Elixys Web Server Interface

Contents

[Overview 2](#_Toc296991171)

[Configuration 2](#_Toc296991172)

[User 3](#_Toc296991173)

[Client State 4](#_Toc296991174)

[Server State 4](#_Toc296991175)

[Sequences 6](#_Toc296991176)

[Components 9](#_Toc296991177)

[CASSETTE 9](#_Toc296991178)

[ADD 9](#_Toc296991179)

[EVAPORATE 10](#_Toc296991180)

[TRANSFER 10](#_Toc296991181)

[ELUTE 10](#_Toc296991182)

[REACT 11](#_Toc296991183)

[PROMPT 11](#_Toc296991184)

[MOVE 11](#_Toc296991185)

[INSTALL 12](#_Toc296991186)

[COMMENT 12](#_Toc296991187)

[ACTIVITY 12](#_Toc296991188)

[Reagents 12](#_Toc296991189)

[Run 13](#_Toc296991190)

[Run Sequence 14](#_Toc296991191)

[Manual Run 15](#_Toc296991192)

[Thoughts 18](#_Toc296991193)

# Overview

This document describes the internal interface between the web server and the core Python server. All of the functions take the name of the current user as a parameter. Many of these functions return a simple value indicating success or failure which can be used to inform the user if something went wrong. A user with sufficient privileges can then view the system logs to obtain additional error information in the event of a failure.

# Configuration

**GetConfiguration()** – Returns details of the Elixys system.

Parameters:

* Username – Name of the current user.

Returns:

* Name – Name of this Elixys system (e.g. “Mini cell 3”).
* Version – System version (e.g. “2.0”).
* Debug – Boolean values that specifies if the client will display additional debug information to the user.

**GetSupportedOperations()** – Returns an array of operations supported by this system.

Parameters:

* Username – Name of the current user.

Returns:

* Supported operations – Array of operations supported by this system (e.g. “Add”, “Evaporate”, etc.).

# Server State

**GetServerState()** – Returns the state of the server.

Parameters:

* Username – Name of the current user.

Returns:

* Run status:
  + Mode – Run mode. Return values include “idle”, “runsequence” and “manualrun”.
  + Sequence ID – ID of the currently running sequence.
  + Username – Name of the user that is operating the system.
  + Status – String describing the current system status (e.g. “Reacting, 8:23 minutes”).
  + Active reactor – The active reactor number.
* Hardware state:
  + Pressure regulators – Details of each pressure regulator in the system:
    - Name – String describing the pressure regulator (e.g. “Main value pressure”).
    - Set pressure – The target pressure in millimeters of mercury.
    - Actual pressure – The actual pressure in millimeters of mercury.
  + Cooling – Boolean values that specifies if the cooling system is on.
  + Vacuum – Boolean value that specifies if the vacuum system is on.
  + Door – Boolean value that specifies if the main door is open.
  + Reagent Robot – Details of the reagent robot:
    - Position – Descriptive string of the robot position.
    - Raw X – Gives the raw X position of the robot in millimeters.
    - Raw Y – Gives the raw Y position of the robot in millimeters.
    - Actuator – String that specifies the state of the actuator. Possible values are “up”, “down” and “indeterminate”.
    - Gripper – Boolean value that specifies if the robot gripper is closed.
  + Reactors – Details of each reactor:
    - Number – The reactor number.
    - Set temperature – The set temperature of the reactor in degrees Celsius.
    - Actual temperature – The actual temperature of the reactor in degrees Celsius.
    - Position – The reactor position.
    - Vial – The vial state. Possible values are “up”, “down” and “indeterminate”.
    - Activity – The last know radiation activity level of the active reactor in millicuries.
    - Activity time – The time the activity was last measured.
    - Evaporation valves – Value that specifies if the evaporation values (nitrogen and vacuum) are open. Possible values are “open” and “closed”.
    - Transfer valve – Value that specifies the state of the transfer valve.
    - Reagent 1 transfer valve – Value that specifies the state of the first reagent transfer valve.
    - Reagent 2 transfer valve – Value that specifies the state of the second reagent transfer valve.
    - Stopcock 1 valve – Value that specifies the state of the first stopcock valve.
    - Stopcock 2 valve – Value that specifies the state of the second stopcock valve.
    - Stopcock 3 valve – Value that specifies the state of the third stopcock valve.

# Sequences

**GetSequence()** – Returns details of a sequence and all components.

Parameters:

* Username – Name of the current user.
* Sequence ID – String that uniquely identifies the sequence.

Returns:

* Sequence metadata – Returns metadata for this sequence as described above in **GetSequenceList()**.
* Component information – Returns the following information for each sequence component:
  + Component Type – String that specifies the type of component. Possible values are the subheading under **Components** (e.g. “EVAPORATE”).
  + Component ID – Unique ID that is used by the client to refer to the component when communicating with the server.
  + Component Name – Display name of this component.
  + Reactor – The reactor associated with this component.
  + Reactor description – Description of the reactor field.
  + Reactor validation – Contains a string describing the reactor validation.
  + Additional details – Each component type contain additional information as documented below under **Components**.

**DeleteSequenceComponent()** – Deletes a component from a sequence.

Parameters:

* Username – Name of the current user.
* Sequence ID – The ID of the sequence to associated with the component.
* Component ID – The ID of the component to delete.

Returns:

* Result – Boolean value (true on success, false otherwise).

**CopySequence()** – Duplicates a sequence. The source sequence may be either a saved sequence or a manual run. The sequence copy will always be a saved sequence.

Parameters:

* Username – Name of the current user.
* Sequence ID – The ID of the sequence to copy.
* Sequence metadata – Metadata of the new sequence:
  + Name – Sequence name.
  + Comment – Any comment associated with the sequence.
  + Creator – User that created the sequence.

# Run

**GetRunState()** – Returns the run state of the system.

Parameters:

* Username – Name of the current user.

Returns:

* Run state – String describing the state of the system. This string may be delimited and contain state-specific information that will be understood by the web server (e.g. “SEQUENCE.1.15”) but should be treated by the core server as just a string.

**SaveRunState()** – Updates the run state of the system.

Parameters:

* Username – Name of the current user.
* Run state – String describing the state of the system.

Returns:

* Result – Boolean value (true on success, false otherwise).

**GetRunUser()** – Returns the user currently running the system.

Parameters:

* Username – Name of the current user.

Returns:

* Run username – Name of the user currently operating the system or an empty string if the system is not running.

## Run Sequence

**RunSequence()** – Starts executing a sequence from the database.

Parameters:

* Username – Name of the current user.
* Sequence ID – Unique ID of the sequence to run.

Returns:

* Result – Boolean value (true on success, false otherwise).

**AbortRun()** – Aborts the run that is in progress.

Parameters:

* Username – Name of the current user.

Returns:

* Result – Boolean value (true on success, false otherwise).

**ContinueRun()** – Continues the run that has paused for a Prompt or Install unit operation.

Parameters:

* Username – Name of the current user.

Returns:

* Result – Boolean value (true on success, false otherwise).

## Manual Run

**StartManualRun()** – Starts a manual run and create a new manual run sequence in the database.

Parameters:

* Username – Name of the current user.

Returns:

* Sequence ID – Unique ID of the newly created manual run sequence or blank on error.

**PerformOperation()** – Performs a unit operation that has been added to the manual run sequence using **SaveSequenceComponent()**.

Parameters:

* Username – Name of the current user.
* Component ID – The unique ID of the sequence component. This unit operation will always be the last one in the sequence and will have just recently been added.
* Sequence ID – The unique ID of the manual run sequence that is associated with the component.

Returns:

* Result – Boolean value (true on success, false otherwise).

**AbortOperation()** – Abort the unit operation that is in progress.

Parameters:

* Username – Name of the current user.

Returns:

* Result – Boolean value (true on success, false otherwise).

**ContinueOperation()** – Continues the operation that has paused for a Prompt or Install unit operation.

Parameters:

* Username – Name of the current user.

Returns:

* Result – Boolean value (true on success, false otherwise).

**FinishManualRun()** – Completes the manual run and releases the lock on the Elixys system.

Parameters:

* Username – Name of the current user.

Returns:

* Result – Boolean value (true on success, false otherwise).