
winId(self)
window(self)

continues on next page

Table 3.7 – continued from previous page

windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)
windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)
windowTitle(self)
windowType(self)
x(self)
y(self)

Attributes

DrawChildren
DrawWindowBackground
East
IgnoreMask
North
PdmDepth
PdmDevicePixelRatio
PdmDevicePixelRatioScaled
PdmDpiX
PdmDpiY

continues on next page

Table 3.8 – continued from previous page

PdmHeight			
PdmHeightMM			
PdmNumColors			
PdmPhysicalDpiX			
PdmPhysicalDpiY			
PdmWidth			
PdmWidthMM			
Rounded			
South			
Triangular			
West			
currentChanged	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
customContextMenuRequested	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
destroyed	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
objectNameChanged	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
staticMetaObject			
tabBarClicked	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
tabBarDoubleClicked	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
tabCloseRequested	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
windowIconChanged	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
windowTitleChanged	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL

addTab(*self*, *widget: QWidget | None*, *a1: str | None*) → int

addTab(*self*, *widget: QWidget | None*, *icon: QIcon*, *label: str | None*) → int

setTabClosable(*index*, *closable*)

3.6.11 runmanager.__main__.GroupTab

class runmanager.__main__.**GroupTab**(*tabWidget*, *globals_file*, *group_name*)

Bases: object

__init__(*tabWidget*, *globals_file*, *group_name*)

Methods

__init__(tabWidget, globals_file, group_name)	
change_global_expansion(global_name, ...)	
change_global_units(global_name, ...)	
change_global_value(global_name, ...[, ...])	
check_for_boolean_values(item)	Checks if the value is 'True' or 'False'.
close()	
complete_change_global_value(global_name, ...)	
connect_signals()	
delete_global(global_name[, confirm])	
do_model_sort()	
get_global_item_by_name(global_name, column)	Returns an item from the row representing a global in the globals model.
globals_changed()	Called whenever something about a global has changed.
make_global_row(name[, value, units, expansion])	
new_global(global_name)	
on_globals_delete_selected_triggered()	
on_globals_model_expansion_changed(item)	
on_globals_model_item_changed(item)	
on_globals_model_name_changed(item)	Handles global renaming and creation of new globals due to the user editing the <click to add global> item
on_globals_model_units_changed(item)	
on_globals_model_value_changed(item)	
on_globals_set_selected_bools_triggered(st	
on_tableView_globals_context_menu_request	
on_tableView_globals_leftClicked(index)	
populate_model()	
rename_global(previous_global_name, ...)	
set_file_and_group_name(globals_file, group_name)	Provided as a separate method so the main app can call it if the group gets renamed
set_tab_icon(icon_string)	
update_parse_indication(active_groups, ...)	

Attributes

COLOR_BOOL_OFF
COLOR_BOOL_ON
COLOR_ERROR
COLOR_OK
GLOBALS_COL_DELETE
GLOBALS_COL_EXPANSION
GLOBALS_COL_NAME
GLOBALS_COL_UNITS
GLOBALS_COL_VALUE
GLOBALS_DUMMY_ROW_TEXT
GLOBALS_ROLE_IS_BOOL
GLOBALS_ROLE_IS_DUMMY_ROW
GLOBALS_ROLE_PREVIOUS_TEXT
GLOBALS_ROLE_SORT_DATA

COLOR_BOOL_OFF = '#608060'

COLOR_BOOL_ON = '#63F731'

COLOR_ERROR = '#F79494'

COLOR_OK = '#A5F7C6'

GLOBALS_COL_DELETE = 0

GLOBALS_COL_EXPANSION = 4

GLOBALS_COL_NAME = 1

GLOBALS_COL_UNITS = 3

GLOBALS_COL_VALUE = 2

GLOBALS_DUMMY_ROW_TEXT = '<Click to add global>'

GLOBALS_ROLE_IS_BOOL = 260

GLOBALS_ROLE_IS_DUMMY_ROW = 257

GLOBALS_ROLE_PREVIOUS_TEXT = 259

GLOBALS_ROLE_SORT_DATA = 258

change_global_expansion(*global_name*, *previous_expansion*, *new_expansion*)

change_global_units(*global_name*, *previous_units*, *new_units*)

change_global_value(*global_name*, *previous_value*, *new_value*, *interactive=True*)

check_for_boolean_values(*item*)

Checks if the value is 'True' or 'False'. If either, makes the units cell checkable, uneditable, and coloured to indicate the state. The units cell can then be clicked to toggle the value.

close()

complete_change_global_value(*global_name*, *previous_value*, *new_value*, *item*, *previous_background*, *previous_icon*, *interactive=True*)

connect_signals()

delete_global(*global_name*, *confirm=True*)

do_model_sort()

get_global_item_by_name(*global_name*, *column*, *previous_name=None*)

Returns an item from the row representing a global in the globals model. Which item is returned is set by the column argument.

globals_changed()

Called whenever something about a global has changed. call app.globals_changed to inform the main application that it needs to parse globals again. self.update_parse_indication will be called by the main app when parsing is done, and will set the colours and tooltips appropriately

make_global_row(*name*, *value=''*, *units=''*, *expansion=''*)

new_global(*global_name*)

on_globals_delete_selected_triggered()

on_globals_model_expansion_changed(*item*)

on_globals_model_item_changed(*item*)

on_globals_model_name_changed(*item*)

Handles global renaming and creation of new globals due to the user editing the <click to add global> item

on_globals_model_units_changed(*item*)

on_globals_model_value_changed(*item*)

on_globals_set_selected_bools_triggered(*state*)

on_tableView_globals_context_menu_requested(*point*)

on_tableView_globals_leftClicked(*index*)

populate_model()

rename_global(*previous_global_name*, *new_global_name*)

set_file_and_group_name(*globals_file*, *group_name*)

Provided as a separate method so the main app can call it if the group gets renamed

set_tab_icon(*icon_string*)

update_parse_indication(*active_groups*, *sequence_globals*, *evaled_globals*)

3.6.12 runmanager.__main__.ItemDelegate

class runmanager.__main__.**ItemDelegate**(**args*, ***kwargs*)

Bases: QStyledItemDelegate

An item delegate with a larger row height and column width, faint grey vertical lines between columns, and a custom editor for handling multi-line data

__init__(**args*, ***kwargs*)

Methods

__init__(*args, **kwargs)
blockSignals(self, b)
childEvent(self, a0)
children(self)
connectNotify(self, signal)
createEditor(self, parent, option, index)
customEvent(self, a0)
deleteLater(self)
destroyEditor(self, editor, index)
disconnect(-> bool)
disconnectNotify(self, signal)
displayText(self, value, locale)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
editorEvent(self, event, model, option, index)

continues on next page

Table 3.9 – continued from previous page

event(self, a0)	
eventFilter(obj, event)	Filter events before they get to the editor, so that editing is ended when the user presses tab, shift-tab or enter (which otherwise would not end editing in a QTextEdit).
findChild(-> QObjectT)	
findChildren(...)	
helpEvent(self, event, view, option, index)	
inherits(self, classname)	
initStyleOption(self, option, index)	
installEventFilter(self, a0)	
isSignalConnected(self, signal)	
isWidgetType(self)	
isWindowType(self)	
itemEditorFactory(self)	
killTimer(self, id)	
metaObject(self)	
moveToThread(self, thread)	
objectName(self)	
paint(self, painter, option, index)	
parent(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
receivers(self, signal)	
removeEventFilter(self, a0)	
sender(self)	
senderSignalIndex(self)	
setEditorData(self, editor, index)	

continues on next page

Table 3.9 – continued from previous page

setItemEditorFactory(self, factory)
setModelData(self, editor, model, index)
setObjectName(self, name)
setParent(self, a0)
setProperty(self, name, value)
signalsBlocked(self)
sizeHint(self, option, index)
startTimer(self, interval[, timerType])
thread(self)
timerEvent(self, a0)
tr(self, sourceText[, disambiguation, n])
updateEditorGeometry(self, editor, option, index)

Attributes

EXTRA_COL_WIDTH			
EXTRA_ROW_HEIGHT			
EditNextItem			
EditPreviousItem			
MIN_ROW_HEIGHT			
NoHint			
RevertModelCache			
SubmitModelCache			
closeEditor	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
commitData	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
destroyed	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
objectNameChanged	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
sizeHintChanged	int = ..., arguments:	Sequence = ...)	-> PYQT_SIGNAL
staticMetaObject			

EXTRA_COL_WIDTH = 20

EXTRA_ROW_HEIGHT = 6

MIN_ROW_HEIGHT = 22

createEditor(*self*, *parent: QWidget | None*, *option: QStyleOptionViewItem*, *index: QModelIndex*) → QWidget | None

eventFilter(*obj*, *event*)

Filter events before they get to the editor, so that editing is ended when the user presses tab, shift-tab or enter (which otherwise would not end editing in a QTextEdit).

paint(*self*, *painter: QPainter | None*, *option: QStyleOptionViewItem*, *index: QModelIndex*)

setEditorData(*self*, *editor: QWidget | None*, *index: QModelIndex*)

setModelData(*self*, *editor: QWidget | None*, *model: QAbstractItemModel | None*, *index: QModelIndex*)

sizeHint(*self*, *option: QStyleOptionViewItem*, *index: QModelIndex*) → QSize

3.6.13 runmanager.__main__.ItemView

class runmanager.__main__.**ItemView**(**args*)

Bases: object

Mixin for QTableView and QTreeView that emits a custom signal leftClicked(index) after a left click on a valid index, and doubleLeftClicked(index) (in addition) on double click. Also has modified tab and arrow key behaviour and custom selection highlighting.

__init__(**args*)

Methods

__init__(*args)	
keyPressEvent(event)	
leaveEvent(event)	
mouseDoubleClickEvent(event)	
mousePressEvent(event)	
mouseReleaseEvent(event)	
moveCursor(cursor_action, keyboard_modifiers)	

Attributes

COLOR_HIGHLIGHT	
doubleLeftClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
leftClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

COLOR_HIGHLIGHT = '#40308CC6'

doubleLeftClicked

int = . . . , arguments: Sequence = . . .) -> PYQT_SIGNAL

types is normally a sequence of individual types. Each type is either a type object or a string that is the name of a C++ type. Alternatively each type could itself be a sequence of types each describing a different overloaded signal. name is the optional C++ name of the signal. If it is not specified then the name of the class attribute that is bound to the signal is used. revision is the optional revision of the signal that is exported to QML. If it is not specified then 0 is used. arguments is the optional sequence of the names of the signal's arguments.

Type
pyqtSignal(*types, name

Type
str = ..., revision

keyPressEvent(*event*)

leaveEvent(*event*)

leftClicked

int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

types is normally a sequence of individual types. Each type is either a type object or a string that is the name of a C++ type. Alternatively each type could itself be a sequence of types each describing a different overloaded signal. name is the optional C++ name of the signal. If it is not specified then the name of the class attribute that is bound to the signal is used. revision is the optional revision of the signal that is exported to QML. If it is not specified then 0 is used. arguments is the optional sequence of the names of the signal's arguments.

Type
pyqtSignal(*types, name

Type
str = ..., revision

mouseDoubleClickEvent(*event*)

mousePressEvent(*event*)

mouseReleaseEvent(*event*)

moveCursor(*cursor_action*, *keyboard_modifiers*)

3.6.14 runmanager.__main__.PoppedOutOutputBoxWindow

class runmanager.__main__.**PoppedOutOutputBoxWindow**

Bases: QDialog

__init__(**args*, ***kwargs*)

Methods

__init__(*args, **kwargs)
accept(self)
acceptDrops(self)
accessibleDescription(self)
accessibleName(self)
actionEvent(self, a0)
actions(self)

continues on next page

Table 3.10 – continued from previous page

activateWindow(self)
addAction(self, action)
addActions(self, actions)
adjustSize(self)
autoFillBackground(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)
changeEvent(self, a0)
childAt(-> Optional[QWidget])
childEvent(self, a0)
children(self)
childrenRect(self)
childrenRegion(self)
clearFocus(self)
clearMask(self)
close(self)
closeEvent(self, a0)
colorCount(self)
connectNotify(self, signal)
contentsMargins(self)
contentsRect(self)
contextMenuEvent(self, a0)
contextMenuPolicy(self)
create(self[, window, initializeWindow, ...])
createWindowContainer(window[, parent, flags])

continues on next page

Table 3.10 – continued from previous page

cursor(self)
customEvent(self, a0)
deleteLater(self)
depth(self)
destroy(self[, destroyWindow, destroySubWindows])
devType(self)
devicePixelRatio(self)
devicePixelRatioF(self)
devicePixelRatioFScale()
disconnect(-> bool)
disconnectNotify(self, signal)
done(self, a0)
dragEnterEvent(self, a0)
dragLeaveEvent(self, a0)
dragMoveEvent(self, a0)
dropEvent(self, a0)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
effectiveWinId(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, a0)
eventFilter(self, a0, a1)
exec(self)
exec_(self)

continues on next page

Table 3.10 – continued from previous page

find(a0)
findChild(-> QObjectT)
findChildren(...)
focusInEvent(self, a0)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, a0)
focusPolicy(self)
focusPreviousChild(self)
focusProxy(self)
focusWidget(self)
font(self)
fontInfo(self)
fontMetrics(self)
foregroundRole(self)
frameGeometry(self)
frameSize(self)
geometry(self)
getContentsMargins(self)
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])
graphicsEffect(self)
graphicsProxyWidget(self)

continues on next page

Table 3.10 – continued from previous page

hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)
hasTabletTracking(self)
height(self)
heightForWidth(self, a0)
heightMM(self)
hide(self)
hideEvent(self, a0)
inherits(self, classname)
initPainter(self, painter)
inputMethodEvent(self, a0)
inputMethodHints(self)
inputMethodQuery(self, a0)
insertAction(self, before, action)
insertActions(self, before, actions)
installEventFilter(self, a0)
isActiveWindow(self)
isAncestorOf(self, child)
isEnabled(self)
isEnabledTo(self, a0)
isFullScreen(self)
isHidden(self)
isLeftToRight(self)
isMaximized(self)
isMinimized(self)

continues on next page

Table 3.10 – continued from previous page

isModal(self)
isRightToLeft(self)
isSignalConnected(self, signal)
isSizeGripEnabled(self)
isVisible(self)
isVisibleTo(self, a0)
isWidgetType(self)
isWindow(self)
isWindowModified(self)
isWindowType(self)
keyPressEvent(self, a0)
keyReleaseEvent(self, a0)
keyboardGrabber()
killTimer(self, id)
layout(self)
layoutDirection(self)
leaveEvent(self, a0)
locale(self)
logicalDpiX(self)
logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)

continues on next page

Table 3.10 – continued from previous page

mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)
maximumWidth(self)
metaObject(self)
metric(self, a0)
minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumWidth(self)
mouseDoubleClickEvent(self, a0)
mouseGrabber()
mouseMoveEvent(self, a0)
mousePressEvent(self, a0)
mouseReleaseEvent(self, a0)
move()
moveEvent(self, a0)
moveToThread(self, thread)
nativeEvent(self, eventType, message)
nativeParentWidget(self)
nextInFocusChain(self)
normalGeometry(self)
objectName(self)
open(self)
overrideWindowFlags(self, type)

continues on next page

Table 3.10 – continued from previous page

overrideWindowState(self, state)	
paintEngine(self)	
paintEvent(self, a0)	
paintingActive(self)	
palette(self)	
parent(self)	
parentWidget(self)	
physicalDpiX(self)	
physicalDpiY(self)	
pos(self)	
previousInFocusChain(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
raise_(self)	
receivers(self, signal)	
rect(self)	
reject(self)	
releaseKeyboard(self)	
releaseMouse(self)	
releaseShortcut(self, id)	
removeAction(self, action)	
removeEventFilter(self, a0)	
render(, sourceRegion, flags, ...)	
repaint(-> None -> None)	
resize()	
resizeEvent(self, a0)	

continues on next page

Table 3.10 – continued from previous page

restoreGeometry(self, geometry)
result(self)
saveGeometry(self)
screen(self)
scroll()
sender(self)
senderSignalIndex(self)
setAcceptDrops(self, on)
setAccessibleDescription(self, description)
setAccessibleName(self, name)
setAttribute(self, attribute[, on])
setAutoFillBackground(self, enabled)
setBackgroundRole(self, a0)
setBaseSize()
setContentsMargins()
setContextMenuPolicy(self, policy)
setCursor(self, a0)
setDisabled(self, a0)
setEnabled(self, a0)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()
setFocusPolicy(self, policy)
setFocusProxy(self, a0)
setFont(self, a0)

continues on next page

Table 3.10 – continued from previous page

setForegroundRole(self, a0)
setGeometry()
setGraphicsEffect(self, effect)
setHidden(self, hidden)
setInputMethodHints(self, hints)
setLayout(self, a0)
setLayoutDirection(self, direction)
setLocale(self, locale)
setMask()
setMaximumHeight(self, maxh)
setMaximumSize()
setMaximumWidth(self, maxw)
setMinimumHeight(self, minh)
setMinimumSize()
setMinimumWidth(self, minw)
setModal(self, modal)
setMouseTracking(self, enable)
setObjectName(self, name)
setPalette(self, a0)
setParent()
setProperty(self, name, value)
setResult(self, r)
setShortcutAutoRepeat(self, id[, enabled])
setShortcutEnabled(self, id[, enabled])
setSizeGripEnabled(self, a0)
setSizeIncrement()

continues on next page

Table 3.10 – continued from previous page

setSizePolicy()
setStatusTip(self, a0)
setStyle(self, a0)
setStyleSheet(self, styleSheet)
setTabOrder(a0, a1)
setTabletTracking(self, enable)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setUpdatesEnabled(self, enable)
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])
setWindowFlags(self, type)
setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)
setWindowOpacity(self, level)
setWindowRole(self, a0)
setWindowState(self, state)
setWindowTitle(self, a0)
sharedPainter(self)
show(self)
showEvent(self, a0)
showFullScreen(self)

continues on next page

Table 3.10 – continued from previous page

showMaximized(self)
showMinimized(self)
showNormal(self)
signalsBlocked(self)
size(self)
sizeHint(self)
sizeIncrement(self)
sizePolicy(self)
stackUnder(self, a0)
startTimer(self, interval[, timerType])
statusTip(self)
style(self)
styleSheet(self)
tabletEvent(self, a0)
testAttribute(self, attribute)
thread(self)
timerEvent(self, a0)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
underMouse(self)
ungrabGesture(self, type)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)
update(-> None -> None)

continues on next page

Table 3.10 – continued from previous page

updateGeometry(self)
updateMicroFocus(self)
updatesEnabled(self)
visibleRegion(self)
whatsThis(self)
wheelEvent(self, a0)
width(self)
widthMM(self)
winId(self)
window(self)
windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)
windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)
windowTitle(self)
windowType(self)
x(self)
y(self)

Attributes

Accepted	
DrawChildren	
DrawWindowBackground	
IgnoreMask	
PdmDepth	
PdmDevicePixelRatio	
PdmDevicePixelRatioScaled	
PdmDpiX	
PdmDpiY	
PdmHeight	
PdmHeightMM	
PdmNumColors	
PdmPhysicalDpiX	
PdmPhysicalDpiY	
PdmWidth	
PdmWidthMM	
Rejected	
accepted	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
customContextMenuRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
finished	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
rejected	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	
windowIconChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowTitleChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

closeEvent(*self*, *a0: QCloseEvent | None*)

3.6.15 runmanager.__main__.RemoteServer

class runmanager.__main__.RemoteServer

Bases: ZMQServer

__init__()

Methods

__init__()	
handle_abort()	
handle_engage()	
handle_error_in_globals()	
handle_get_globals([raw])	
handle_get_labscript_file()	
handle_get_run_shots()	
handle_get_shot_output_folder()	
handle_get_shuffle()	
handle_get_view_shots()	
handle_is_output_folder_default()	
handle_n_shots()	
handle_reset_shot_output_folder()	
handle_set_globals(globals[, raw])	
handle_set_labscript_file(value)	
handle_set_run_shots(value)	
handle_set_shot_output_folder(value)	
handle_set_shuffle(value)	
handle_set_view_shots(value)	
handler(request_data)	To be overridden by subclasses.
mainloop()	
setup_auth(context)	Deprecated.
shutdown()	
shutdown_on_interrupt()	
timeout()	A function to call every self.timeout_interval seconds in the same thread as the handler.

handle_abort()

handle_engage()

handle_error_in_globals()

handle_get_globals(*raw=False*)

handle_get_labscript_file()

handle_get_run_shots()

handle_get_shot_output_folder()

handle_get_shuffle()

handle_get_view_shots()

handle_is_output_folder_default()

handle_n_shots()

handle_reset_shot_output_folder()

handle_set_globals(*globals*, *raw=False*)

handle_set_labscript_file(*value*)

handle_set_run_shots(*value*)

handle_set_shot_output_folder(*value*)

handle_set_shuffle(*value*)

handle_set_view_shots(*value*)

handler(*request_data*)

To be overridden by subclasses. This is an example implementation

3.6.16 runmanager.__main__.RunManager

class runmanager.__main__.**RunManager**

Bases: object

__init__()

Methods

__init__()	
add_item_to_axes_model(expansion_name, shuffle)	
check_child_exited(timeout_time[, kill])	
check_output_folder_update()	Do a single check of whether the output folder needs updating.

continues on next page

Table 3.11 – continued from previous page

close_current_tab()	
close_globals_file(globals_file[, confirm])	
close_group(globals_file, group_name)	
compile_loop()	
connect_signals()	
copy_group(source_globals_file, ...[, ...])	This function copys a group of globals with the name source_group_name from the file source_globals_file to a new file dest_globals_file.
delete_group(globals_file, group_name[, confirm])	
do_model_sort()	
get_active_groups([interactive])	Returns active groups in the format {group_name: globals_file}.
get_default_output_folder()	Returns what the default output folder would be right now, based on the current date and selected labscript file.
get_group_item_by_name(globals_file, ...[, ...])	Returns an item from the row representing a globals group in the groups model.
get_save_data()	
globals_changed()	Called from either self, a GroupTab, or the RemoteServer to inform runmanager that something about globals has changed, and that they need parsing again.
guess_expansion_modes(active_groups, ...)	This function is designed to be called iteratively.
load_configuration(filename)	
make_group_row(group_name)	Returns a new row representing one group in the groups tab, ready to be inserted into the model.
make_h5_files(labscript_file, output_folder, ...)	
new_group(globals_file, group_name)	
on_abort_clicked()	
on_axes_check_selected_triggered(*args)	
on_axes_item_changed(item)	
on_axes_uncheck_selected_triggered(*args)	
on_axis_down_clicked(checked)	
on_axis_to_bottom_clicked(checked)	
on_axis_to_top_clicked(checked)	

continues on next page

Table 3.11 – continued from previous page

on_axis_up_clicked(checked)	
on_close_event()	
on_diff_globals_file_clicked()	
on_edit_labscript_file_clicked(checked)	
on_engage_clicked()	
on_groups_close_selected_files_triggered()	
on_groups_close_selected_groups_triggered	
on_groups_copy_selected_groups_triggered([	
on_groups_delete_selected_triggered()	
on_groups_model_active_changed(item)	Sets the sort data for the item in response to its check state changing.
on_groups_model_item_changed(item)	This function is for responding to data changes in the model.
on_groups_model_name_changed(item)	Handles group renaming and creation of new groups due to the user editing the <click to add group> item
on_groups_model_openclose_changed(item)	Sets item sort data and icon in response to the open/close state of a group changing.
on_groups_open_selected_triggered()	
on_groups_set_selection_active_triggered(.	
on_labscript_file_text_changed(text)	
on_load_configuration_triggered()	
on_master_shuffle_clicked(state)	
on_new_globals_file_clicked()	
on_open_globals_file_clicked()	
on_output_popout_button_clicked()	
on_reset_shot_output_folder_clicked(checke	
on_restart_subprocess_clicked()	
on_revert_configuration_triggered()	
on_save_configuration_as_triggered()	
on_save_configuration_triggered()	

continues on next page

Table 3.11 – continued from previous page

on_select_labscript_file_clicked(checked)	
on_select_shot_output_folder_clicked(check	
on_shot_output_folder_text_changed(text)	
on_tabCloseRequested(index)	
on_treeView_axes_context_menu_requested(pc	
on_treeView_groups_context_menu_requested	
on_treeView_groups_doubleLeftClicked(index	
on_treeView_groups_leftClicked(index)	Here we respond to user clicks on the treeview.
open_globals_file(globals_file)	
open_group(globals_file, group_name[, ...])	
parse_globals(active_groups[, ...])	
preparse_globals()	
preparse_globals_loop()	Runs in a thread, waiting on a threading.Event that tells us when some globals have changed, and calls parse_globals to evaluate them all before feeding the results back to the relevant tabs to be displayed.
rename_group(globals_file, ...)	
rollover_shot_output_folder()	Runs in a thread, checking every 30 seconds if the default output folder has changed, likely because the date has changed, but also possible because another instance of runmanager has incremented the sequence index.
save_configuration(save_file)	
send_to_BLACS(run_file, BLACS_hostname)	
send_to_runviewer(run_file)	
setup_axes_tab()	
setup_config()	
setup_groups_tab()	
switch_tabs(change)	
update_axes_indentation()	
update_axes_tab(expansions, dimensions)	

continues on next page

Table 3.11 – continued from previous page

update_global_shuffle_state(*args, **kwargs)	
update_tabs_parsing_indication(...)	
wait_until_preparse_complete()	Block until the preparse loop has finished pending work

Attributes

AXES_COL_LENGTH
AXES_COL_NAME
AXES_COL_SHUFFLE
AXES_ROLE_NAME
GROUPS_COL_ACTIVE
GROUPS_COL_DELETE
GROUPS_COL_NAME
GROUPS_COL_OPENCLOSE
GROUPS_DUMMY_ROW_TEXT
GROUPS_ROLE_GROUP_IS_OPEN
GROUPS_ROLE_IS_DUMMY_ROW
GROUPS_ROLE_PREVIOUS_NAME
GROUPS_ROLE_SORT_DATA

AXES_COL_LENGTH = 1

AXES_COL_NAME = 0

AXES_COL_SHUFFLE = 2

AXES_ROLE_NAME = 257

GROUPS_COL_ACTIVE = 1

GROUPS_COL_DELETE = 2

GROUPS_COL_NAME = 0

GROUPS_COL_OPENCLOSE = 3

GROUPS_DUMMY_ROW_TEXT = '<Click to add group>'

GROUPS_ROLE_GROUP_IS_OPEN = 260

GROUPS_ROLE_IS_DUMMY_ROW = 257

GROUPS_ROLE_PREVIOUS_NAME = 258

GROUPS_ROLE_SORT_DATA = 259

add_item_to_axes_model(*expansion_name*, *shuffle*, *dimensions=None*)

check_child_exited(*timeout_time*, *kill=False*)

check_output_folder_update()

Do a single check of whether the output folder needs updating. This is implemented as a separate function to the above loop so that the whole check happens at once in the Qt main thread and hence is atomic and can't be interfered with by other Qt calls in the program.

close_current_tab()

close_globals_file(*globals_file*, *confirm=True*)

close_group(*globals_file*, *group_name*)

compile_loop()

connect_signals()

copy_group(*source_globals_file*, *source_group_name*, *dest_globals_file=None*, *delete_source_group=False*)

This function copys a group of globals with the name source_group_name from the file source_globals_file to a new file dest_globals_file. If delete_source_group is True the source group is deleted after copying

delete_group(*globals_file*, *group_name*, *confirm=True*)

do_model_sort()

get_active_groups(*interactive=True*)

Returns active groups in the format {group_name: globals_file}. Displays an error dialog and returns None if multiple groups of the same name are selected, this is invalid - selected groups must be uniquely named. If interactive=False, raises the exception instead.

get_default_output_folder()

Returns what the default output folder would be right now, based on the current date and selected labscript file. Returns empty string if no labscript file is selected. Does not create the default output folder, does not check if it exists.

get_group_item_by_name(*globals_file*, *group_name*, *column*, *previous_name=None*)

Returns an item from the row representing a globals group in the groups model. Which item is returned is set by the column argument.

get_save_data()

globals_changed()

Called from either self, a GroupTab, or the RemoteServer to inform runmanager that something about globals has changed, and that they need parsing again.

guess_expansion_modes(*active_groups*, *evaled_globals*, *global_hierarchy*, *expansions*)

This function is designed to be called iteratively. It changes the expansion type of globals that reference other globals - such that globals referencing an iterable global will be zipped with it, rather than outer producted. Each time this method is called, self.parse_globals should also be called, so that the globals are evaluated with their new expansion modes, if they changed. This should be performed repeatedly until there are no more changes. Note that this method does not return what expansion types it thinks globals should have - it *actually writes them to the globals HDF5 file*. So it is up to later code to ensure it re-reads the expansion mode from the HDF5 file before proceeding. At present this method is only called from self.preparse_globals(), so see there to see how it fits in with everything else. This method uses four instance attributes to store state: self.previous_evaled_globals, self.previous_global_hierarchy, self.previous_expansion_types and self.previous_expansions. This is neccesary so that it can detect changes.

load_configuration(*filename*)

make_group_row(*group_name*)

Returns a new row representing one group in the groups tab, ready to be inserted into the model.

make_h5_files(*labscript_file*, *output_folder*, *sequence_globals*, *shots*, *shuffle*)

new_group(*globals_file*, *group_name*)

on_abort_clicked()

on_axes_check_selected_triggered(**args*)

on_axes_item_changed(*item*)

on_axes_uncheck_selected_triggered(**args*)

on_axis_down_clicked(*checked*)

on_axis_to_bottom_clicked(*checked*)

on_axis_to_top_clicked(*checked*)

on_axis_up_clicked(*checked*)

on_close_event()

on_diff_globals_file_clicked()

on_edit_labscript_file_clicked(*checked*)

on_engage_clicked()

on_groups_close_selected_files_triggered()

on_groups_close_selected_groups_triggered()

on_groups_copy_selected_groups_triggered(*dest_globals_file=None*, *delete_source_group=False*)

on_groups_delete_selected_triggered()

on_groups_model_active_changed(*item*)

Sets the sort data for the item in response to its check state changing. Also, if this is the first time this function has been called on the stack, that is, the change was initiated externally instead of via recursion from this function itself, then set the check state of other items for consistency. This entails checking/unchecking all group rows in response to the file row's check state changing, or changing the file row's check state to

reflect the check state of the child group rows. That's why we need to keep track of the recursion depth - so that those changes we make don't in turn cause further changes. But we don't disconnect the on_changed signal altogether, because we still want to do the update of the sort data, and anything else that might be added in future.

on_groups_model_item_changed(*item*)

This function is for responding to data changes in the model. The methods for responding to changes different columns do different things. Mostly they make other data changes for model consistency, but also group creation and renaming is handled in response to changes to the 'name' column. When we change things elsewhere, we prefer to only change one thing, and the rest of the changes are triggered here. So here we do the following:

Be careful not to recurse unsafely into this method - changing something that itself triggers further changes is fine so long as they peter out and don't get stuck in a loop. If recursion needs to be stopped, one can disconnect the signal temporarily with the context manager self.groups_model_item_changed_disconnected. But use this sparingly, otherwise there's the risk that some required data updates will be forgotten about and won't happen.

on_groups_model_name_changed(*item*)

Handles group renaming and creation of new groups due to the user editing the <click to add group> item

on_groups_model_openclose_changed(*item*)

Sets item sort data and icon in response to the open/close state of a group changing.

on_groups_open_selected_triggered()

on_groups_set_selection_active_triggered(*checked_state*)

on_labscript_file_text_changed(*text*)

on_load_configuration_triggered()

on_master_shuffle_clicked(*state*)

on_new_globals_file_clicked()

on_open_globals_file_clicked()

on_output_popout_button_clicked()

on_reset_shot_output_folder_clicked(*checked*)

on_restart_subprocess_clicked()

on_revert_configuration_triggered()

on_save_configuration_as_triggered()

on_save_configuration_triggered()

on_select_labscript_file_clicked(*checked*)

on_select_shot_output_folder_clicked(*checked*)

on_shot_output_folder_text_changed(*text*)

on_tabCloseRequested(*index*)

on_treeView_axes_context_menu_requested(*point*)

on_treeView_groups_context_menu_requested(*point*)

on_treeView_groups_doubleLeftClicked(*index*)

on_treeView_groups_leftClicked(*index*)

Here we respond to user clicks on the treeview. We do the following:

- If the user clicks on the <click to add group> dummy row, we go into edit mode on it so they can enter the name of the new group they want.
- If the user clicks on the icon to open or close a globals file or a group, we call the appropriate open and close methods and update the open/close data role on the model.
- If the user clicks delete on a globals group, we call a delete method, which deletes it after confirmation, and closes it if it was open.

open_globals_file(*globals_file*)

open_group(*globals_file*, *group_name*, *trigger_preparse=True*)

parse_globals(*active_groups*, *raise_exceptions=True*, *expand_globals=True*, *expansion_order=None*, *return_dimensions=False*)

preparse_globals()

preparse_globals_loop()

Runs in a thread, waiting on a threading.Event that tells us when some globals have changed, and calls parse_globals to evaluate them all before feeding the results back to the relevant tabs to be displayed.

rename_group(*globals_file*, *previous_group_name*, *new_group_name*)

rollover_shot_output_folder()

Runs in a thread, checking every 30 seconds if the default output folder has changed, likely because the date has changed, but also possible because another instance of runmanager has incremented the sequence index. If the defaulr output folder has changed, and if runmanager is configured to use the default output folder, sets the folder in which compiled shots will be put. Does not create the folder if it does not already exist, this will be done at compile-time.

save_configuration(*save_file*)

send_to_BLACS(*run_file*, *BLACS_hostname*)

send_to_runviewer(*run_file*)

setup_axes_tab()

setup_config()

setup_groups_tab()

switch_tabs(*change*)

update_axes_indentation()

update_axes_tab(*expansions*, *dimensions*)

update_global_shuffle_state(**args*, ***kwargs*)

update_tabs_parsing_indication(*active_groups*, *sequence_globals*, *evaled_globals*, *n_shots*)

wait_until_preparse_complete()

Block until the preparse loop has finished pending work

3.6.17 runmanager.__main__.RunmanagerMainWindow

class runmanager.__main__.**RunmanagerMainWindow**(**args*, ***kwargs*)

Bases: QMainWindow

__init__(**args*, ***kwargs*)

Methods

__init__(*args, **kwargs)
acceptDrops(self)
accessibleDescription(self)
accessibleName(self)
actionEvent(self, a0)
actions(self)
activateWindow(self)
addAction(self, action)
addActions(self, actions)
addDockWidget()
addToolBar(-> None)
addToolBarBreak(self[, area])
adjustSize(self)
autoFillBackground(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)
centralWidget(self)
changeEvent(self, a0)
childAt(-> Optional[QWidget])
childEvent(self, a0)

continues on next page

Table 3.12 – continued from previous page

children(self)
childrenRect(self)
childrenRegion(self)
clearFocus(self)
clearMask(self)
close(self)
closeEvent(self, a0)
colorCount(self)
connectNotify(self, signal)
contentsMargins(self)
contentsRect(self)
contextMenuEvent(self, event)
contextMenuPolicy(self)
corner(self, corner)
create(self[, window, initializeWindow, ...])
createPopupMenu(self)
createWindowContainer(window[, parent, flags])
cursor(self)
customEvent(self, a0)
deleteLater(self)
depth(self)
destroy(self[, destroyWindow, destroySubWindows])
devType(self)
devicePixelRatio(self)
devicePixelRatioF(self)
devicePixelRatioFScale()

continues on next page

Table 3.12 – continued from previous page

disconnect(-> bool)
disconnectNotify(self, signal)
dockOptions(self)
dockWidgetArea(self, dockwidget)
documentMode(self)
dragEnterEvent(self, a0)
dragLeaveEvent(self, a0)
dragMoveEvent(self, a0)
dropEvent(self, a0)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
effectiveWinId(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, event)
eventFilter(self, a0, a1)
find(a0)
findChild(-> QObjectT)
findChildren(...)
focusInEvent(self, a0)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, a0)
focusPolicy(self)
focusPreviousChild(self)

continues on next page

Table 3.12 – continued from previous page

focusProxy(self)
focusWidget(self)
font(self)
fontInfo(self)
fontMetrics(self)
foregroundRole(self)
frameGeometry(self)
frameSize(self)
geometry(self)
getContentsMargins(self)
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])
graphicsEffect(self)
graphicsProxyWidget(self)
hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)
hasTabletTracking(self)
height(self)
heightForWidth(self, a0)
heightMM(self)
hide(self)
hideEvent(self, a0)

continues on next page

Table 3.12 – continued from previous page

iconSize(self)
inherits(self, classname)
initPainter(self, painter)
inputMethodEvent(self, a0)
inputMethodHints(self)
inputMethodQuery(self, a0)
insertAction(self, before, action)
insertActions(self, before, actions)
insertToolBar(self, before, toolbar)
insertToolBarBreak(self, before)
installEventFilter(self, a0)
isActiveWindow(self)
isAncestorOf(self, child)
isAnimated(self)
isDockNestingEnabled(self)
isEnabled(self)
isEnabledTo(self, a0)
isFullScreen(self)
isHidden(self)
isLeftToRight(self)
isMaximized(self)
isMinimized(self)
isModal(self)
isRightToLeft(self)
isSeparator(self, pos)
isSignalConnected(self, signal)

continues on next page

Table 3.12 – continued from previous page

isVisible(self)
isVisibleTo(self, a0)
isWidgetType(self)
isWindow(self)
isWindowModified(self)
isWindowType(self)
keyPressEvent(self, a0)
keyReleaseEvent(self, a0)
keyboardGrabber()
killTimer(self, id)
layout(self)
layoutDirection(self)
leaveEvent(self, a0)
locale(self)
logicalDpiX(self)
logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)
mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)

continues on next page

Table 3.12 – continued from previous page

maximumWidth(self)
menuBar(self)
menuWidget(self)
metaObject(self)
metric(self, a0)
minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumWidth(self)
mouseDoubleClickEvent(self, a0)
mouseGrabber()
mouseMoveEvent(self, a0)
mousePressEvent(self, a0)
mouseReleaseEvent(self, a0)
move()
moveEvent(self, a0)
moveToThread(self, thread)
nativeEvent(self, eventType, message)
nativeParentWidget(self)
nextInFocusChain(self)
normalGeometry(self)
objectName(self)
overrideWindowFlags(self, type)
overrideWindowState(self, state)
paintEngine(self)
paintEvent(self, a0)

continues on next page

Table 3.12 – continued from previous page

paintingActive(self)	
palette(self)	
parent(self)	
parentWidget(self)	
physicalDpiX(self)	
physicalDpiY(self)	
pos(self)	
previousInFocusChain(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
raise_(self)	
receivers(self, signal)	
rect(self)	
releaseKeyboard(self)	
releaseMouse(self)	
releaseShortcut(self, id)	
removeAction(self, action)	
removeDockWidget(self, dockwidget)	
removeEventFilter(self, a0)	
removeToolBar(self, toolbar)	
removeToolBarBreak(self, before)	
render(, sourceRegion, flags, ...)	
repaint(-> None -> None)	
resize()	
resizeDocks(self, docks, sizes, orientation)	
resizeEvent(self, a0)	

continues on next page

Table 3.12 – continued from previous page

restoreDockWidget(self, dockwidget)
restoreGeometry(self, geometry)
restoreState(self, state[, version])
saveGeometry(self)
saveState(self[, version])
screen(self)
scroll()
sender(self)
senderSignalIndex(self)
setAcceptDrops(self, on)
setAccessibleDescription(self, description)
setAccessibleName(self, name)
setAnimated(self, enabled)
setAttribute(self, attribute[, on])
setAutoFillBackground(self, enabled)
setBackgroundRole(self, a0)
setBaseSize()
setCentralWidget(self, widget)
setContentsMargins()
setContextMenuPolicy(self, policy)
setCorner(self, corner, area)
setCursor(self, a0)
setDisabled(self, a0)
setDockNestingEnabled(self, enabled)
setDockOptions(self, options)
setDocumentMode(self, enabled)

continues on next page

Table 3.12 – continued from previous page

setEnabled(self, a0)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()
setFocusPolicy(self, policy)
setFocusProxy(self, a0)
setFont(self, a0)
setForegroundRole(self, a0)
setGeometry()
setGraphicsEffect(self, effect)
setHidden(self, hidden)
setIconSize(self, iconSize)
setInputMethodHints(self, hints)
setLayout(self, a0)
setLayoutDirection(self, direction)
setLocale(self, locale)
setMask()
setMaximumHeight(self, maxh)
setMaximumSize()
setMaximumWidth(self, maxw)
setMenuBar(self, menubar)
setMenuWidget(self, menubar)
setMinimumHeight(self, minh)
setMinimumSize()
setMinimumWidth(self, minw)

continues on next page

Table 3.12 – continued from previous page

setMouseTracking(self, enable)
setObjectName(self, name)
setPalette(self, a0)
setParent()
setProperty(self, name, value)
setShortcutAutoRepeat(self, id[, enabled])
setShortcutEnabled(self, id[, enabled])
setSizeIncrement()
setSizePolicy()
setStatusBar(self, statusbar)
setStatusTip(self, a0)
setStyle(self, a0)
setStyleSheet(self, styleSheet)
setTabOrder(a0, a1)
setTabPosition(self, areas, tabPosition)
setTabShape(self, tabShape)
setTabletTracking(self, enable)
setToolButtonStyle(self, toolButtonStyle)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setUnifiedTitleAndToolBarOnMac(self, set)
setUpdatesEnabled(self, enable)
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])

continues on next page

Table 3.12 – continued from previous page

setWindowFlags(self, type)
setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)
setWindowOpacity(self, level)
setWindowRole(self, a0)
setWindowState(self, state)
setWindowTitle(self, a0)
sharedPainter(self)
show(self)
showEvent(self, a0)
showFullScreen(self)
showMaximized(self)
showMinimized(self)
showNormal(self)
signalsBlocked(self)
size(self)
sizeHint(self)
sizeIncrement(self)
sizePolicy(self)
splitDockWidget(self, after, dockwidget, ...)
stackUnder(self, a0)
startTimer(self, interval[, timerType])
statusBar(self)
statusTip(self)

continues on next page

Table 3.12 – continued from previous page

style(self)
styleSheet(self)
tabPosition(self, area)
tabShape(self)
tabifiedDockWidgets(self, dockwidget)
tabifyDockWidget(self, first, second)
tabletEvent(self, a0)
takeCentralWidget(self)
testAttribute(self, attribute)
thread(self)
timerEvent(self, a0)
toolBarArea(self, toolbar)
toolBarBreak(self, toolbar)
toolButtonStyle(self)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
underMouse(self)
ungrabGesture(self, type)
unifiedTitleAndToolBarOnMac(self)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)
update(-> None -> None)
updateGeometry(self)
updateMicroFocus(self)

continues on next page

Table 3.12 – continued from previous page

updatesEnabled(self)
visibleRegion(self)
whatsThis(self)
wheelEvent(self, a0)
width(self)
widthMM(self)
winId(self)
window(self)
windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)
windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)
windowTitle(self)
windowType(self)
x(self)
y(self)

Attributes

AllowNestedDocks	
AllowTabbedDocks	
AnimatedDocks	
DrawChildren	
DrawWindowBackground	
ForceTabbedDocks	
GroupedDragging	
IgnoreMask	
PdmDepth	
PdmDevicePixelRatio	
PdmDevicePixelRatioScaled	
PdmDpiX	
PdmDpiY	
PdmHeight	
PdmHeightMM	
PdmNumColors	
PdmPhysicalDpiX	
PdmPhysicalDpiY	
PdmWidth	
PdmWidthMM	
VerticalTabs	
customContextMenuRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
firstPaint	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
iconSizeChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

continues on next page

Table 3.13 – continued from previous page

objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	
tabifiedDockWidgetActivated	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
toolButtonStyleChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowTitleChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

closeEvent(*self*, *a0: QCloseEvent | None*)

firstPaint

int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

types is normally a sequence of individual types. Each type is either a type object or a string that is the name of a C++ type. Alternatively each type could itself be a sequence of types each describing a different overloaded signal. name is the optional C++ name of the signal. If it is not specified then the name of the class attribute that is bound to the signal is used. revision is the optional revision of the signal that is exported to QML. If it is not specified then 0 is used. arguments is the optional sequence of the names of the signal's arguments.

Type
pyqtSignal(*types, name

Type
str = ..., revision

paintEvent(*self*, *a0: QPaintEvent | None*)

3.6.18 runmanager.__main__.TabToolButton

class runmanager.__main__.**TabToolButton**(**args*, ***kwargs*)

Bases: QToolButton

__init__(**args*, ***kwargs*)

Methods

__init__(*args, **kwargs)
acceptDrops(self)
accessibleDescription(self)

continues on next page

Table 3.14 – continued from previous page

accessibleName(self)
actionEvent(self, a0)
actions(self)
activateWindow(self)
addAction(self, action)
addActions(self, actions)
adjustSize(self)
animateClick(self[, msecs])
arrowType(self)
autoExclusive(self)
autoFillBackground(self)
autoRaise(self)
autoRepeat(self)
autoRepeatDelay(self)
autoRepeatInterval(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)
changeEvent(self, a0)
checkStateSet(self)
childAt(-> Optional[QWidget])
childEvent(self, a0)
children(self)
childrenRect(self)
childrenRegion(self)
clearFocus(self)

continues on next page

Table 3.14 – continued from previous page

clearMask(self)
click(self)
close(self)
closeEvent(self, a0)
colorCount(self)
connectNotify(self, signal)
contentsMargins(self)
contentsRect(self)
contextMenuEvent(self, a0)
contextMenuPolicy(self)
create(self[, window, initializeWindow, ...])
createWindowContainer(window[, parent, flags])
cursor(self)
customEvent(self, a0)
defaultAction(self)
deleteLater(self)
depth(self)
destroy(self[, destroyWindow, destroySubWindows])
devType(self)
devicePixelRatio(self)
devicePixelRatioF(self)
devicePixelRatioFScale()
disconnect(-> bool)
disconnectNotify(self, signal)
dragEnterEvent(self, a0)
dragLeaveEvent(self, a0)

continues on next page

Table 3.14 – continued from previous page

dragMoveEvent(self, a0)
dropEvent(self, a0)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
effectiveWinId(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, e)
eventFilter(self, a0, a1)
find(a0)
findChild(-> QObjectT)
findChildren(...)
focusInEvent(self, e)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, e)
focusPolicy(self)
focusPreviousChild(self)
focusProxy(self)
focusWidget(self)
font(self)
fontInfo(self)
fontMetrics(self)
foregroundRole(self)
frameGeometry(self)

continues on next page

Table 3.14 – continued from previous page

frameSize(self)
geometry(self)
getContentsMargins(self)
get_correct_position()
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])
graphicsEffect(self)
graphicsProxyWidget(self)
group(self)
hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)
hasTabletTracking(self)
height(self)
heightForWidth(self, a0)
heightMM(self)
hide(self)
hideEvent(self, a0)
hitButton(self, pos)
icon(self)
iconSize(self)
inherits(self, classname)
initPainter(self, painter)

continues on next page

Table 3.14 – continued from previous page

initStyleOption(self, option)
inputMethodEvent(self, a0)
inputMethodHints(self)
inputMethodQuery(self, a0)
insertAction(self, before, action)
insertActions(self, before, actions)
installEventFilter(self, a0)
isActiveWindow(self)
isAncestorOf(self, child)
isCheckable(self)
isChecked(self)
isDown(self)
isEnabled(self)
isEnabledTo(self, a0)
isFullScreen(self)
isHidden(self)
isLeftToRight(self)
isMaximized(self)
isMinimized(self)
isModal(self)
isRightToLeft(self)
isSignalConnected(self, signal)
isVisible(self)
isVisibleTo(self, a0)
isWidgetType(self)
isWindow(self)

continues on next page

Table 3.14 – continued from previous page

isWindowModified(self)
isWindowType(self)
keyPressEvent(self, e)
keyReleaseEvent(self, e)
keyboardGrabber()
killTimer(self, id)
layout(self)
layoutDirection(self)
leaveEvent(self, a0)
locale(self)
logicalDpiX(self)
logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)
mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)
maximumWidth(self)
menu(self)
metaObject(self)
metric(self, a0)

continues on next page

Table 3.14 – continued from previous page

minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumWidth(self)
mouseDoubleClickEvent(self, a0)
mouseGrabber()
mouseMoveEvent(self, e)
mousePressEvent(self, a0)
mouseReleaseEvent(self, a0)
move()
moveEvent(self, a0)
moveToThread(self, thread)
nativeEvent(self, eventType, message)
nativeParentWidget(self)
nextCheckState(self)
nextInFocusChain(self)
normalGeometry(self)
objectName(self)
overrideWindowFlags(self, type)
overrideWindowState(self, state)
paintEngine(self)
paintEvent(self, a0)
paintingActive(self)
palette(self)
parent(self)
parentWidget(self)

continues on next page

Table 3.14 – continued from previous page

physicalDpiX(self)	
physicalDpiY(self)	
popupMode(self)	
pos(self)	
previousInFocusChain(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
raise_(self)	
receivers(self, signal)	
rect(self)	
releaseKeyboard(self)	
releaseMouse(self)	
releaseShortcut(self, id)	
removeAction(self, action)	
removeEventFilter(self, a0)	
render(, sourceRegion, flags, ...)	
repaint(-> None -> None)	
resize()	
resizeEvent(self, a0)	
restoreGeometry(self, geometry)	
saveGeometry(self)	
screen(self)	
scroll()	
sender(self)	
senderSignalIndex(self)	
setAcceptDrops(self, on)	

continues on next page

Table 3.14 – continued from previous page

setAccessibleDescription(self, description)
setAccessibleName(self, name)
setArrowType(self, type)
setAttribute(self, attribute[, on])
setAutoExclusive(self, a0)
setAutoFillBackground(self, enabled)
setAutoRaise(self, enable)
setAutoRepeat(self, a0)
setAutoRepeatDelay(self, a0)
setAutoRepeatInterval(self, a0)
setBackgroundRole(self, a0)
setBaseSize()
setCheckable(self, a0)
setChecked(self, a0)
setContentsMargins()
setContextMenuPolicy(self, policy)
setCursor(self, a0)
setDefaultAction(self, a0)
setDisabled(self, a0)
setDown(self, a0)
setEnabled(self, a0)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()
setFocusPolicy(self, policy)

continues on next page

Table 3.14 – continued from previous page

setFocusProxy(self, a0)
setFont(self, a0)
setForegroundRole(self, a0)
setGeometry()
setGraphicsEffect(self, effect)
setHidden(self, hidden)
setIcon(self, icon)
setIconSize(self, size)
setInputMethodHints(self, hints)
setLayout(self, a0)
setLayoutDirection(self, direction)
setLocale(self, locale)
setMask()
setMaximumHeight(self, maxh)
setMaximumSize()
setMaximumWidth(self, maxw)
setMenu(self, menu)
setMinimumHeight(self, minh)
setMinimumSize()
setMinimumWidth(self, minw)
setMouseTracking(self, enable)
setObjectName(self, name)
setPalette(self, a0)
setParent()
setPopupMode(self, mode)
setProperty(self, name, value)

continues on next page

Table 3.14 – continued from previous page

setShortcut(self, key)
setShortcutAutoRepeat(self, id[, enabled])
setShortcutEnabled(self, id[, enabled])
setSizeIncrement()
setSizePolicy()
setStatusTip(self, a0)
setStyle(self, a0)
setStyleSheet(self, styleSheet)
setTabOrder(a0, a1)
setTabletTracking(self, enable)
setText(self, text)
setToolButtonStyle(self, style)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setUpdatesEnabled(self, enable)
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])
setWindowFlags(self, type)
setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)
setWindowOpacity(self, level)
setWindowRole(self, a0)

continues on next page

Table 3.14 – continued from previous page

setWindowState(self, state)
setWindowTitle(self, a0)
sharedPainter(self)
shortcut(self)
show(self)
showEvent(self, a0)
showFullScreen(self)
showMaximized(self)
showMenu(self)
showMinimized(self)
showNormal(self)
signalsBlocked(self)
size(self)
sizeHint(self)
sizeIncrement(self)
sizePolicy(self)
stackUnder(self, a0)
startTimer(self, interval[, timerType])
statusTip(self)
style(self)
styleSheet(self)
tabletEvent(self, a0)
testAttribute(self, attribute)
text(self)
thread(self)
timerEvent(self, a0)

continues on next page

Table 3.14 – continued from previous page

toggle(self)
toolButtonStyle(self)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
underMouse(self)
ungrabGesture(self, type)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)
update(-> None -> None)
updateGeometry(self)
updateMicroFocus(self)
updatesEnabled(self)
visibleRegion(self)
whatsThis(self)
wheelEvent(self, a0)
width(self)
widthMM(self)
winId(self)
window(self)
windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)

continues on next page

Table 3.14 – continued from previous page

windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)
windowTitle(self)
windowType(self)
x(self)
y(self)

Attributes

DelayedPopup	
DrawChildren	
DrawWindowBackground	
IgnoreMask	
InstantPopup	
MenuButtonPopup	
PdmDepth	
PdmDevicePixelRatio	
PdmDevicePixelRatioScaled	
PdmDpiX	
PdmDpiY	
PdmHeight	
PdmHeightMM	
PdmNumColors	
PdmPhysicalDpiX	
PdmPhysicalDpiY	
PdmWidth	
PdmWidthMM	
clicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
customContextMenuRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
pressed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
released	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	
toggled	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
triggered	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments: Sequence = ...) ->

get_correct_position()

moveEvent(*self*, *a0: QMoveEvent | None*)

paintEvent(*self*, *a0: QPaintEvent | None*)

3.6.19 runmanager.__main__.TableView

class runmanager.__main__.**TableView**(*parent=None*)

Bases: *ItemView*, QTableView

TableView version of our customised ItemView

__init__(*parent=None*)

Methods

__init__([parent])
acceptDrops(self)
accessibleDescription(self)
accessibleName(self)
actionEvent(self, a0)
actions(self)
activateWindow(self)
addAction(self, action)
addActions(self, actions)
addScrollBarWidget(self, widget, alignment)
adjustSize(self)
alternatingRowColors(self)
autoFillBackground(self)
autoScrollMargin(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)

continues on next page

Table 3.15 – continued from previous page

changeEvent(self, a0)
childAt(-> Optional[QWidget])
childEvent(self, a0)
children(self)
childrenRect(self)
childrenRegion(self)
clearFocus(self)
clearMask(self)
clearSelection(self)
clearSpans(self)
close(self)
closeEditor(self, editor, hint)
closeEvent(self, a0)
closePersistentEditor(self, index)
colorCount(self)
columnAt(self, x)
columnCountChanged(self, oldCount, newCount)
columnMoved(self, column, oldIndex, newIndex)
columnResized(self, column, oldWidth, newWidth)
columnSpan(self, row, column)
columnViewportPosition(self, column)
columnWidth(self, column)
commitData(self, editor)
connectNotify(self, signal)
contentsMargins(self)
contentsRect(self)

continues on next page

Table 3.15 – continued from previous page

contextMenuEvent(self, a0)
contextMenuPolicy(self)
cornerWidget(self)
create(self[, window, initializeWindow, ...])
createWindowContainer(window[, parent, flags])
currentChanged(self, current, previous)
currentIndex(self)
cursor(self)
customEvent(self, a0)
dataChanged(self, topLeft, bottomRight[, roles])
defaultDropAction(self)
deleteLater(self)
depth(self)
destroy(self[, destroyWindow, destroySubWindows])
devType(self)
devicePixelRatio(self)
devicePixelRatioF(self)
devicePixelRatioFScale()
dirtyRegionOffset(self)
disconnect(-> bool)
disconnectNotify(self, signal)
dragDropMode(self)
dragDropOverwriteMode(self)
dragEnabled(self)
dragEnterEvent(self, e)
dragLeaveEvent(self, e)

continues on next page

Table 3.15 – continued from previous page

dragMoveEvent(self, e)
drawFrame(self, a0)
dropEvent(self, e)
dropIndicatorPosition(self)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
edit()
editTriggers(self)
editorDestroyed(self, editor)
effectiveWinId(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, event)
eventFilter(self, object, event)
executeDelayedItemsLayout(self)
find(a0)
findChild(-> QObjectT)
findChildren(...)
focusInEvent(self, e)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, e)
focusPolicy(self)
focusPreviousChild(self)
focusProxy(self)

continues on next page

Table 3.15 – continued from previous page

focusWidget(self)
font(self)
fontInfo(self)
fontMetrics(self)
foregroundRole(self)
frameGeometry(self)
frameRect(self)
frameShadow(self)
frameShape(self)
frameSize(self)
frameStyle(self)
frameWidth(self)
geometry(self)
getContentsMargins(self)
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])
graphicsEffect(self)
graphicsProxyWidget(self)
gridStyle(self)
hasAutoScroll(self)
hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)

continues on next page

Table 3.15 – continued from previous page

hasTabletTracking(self)
height(self)
heightForWidth(self, a0)
heightMM(self)
hide(self)
hideColumn(self, column)
hideEvent(self, a0)
hideRow(self, row)
horizontalHeader(self)
horizontalOffset(self)
horizontalScrollBar(self)
horizontalScrollBarPolicy(self)
horizontalScrollMode(self)
horizontalScrollbarAction(self, action)
horizontalScrollbarValueChanged(self, value)
iconSize(self)
indexAt(self, p)
indexWidget(self, index)
inherits(self, classname)
initPainter(self, painter)
initStyleOption(self, option)
inputMethodEvent(self, event)
inputMethodHints(self)
inputMethodQuery(self, query)
insertAction(self, before, action)
insertActions(self, before, actions)

continues on next page

Table 3.15 – continued from previous page

installEventFilter(self, a0)	
isActiveWindow(self)	
isAncestorOf(self, child)	
isColumnHidden(self, column)	
isCornerButtonEnabled(self)	
isEnabled(self)	
isEnabledTo(self, a0)	
isFullScreen(self)	
isHidden(self)	
isIndexHidden(self, index)	
isLeftToRight(self)	
isMaximized(self)	
isMinimized(self)	
isModal(self)	
isPersistentEditorOpen(self, index)	
isRightToLeft(self)	
isRowHidden(self, row)	
isSignalConnected(self, signal)	
isSortingEnabled(self)	
isVisible(self)	
isVisibleTo(self, a0)	
isWidgetType(self)	
isWindow(self)	
isWindowModified(self)	
isWindowType(self)	
itemDelegate(-> Optional[QAbstractItemDelegate])	

continues on next page

Table 3.15 – continued from previous page

itemDelegateForColumn(self, column)
itemDelegateForRow(self, row)
keyPressEvent(self, e)
keyReleaseEvent(self, a0)
keyboardGrabber()
keyboardSearch(self, search)
killTimer(self, id)
layout(self)
layoutDirection(self)
leaveEvent(self, a0)
lineWidth(self)
locale(self)
logicalDpiX(self)
logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)
mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)
maximumViewportSize(self)
maximumWidth(self)

continues on next page

Table 3.15 – continued from previous page

metaObject(self)
metric(self, a0)
midLineWidth(self)
minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumWidth(self)
model(self)
mouseDoubleClickEvent(self, e)
mouseGrabber()
mouseMoveEvent(self, e)
mousePressEvent(self, e)
mouseReleaseEvent(self, e)
move()
moveCursor(self, cursorAction, modifiers)
moveEvent(self, a0)
moveToThread(self, thread)
nativeEvent(self, eventType, message)
nativeParentWidget(self)
nextInFocusChain(self)
normalGeometry(self)
objectName(self)
on_column_resized(col)
openPersistentEditor(self, index)
overrideWindowFlags(self, type)
overrideWindowState(self, state)

continues on next page

Table 3.15 – continued from previous page

paintEngine(self)	
paintEvent(self, e)	
paintingActive(self)	
palette(self)	
parent(self)	
parentWidget(self)	
physicalDpiX(self)	
physicalDpiY(self)	
pos(self)	
previousInFocusChain(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
raise_(self)	
receivers(self, signal)	
rect(self)	
releaseKeyboard(self)	
releaseMouse(self)	
releaseShortcut(self, id)	
removeAction(self, action)	
removeEventFilter(self, a0)	
render(, sourceRegion, flags, ...)	
repaint(-> None -> None)	
reset(self)	
resetHorizontalScrollMode(self)	
resetVerticalScrollMode(self)	
resize()	

continues on next page

Table 3.15 – continued from previous page

resizeColumnToContents(self, column)
resizeColumnsToContents(self)
resizeEvent(self, e)
resizeRowToContents(self, row)
resizeRowsToContents(self)
restoreGeometry(self, geometry)
rootIndex(self)
rowAt(self, y)
rowCountChanged(self, oldCount, newCount)
rowHeight(self, row)
rowMoved(self, row, oldIndex, newIndex)
rowResized(self, row, oldHeight, newHeight)
rowSpan(self, row, column)
rowViewportPosition(self, row)
rowsAboutToBeRemoved(self, parent, start, end)
rowsInserted(self, parent, start, end)
saveGeometry(self)
scheduleDelayedItemsLayout(self)
screen(self)
scroll()
scrollBarWidgets(self, alignment)
scrollContentsBy(self, dx, dy)
scrollDirtyRegion(self, dx, dy)
scrollTo(self, index[, hint])
scrollToBottom(self)
scrollToTop(self)

continues on next page

Table 3.15 – continued from previous page

selectAll(self)
selectColumn(self, column)
selectRow(self, row)
selectedIndexes(self)
selectionBehavior(self)
selectionChanged(self, selected, deselected)
selectionCommand(self, index[, event])
selectionMode(self)
selectionModel(self)
sender(self)
senderSignalIndex(self)
setAcceptDrops(self, on)
setAccessibleDescription(self, description)
setAccessibleName(self, name)
setAlternatingRowColors(self, enable)
setAttribute(self, attribute[, on])
setAutoFillBackground(self, enabled)
setAutoScroll(self, enable)
setAutoScrollMargin(self, margin)
setBackgroundRole(self, a0)
setBaseSize()
setColumnHidden(self, column, hide)
setColumnWidth(self, column, width)
setContentsMargins()
setContextMenuPolicy(self, policy)
setCornerButtonEnabled(self, enable)

continues on next page

Table 3.15 – continued from previous page

setCornerWidget(self, widget)
setCurrentIndex(self, index)
setCursor(self, a0)
setDefaultDropAction(self, dropAction)
setDirtyRegion(self, region)
setDisabled(self, a0)
setDragDropMode(self, behavior)
setDragDropOverwriteMode(self, overwrite)
setDragEnabled(self, enable)
setDropIndicatorShown(self, enable)
setEditTriggers(self, triggers)
setEnabled(self, a0)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()
setFocusPolicy(self, policy)
setFocusProxy(self, a0)
setFont(self, a0)
setForegroundRole(self, a0)
setFrameRect(self, a0)
setFrameShadow(self, a0)
setFrameShape(self, a0)
setFrameStyle(self, a0)
setGeometry()
setGraphicsEffect(self, effect)

continues on next page

Table 3.15 – continued from previous page

setGridStyle(self, style)	
setHidden(self, hidden)	
setHorizontalHeader(self, header)	
setHorizontalScrollBar(self, scrollbar)	
setHorizontalScrollBarPolicy(self, a0)	
setHorizontalScrollMode(self, mode)	
setIconSize(self, size)	
setIndexWidget(self, index, widget)	
setInputMethodHints(self, hints)	
setItemDelegate(self, delegate)	
setItemDelegateForColumn(self, column, delegate)	
setItemDelegateForRow(self, row, delegate)	
setLayout(self, a0)	
setLayoutDirection(self, direction)	
setLineWidth(self, a0)	
setLocale(self, locale)	
setMask()	
setMaximumHeight(self, maxh)	
setMaximumSize()	
setMaximumWidth(self, maxw)	
setMidLineWidth(self, a0)	
setMinimumHeight(self, minh)	
setMinimumSize()	
setMinimumWidth(self, minw)	
setModel(self, model)	
setMouseTracking(self, enable)	

continues on next page

Table 3.15 – continued from previous page

setObjectName(self, name)
setPalette(self, a0)
setParent()
setProperty(self, name, value)
setRootIndex(self, index)
setRowHeight(self, row, height)
setRowHidden(self, row, hide)
setSelection(self, rect, command)
setSelectionBehavior(self, behavior)
setSelectionMode(self, mode)
setSelectionModel(self, selectionModel)
setShortcutAutoRepeat(self, id[, enabled])
setShortcutEnabled(self, id[, enabled])
setShowGrid(self, show)
setSizeAdjustPolicy(self, policy)
setSizeIncrement()
setSizePolicy()
setSortingEnabled(self, enable)
setSpan(self, row, column, rowSpan, columnSpan)
setState(self, state)
setStatusTip(self, a0)
setStyle(self, a0)
setStyleSheet(self, styleSheet)
setTabKeyNavigation(self, enable)
setTabOrder(a0, a1)
setTabletTracking(self, enable)

continues on next page

Table 3.15 – continued from previous page

setTextElideMode(self, mode)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setUpdatesEnabled(self, enable)
setVerticalHeader(self, header)
setVerticalScrollBar(self, scrollbar)
setVerticalScrollBarPolicy(self, a0)
setVerticalScrollMode(self, mode)
setViewport(self, widget)
setViewportMargins()
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])
setWindowFlags(self, type)
setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)
setWindowOpacity(self, level)
setWindowRole(self, a0)
setWindowState(self, state)
setWindowTitle(self, a0)
setWordWrap(self, on)
setupViewport(self, viewport)
sharedPainter(self)

continues on next page

Table 3.15 – continued from previous page

show(self)
showColumn(self, column)
showDropIndicator(self)
showEvent(self, a0)
showFullScreen(self)
showGrid(self)
showMaximized(self)
showMinimized(self)
showNormal(self)
showRow(self, row)
signalsBlocked(self)
size(self)
sizeAdjustPolicy(self)
sizeHint(self)
sizeHintForColumn(self, column)
sizeHintForIndex(self, index)
sizeHintForRow(self, row)
sizeIncrement(self)
sizePolicy(self)
sortByColumn(self, column, order)
stackUnder(self, a0)
startDrag(self, supportedActions)
startTimer(self, interval[, timerType])
state(self)
statusTip(self)
style(self)

continues on next page

Table 3.15 – continued from previous page

styleSheet(self)
tabKeyNavigation(self)
tabletEvent(self, a0)
testAttribute(self, attribute)
textElideMode(self)
thread(self)
timerEvent(self, event)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
underMouse(self)
ungrabGesture(self, type)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)
update()
updateEditorData(self)
updateEditorGeometries(self)
updateGeometries(self)
updateGeometry(self)
updateMicroFocus(self)
updatesEnabled(self)
verticalHeader(self)
verticalOffset(self)
verticalScrollBar(self)
verticalScrollBarPolicy(self)

continues on next page

Table 3.15 – continued from previous page

verticalScrollMode(self)
verticalScrollbarAction(self, action)
verticalScrollbarValueChanged(self, value)
viewOptions(self)
viewport(self)
viewportEvent(self, e)
viewportMargins(self)
viewportSizeHint(self)
visibleRegion(self)
visualRect(self, index)
visualRegionForSelection(self, selection)
whatsThis(self)
wheelEvent(self, a0)
width(self)
widthMM(self)
winId(self)
window(self)
windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)
windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)

continues on next page

Table 3.15 – continued from previous page

windowTitle(self)
windowType(self)
wordWrap(self)
x(self)
y(self)

Attributes

AboveItem
AdjustIgnored
AdjustToContents
AdjustToContentsOnFirstShow
AllEditTriggers
AnimatingState
AnyKeyPressed
BelowItem
Box
COLOR_HIGHLIGHT
CollapsingState
ContiguousSelection
CurrentChanged
DoubleClicked
DragDrop
DragOnly
DragSelectingState
DraggingState

continues on next page

Table 3.16 – continued from previous page

DrawChildren
DrawWindowBackground
DropOnly
EditKeyPressed
EditingState
EnsureVisible
ExpandingState
ExtendedSelection
HLine
IgnoreMask
InternalMove
MoveDown
MoveEnd
MoveHome
MoveLeft
MoveNext
MovePageDown
MovePageUp
MovePrevious
MoveRight
MoveUp
MultiSelection
NoDragDrop
NoEditTriggers
NoFrame
NoSelection

continues on next page

Table 3.16 – continued from previous page

NoState
OnItem
OnViewport
Panel
PdmDepth
PdmDevicePixelRatio
PdmDevicePixelRatioScaled
PdmDpiX
PdmDpiY
PdmHeight
PdmHeightMM
PdmNumColors
PdmPhysicalDpiX
PdmPhysicalDpiY
PdmWidth
PdmWidthMM
Plain
PositionAtBottom
PositionAtCenter
PositionAtTop
Raised
ScrollPerItem
ScrollPerPixel
SelectColumns
SelectItems
SelectRows

continues on next page

Table 3.16 – continued from previous page

SelectedClicked	
Shadow_Mask	
Shape_Mask	
SingleSelection	
StyledPanel	
Sunken	
VLine	
WinPanel	
activated	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
clicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
customContextMenuRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
doubleClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
doubleLeftClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
entered	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
iconSizeChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
leftClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
pressed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	
viewportEntered	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowTitleChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

on_column_resized(*col*)

3.6.20 runmanager.__main__.TreeView

class runmanager.__main__.**TreeView**(*parent=None*)

Bases: *ItemView*, QTreeView

Treeview version of our customised ItemView

__init__(*parent=None*)

Methods

__init__([parent])
acceptDrops(self)
accessibleDescription(self)
accessibleName(self)
actionEvent(self, a0)
actions(self)
activateWindow(self)
addAction(self, action)
addActions(self, actions)
addScrollBarWidget(self, widget, alignment)
adjustSize(self)
allColumnsShowFocus(self)
alternatingRowColors(self)
autoExpandDelay(self)
autoFillBackground(self)
autoScrollMargin(self)
backgroundRole(self)
baseSize(self)
blockSignals(self, b)
changeEvent(self, a0)

continues on next page

Table 3.17 – continued from previous page

childAt(-> Optional[QWidget])
childEvent(self, a0)
children(self)
childrenRect(self)
childrenRegion(self)
clearFocus(self)
clearMask(self)
clearSelection(self)
close(self)
closeEditor(self, editor, hint)
closeEvent(self, a0)
closePersistentEditor(self, index)
collapse(self, index)
collapseAll(self)
colorCount(self)
columnAt(self, x)
columnCountChanged(self, oldCount, newCount)
columnMoved(self)
columnResized(self, column, oldSize, newSize)
columnViewportPosition(self, column)
columnWidth(self, column)
commitData(self, editor)
connectNotify(self, signal)
contentsMargins(self)
contentsRect(self)
contextMenuEvent(self, a0)

continues on next page

Table 3.17 – continued from previous page

contextMenuPolicy(self)	
cornerWidget(self)	
create(self[, window, initializeWindow, ...])	
createWindowContainer(window[, parent, flags])	
currentChanged(self, current, previous)	
currentIndex(self)	
cursor(self)	
customEvent(self, a0)	
dataChanged(self, topLeft, bottomRight[, roles])	
defaultDropAction(self)	
deleteLater(self)	
depth(self)	
destroy(self[, destroyWindow, destroySubWindows])	
devType(self)	
devicePixelRatio(self)	
devicePixelRatioF(self)	
devicePixelRatioFScale()	
dirtyRegionOffset(self)	
disconnect(-> bool)	
disconnectNotify(self, signal)	
dragDropMode(self)	
dragDropOverwriteMode(self)	
dragEnabled(self)	
dragEnterEvent(self, e)	
dragLeaveEvent(self, e)	
dragMoveEvent(self, event)	

continues on next page

Table 3.17 – continued from previous page

drawBranches(self, painter, rect, index)
drawFrame(self, a0)
drawRow(self, painter, options, index)
drawTree(self, painter, region)
dropEvent(self, e)
dropIndicatorPosition(self)
dumpObjectInfo(self)
dumpObjectTree(self)
dynamicPropertyNames(self)
edit()
editTriggers(self)
editorDestroyed(self, editor)
effectiveWinId(self)
ensurePolished(self)
enterEvent(self, a0)
event(self, event)
eventFilter(self, object, event)
executeDelayedItemsLayout(self)
expand(self, index)
expandAll(self)
expandRecursively(self, index[, depth])
expandToDepth(self, depth)
expandsOnDoubleClick(self)
find(a0)
findChild(-> QObjectT)
findChildren(...)

continues on next page

Table 3.17 – continued from previous page

focusInEvent(self, e)
focusNextChild(self)
focusNextPrevChild(self, next)
focusOutEvent(self, e)
focusPolicy(self)
focusPreviousChild(self)
focusProxy(self)
focusWidget(self)
font(self)
fontInfo(self)
fontMetrics(self)
foregroundRole(self)
frameGeometry(self)
frameRect(self)
frameShadow(self)
frameShape(self)
frameSize(self)
frameStyle(self)
frameWidth(self)
geometry(self)
getContentsMargins(self)
grab(self[, rectangle])
grabGesture(self, type[, flags])
grabKeyboard(self)
grabMouse()
grabShortcut(self, key[, context])

continues on next page

Table 3.17 – continued from previous page

graphicsEffect(self)
graphicsProxyWidget(self)
hasAutoScroll(self)
hasFocus(self)
hasHeightForWidth(self)
hasMouseTracking(self)
hasTabletTracking(self)
header(self)
height(self)
heightForWidth(self, a0)
heightMM(self)
hide(self)
hideColumn(self, column)
hideEvent(self, a0)
horizontalOffset(self)
horizontalScrollBar(self)
horizontalScrollBarPolicy(self)
horizontalScrollMode(self)
horizontalScrollbarAction(self, action)
horizontalScrollbarValueChanged(self, value)
iconSize(self)
indentation(self)
indexAbove(self, index)
indexAt(self, p)
indexBelow(self, index)
indexRowSizeHint(self, index)

continues on next page

Table 3.17 – continued from previous page

indexWidget(self, index)
inherits(self, classname)
initPainter(self, painter)
initStyleOption(self, option)
inputMethodEvent(self, event)
inputMethodHints(self)
inputMethodQuery(self, query)
insertAction(self, before, action)
insertActions(self, before, actions)
installEventFilter(self, a0)
isActiveWindow(self)
isAncestorOf(self, child)
isAnimated(self)
isColumnHidden(self, column)
isEnabled(self)
isEnabledTo(self, a0)
isExpanded(self, index)
isFirstColumnSpanned(self, row, parent)
isFullScreen(self)
isHeaderHidden(self)
isHidden(self)
isIndexHidden(self, index)
isLeftToRight(self)
isMaximized(self)
isMinimized(self)
isModal(self)

continues on next page

Table 3.17 – continued from previous page

isPersistentEditorOpen(self, index)	
isRightToLeft(self)	
isRowHidden(self, row, parent)	
isSignalConnected(self, signal)	
isSortingEnabled(self)	
isVisible(self)	
isVisibleTo(self, a0)	
isWidgetType(self)	
isWindow(self)	
isWindowModified(self)	
isWindowType(self)	
itemDelegate(-> tional[QAbstractItemDelegate])	Op-
itemDelegateForColumn(self, column)	
itemDelegateForRow(self, row)	
itemsExpandable(self)	
keyPressEvent(self, event)	
keyReleaseEvent(self, a0)	
keyboardGrabber()	
keyboardSearch(self, search)	
killTimer(self, id)	
layout(self)	
layoutDirection(self)	
leaveEvent(self, a0)	
lineWidth(self)	
locale(self)	
logicalDpiX(self)	

continues on next page

Table 3.17 – continued from previous page

logicalDpiY(self)
lower(self)
mapFrom(self, a0, a1)
mapFromGlobal(self, a0)
mapFromParent(self, a0)
mapTo(self, a0, a1)
mapToGlobal(self, a0)
mapToParent(self, a0)
mask(self)
maximumHeight(self)
maximumSize(self)
maximumViewportSize(self)
maximumWidth(self)
metaObject(self)
metric(self, a0)
midLineWidth(self)
minimumHeight(self)
minimumSize(self)
minimumSizeHint(self)
minimumWidth(self)
model(self)
mouseDoubleClickEvent(self, e)
mouseGrabber()
mouseMoveEvent(self, event)
mousePressEvent(self, e)
mouseReleaseEvent(self, event)

continues on next page

Table 3.17 – continued from previous page

move()	
moveCursor(self, cursorAction, modifiers)	
moveEvent(self, a0)	
moveToThread(self, thread)	
nativeEvent(self, eventType, message)	
nativeParentWidget(self)	
nextInFocusChain(self)	
normalGeometry(self)	
objectName(self)	
openPersistentEditor(self, index)	
overrideWindowFlags(self, type)	
overrideWindowState(self, state)	
paintEngine(self)	
paintEvent(self, e)	
paintingActive(self)	
palette(self)	
parent(self)	
parentWidget(self)	
physicalDpiX(self)	
physicalDpiY(self)	
pos(self)	
previousInFocusChain(self)	
property(self, name)	
pyqtConfigure(...)	Each keyword argument is either the name of a Qt property or a Qt signal.
raise_(self)	
receivers(self, signal)	

continues on next page

Table 3.17 – continued from previous page

rect(self)
reexpand(self)
releaseKeyboard(self)
releaseMouse(self)
releaseShortcut(self, id)
removeAction(self, action)
removeEventFilter(self, a0)
render(, sourceRegion, flags, ...)
repaint(-> None -> None)
reset(self)
resetHorizontalScrollMode(self)
resetIndentation(self)
resetVerticalScrollMode(self)
resize()
resizeColumnToContents(self, column)
resizeEvent(self, e)
restoreGeometry(self, geometry)
rootIndex(self)
rootIsDecorated(self)
rowHeight(self, index)
rowsAboutToBeRemoved(self, parent, start, end)
rowsInserted(self, parent, start, end)
rowsRemoved(self, parent, first, last)
saveGeometry(self)
scheduleDelayedItemsLayout(self)
screen(self)

continues on next page

Table 3.17 – continued from previous page

scroll()
scrollBarWidgets(self, alignment)
scrollContentsBy(self, dx, dy)
scrollDirtyRegion(self, dx, dy)
scrollTo(self, index[, hint])
scrollToBottom(self)
scrollToTop(self)
selectAll(self)
selectedIndexes(self)
selectionBehavior(self)
selectionChanged(self, selected, deselected)
selectionCommand(self, index[, event])
selectionMode(self)
selectionModel(self)
sender(self)
senderSignalIndex(self)
setAcceptDrops(self, on)
setAccessibleDescription(self, description)
setAccessibleName(self, name)
setAllColumnsShowFocus(self, enable)
setAlternatingRowColors(self, enable)
setAnimated(self, enable)
setAttribute(self, attribute[, on])
setAutoExpandDelay(self, delay)
setAutoFillBackground(self, enabled)
setAutoScroll(self, enable)

continues on next page

Table 3.17 – continued from previous page

setAutoScrollMargin(self, margin)
setBackgroundRole(self, a0)
setBaseSize()
setColumnHidden(self, column, hide)
setColumnWidth(self, column, width)
setContentsMargins()
setContextMenuPolicy(self, policy)
setCornerWidget(self, widget)
setCurrentIndex(self, index)
setCursor(self, a0)
setDefaultDropAction(self, dropAction)
setDirtyRegion(self, region)
setDisabled(self, a0)
setDragDropMode(self, behavior)
setDragDropOverwriteMode(self, overwrite)
setDragEnabled(self, enable)
setDropIndicatorShown(self, enable)
setEditTriggers(self, triggers)
setEnabled(self, a0)
setExpanded(self, index, expand)
setExpandsOnDoubleClick(self, enable)
setFirstColumnSpanned(self, row, parent, span)
setFixedHeight(self, h)
setFixedSize()
setFixedWidth(self, w)
setFocus()

continues on next page

Table 3.17 – continued from previous page

setFocusPolicy(self, policy)
setFocusProxy(self, a0)
setFont(self, a0)
setForegroundRole(self, a0)
setFrameRect(self, a0)
setFrameShadow(self, a0)
setFrameShape(self, a0)
setFrameStyle(self, a0)
setGeometry()
setGraphicsEffect(self, effect)
setHeader(self, header)
setHeaderHidden(self, hide)
setHidden(self, hidden)
setHorizontalScrollBar(self, scrollbar)
setHorizontalScrollBarPolicy(self, a0)
setHorizontalScrollMode(self, mode)
setIconSize(self, size)
setIndentation(self, i)
setIndexWidget(self, index, widget)
setInputMethodHints(self, hints)
setItemDelegate(self, delegate)
setItemDelegateForColumn(self, column, delegate)
setItemDelegateForRow(self, row, delegate)
setItemsExpandable(self, enable)
setLayout(self, a0)
setLayoutDirection(self, direction)

continues on next page

Table 3.17 – continued from previous page

setLineWidth(self, a0)
setLocale(self, locale)
setMask()
setMaximumHeight(self, maxh)
setMaximumSize()
setMaximumWidth(self, maxw)
setMidLineWidth(self, a0)
setMinimumHeight(self, minh)
setMinimumSize()
setMinimumWidth(self, minw)
setModel(self, model)
setMouseTracking(self, enable)
setObjectName(self, name)
setPalette(self, a0)
setParent()
setProperty(self, name, value)
setRootIndex(self, index)
setRootIsDecorated(self, show)
setRowHidden(self, row, parent, hide)
setSelection(self, rect, command)
setSelectionBehavior(self, behavior)
setSelectionMode(self, mode)
setSelectionModel(self, selectionModel)
setShortcutAutoRepeat(self, id[, enabled])
setShortcutEnabled(self, id[, enabled])
setSizeAdjustPolicy(self, policy)

continues on next page

Table 3.17 – continued from previous page

setSizeIncrement()
setSizePolicy()
setSortingEnabled(self, enable)
setState(self, state)
setStatusTip(self, a0)
setStyle(self, a0)
setStyleSheet(self, styleSheet)
setTabKeyNavigation(self, enable)
setTabOrder(a0, a1)
setTabletTracking(self, enable)
setTextElideMode(self, mode)
setToolTip(self, a0)
setToolTipDuration(self, msec)
setTreePosition(self, logicalIndex)
setUniformRowHeights(self, uniform)
setUpdatesEnabled(self, enable)
setVerticalScrollBar(self, scrollbar)
setVerticalScrollBarPolicy(self, a0)
setVerticalScrollMode(self, mode)
setViewport(self, widget)
setViewportMargins()
setVisible(self, visible)
setWhatsThis(self, a0)
setWindowFilePath(self, filePath)
setWindowFlag(self, a0[, on])
setWindowFlags(self, type)

continues on next page

Table 3.17 – continued from previous page

setWindowIcon(self, icon)
setWindowIconText(self, a0)
setWindowModality(self, windowModality)
setWindowModified(self, a0)
setWindowOpacity(self, level)
setWindowRole(self, a0)
setWindowState(self, state)
setWindowTitle(self, a0)
setWordWrap(self, on)
setupViewport(self, viewport)
sharedPainter(self)
show(self)
showColumn(self, column)
showDropIndicator(self)
showEvent(self, a0)
showFullScreen(self)
showMaximized(self)
showMinimized(self)
showNormal(self)
signalsBlocked(self)
size(self)
sizeAdjustPolicy(self)
sizeHint(self)
sizeHintForColumn(self, column)
sizeHintForIndex(self, index)
sizeHintForRow(self, row)

continues on next page

Table 3.17 – continued from previous page

sizeIncrement(self)
sizePolicy(self)
sortByColumn(self, column, order)
stackUnder(self, a0)
startDrag(self, supportedActions)
startTimer(self, interval[, timerType])
state(self)
statusTip(self)
style(self)
styleSheet(self)
tabKeyNavigation(self)
tabletEvent(self, a0)
testAttribute(self, attribute)
textElideMode(self)
thread(self)
timerEvent(self, event)
toolTip(self)
toolTipDuration(self)
tr(self, sourceText[, disambiguation, n])
treePosition(self)
underMouse(self)
ungrabGesture(self, type)
uniformRowHeights(self)
unsetCursor(self)
unsetLayoutDirection(self)
unsetLocale(self)

continues on next page

Table 3.17 – continued from previous page

update()
updateEditorData(self)
updateEditorGeometries(self)
updateGeometries(self)
updateGeometry(self)
updateMicroFocus(self)
updatesEnabled(self)
verticalOffset(self)
verticalScrollBar(self)
verticalScrollBarPolicy(self)
verticalScrollMode(self)
verticalScrollbarAction(self, action)
verticalScrollbarValueChanged(self, value)
viewOptions(self)
viewport(self)
viewportEvent(self, event)
viewportMargins(self)
viewportSizeHint(self)
visibleRegion(self)
visualRect(self, index)
visualRegionForSelection(self, selection)
whatsThis(self)
wheelEvent(self, a0)
width(self)
widthMM(self)
winId(self)

continues on next page

Table 3.17 – continued from previous page

window(self)
windowFilePath(self)
windowFlags(self)
windowHandle(self)
windowIcon(self)
windowIconText(self)
windowModality(self)
windowOpacity(self)
windowRole(self)
windowState(self)
windowTitle(self)
windowType(self)
wordWrap(self)
x(self)
y(self)

Attributes

AboveItem
AdjustIgnored
AdjustToContents
AdjustToContentsOnFirstShow
AllEditTriggers
AnimatingState
AnyKeyPressed
BelowItem

continues on next page

Table 3.18 – continued from previous page

Box
COLOR_HIGHLIGHT
CollapsingState
ContiguousSelection
CurrentChanged
DoubleClicked
DragDrop
DragOnly
DragSelectingState
DraggingState
DrawChildren
DrawWindowBackground
DropOnly
EditKeyPressed
EditingState
EnsureVisible
ExpandingState
ExtendedSelection
HLine
IgnoreMask
InternalMove
MoveDown
MoveEnd
MoveHome
MoveLeft
MoveNext

continues on next page

Table 3.18 – continued from previous page

MovePageDown
MovePageUp
MovePrevious
MoveRight
MoveUp
MultiSelection
NoDragDrop
NoEditTriggers
NoFrame
NoSelection
NoState
OnItem
OnViewport
Panel
PdmDepth
PdmDevicePixelRatio
PdmDevicePixelRatioScaled
PdmDpiX
PdmDpiY
PdmHeight
PdmHeightMM
PdmNumColors
PdmPhysicalDpiX
PdmPhysicalDpiY
PdmWidth
PdmWidthMM

continues on next page

Table 3.18 – continued from previous page

Plain	
PositionAtBottom	
PositionAtCenter	
PositionAtTop	
Raised	
ScrollPerItem	
ScrollPerPixel	
SelectColumns	
SelectItems	
SelectRows	
SelectedClicked	
Shadow_Mask	
Shape_Mask	
SingleSelection	
StyledPanel	
Sunken	
VLine	
WinPanel	
activated	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
clicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
collapsed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
customContextMenuRequested	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
destroyed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
doubleClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
doubleLeftClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
entered	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

continues on next page

Table 3.18 – continued from previous page

expanded	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
iconSizeChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
leftClicked	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
objectNameChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
pressed	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
staticMetaObject	
viewportEntered	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowIconTextChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL
windowTitleChanged	int = ..., arguments: Sequence = ...) -> PYQT_SIGNAL

CHAPTER

FOUR

LABSCRIPT SUITE COMPONENTS

The *labscript suite* is modular by design, and is comprised of:

Table 4.1: Python libraries

labscript — Expressive composition of hardware-timed experiments
labscript-devices — Plugin architecture for controlling experiment hardware
labscript-utils — Shared modules used by the *labscript suite*

Table 4.2: Graphical applications

runmanager — Graphical and remote interface to parameterized experiments
blacs — Graphical interface to scientific instruments and experiment supervision
lyse — Online analysis of live experiment data
runviewer — Visualize hardware-timed experiment instructions

PYTHON MODULE INDEX

r

INDEX

Symbols

A

B

C

D

E

F

G

H

P

Q

R

S

T

U

W