Front end API

This page gathers all the API calls that can be used by the backend.

Backend -> Front end

Warning

The functions can only be called if they are available on the `web/build` directory, therefore if you make a change using `npm run serve` won't show it, you will need to rebuild the front end with `npm run buildWeb` or by using `npm run start`.

Note

These functions can only be called after eel is initiated with `eel.init()`.

Core API

Collection of all functions/API calls available to the backend. You can find them in the `bolinho_api/core.py` file.

The JavaScript file can be found in the api folder.

`def ping()` :

Tries to ping the bolinho front-end, returns 1 if it worked

Python usage example

```
from bolinho_api.core import core_api

while True:
    try:
        if core_api.ping():
            print("got a ping!")
            break
        pass
    except:
        eel.sleep(1)
```

def get_config_params() :

Tries to ping the bolinho front-end, returns 1 if it worked

Python usage example

```
from bolinho_api.core import core_api

config = core_api.get_config_params()
current_save_version = config["configVersion"]
print(current_save_version)
```

This function is located at `src/web/src/App.js`

def go_to_experiment_page() :

Asks the front end to go to the experiment page.

Returns 1 if succeeded.

Python usage example

```
from bolinho_api.core import core_api

change_pages = True
if change_pages:
    core_api.go_to_experiment_page()
```

def go_to_home_page() :

Asks the front end to go to the home page.

Returns 1 if succeeded.

Python usage example

```
from bolinho_api.core import core_api

change_pages = True
if change_pages:
    core_api.go_to_home_page()
```

`def show_connect_prompt() :`

Asks the front end to show the connection prompt.

The connection prompt is used to select the serial port.

Returns 1 if succeeded.

Python usage example

```
from bolinho_api.core import core_api

config = core_api.get_config_params()
device_port = config["port"]

while not device_port:
    core_api.show_connect_prompt()
    device_port = config["port"]
```

UI API

Collection of all functions/API calls available to the backend for UI in general. You can find them in the `bolinho_api/ui.py` file.

The JavaScript file can be found in the api folder.

`def success_alert(text) :`

Uses React-Toastify to create an success alert.

Python usage example

```
from bolinho_api.ui import ui_api

ui_api.success_alert("Success!")
```

`def error_alert(text) :`

Uses React-Toastify to create an error alert.

Python usage example

```
from bolinho_api.ui import ui_api

UIapi.error_alert("Error!")
```

`def prompt_user(description, options, callback_func) :`

Prompts the user with a 'description', and shows the 'options' to the user.

The result is passed to the callback_function

Python usage example

```
from bolinho_api.ui import ui_api

def get_result(result):
    if result == "yes":
        print("The user chose yes")
    print("The user chose no")

UIapi.prompt_user(
    description="Do you want to pay 1000?",
    options=["yes", "no"],
    callback_func= get_result,
    )
```

Experiment page API

Collection of all functions/API calls available to the backend for the **experiment** routine. You can find them in the `bolinho_api/experiment.py` file.

The JavaScript file can be found at `web/src/api/contexts/ExperimentPageContext.tsx` .

`def get_load_percentage() :`

Asks the front for the current load percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Returns the load percentage value

Python usage example

```
from bolinho_api.experiment import experiment_api

print(experiment_api.get_load_percentage())
```

def set_load_percentage(newValue) :

Sets the current load percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Python usage example

```
from bolinho_api.experiment import experiment_api

for number in range(100):
    experiment_api.set_load_percentage(number)
    eel.sleep(0.1)
```

def get_time_percentage() :

Asks the front for the current time percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Returns the load percentage value

Python usage example

```
from bolinho_api.experiment import experiment_api

print(experiment_api.get_time_percentage())
```

def set_time_percentage(newValue) :

Sets the current time percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Python usage example

```
from bolinho_api.experiment import experiment_api

experiment_api.set_time_percentage(22)
```

def get_distance_percentage() :

Asks the front for the current distance percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Returns the load percentage value

Python usage example

```
from bolinho_api.experiment import experiment_api

print(experiment_api.get_distance_percentage())
```

def set_distance_percentage(newValue) :

Sets the current distance percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Python usage example

```
from bolinho_api.experiment import experiment_api

experiment_api.set_distance_percentage(22)
```

def get_delta_load_percentage() :

Asks the front for the current delta load percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Returns the load percentage value

Python usage example

```
from bolinho_api.experiment import experiment_api

print(experiment_api.get_delta_load_percentage())
```

`def set_delta_load_percentage(newValue) :`

Sets the current delta load percentage.

This variable is shown to the user in a progress bar. And is usually between 0-100.

Python usage example

```
from bolinho_api.experiment import experiment_api

experiment_api.set_delta_load_percentage(22)
```

`def get_experiment_parameters() :`

Asks the front for the current experiment parameters.

Returns a formatted string

Python usage example

```
from bolinho_api.experiment import experiment_api

print(experiment_api.get_experiment_parameters())
```

`def set_experiment_parameters(newValue) :`

Sets the current experiment parameters.

Receives a formatted string.

Python usage example

```
from bolinho_api.experiment import experiment_api

experiment_api.set_experiment_parameters("Experiment 202 <br/> Load cell: lxi92")
```

def get_readings() :

Asks the front for the current Readings.

Returns an object of type Readings, this object gathers all the current readings of the machine. Such as Current z axis position, current load, and status

Python usage example

```
from bolinho_api.experiment import experiment_api

reading_obj = experiment_api.get_readings()

print(reading_obj.status)
```

def set_readings(newValue) :

Sets the current Readings.

Receives an object of type Readings, this object gathers all the current readings of the machine. Such as Current z axis position, current load, and status.

This function dumps the object to a JSON and sends it to the front end

Python usage example

```
from bolinho_api.experiment import experiment_api
from bolinho_api.jsClasses import Readings

new_machine_readings = Readings(299, 87, "not good")

experiment_api.set_readings(new_machine_readings)
```

def get_description() :

Asks the front for the current description.

Returns a formatted string

Python usage example

```
from bolinho_api.experiment import experiment_api

print(experiment_api.get_description())
```

`def set_description(newValue) :`

Sets the current description.

Receives a formatted string.

Python usage example

```
from bolinho_api.experiment import experiment_api

experiment_api.set_description("New Experiment description")
```

`def get_material() :`

Asks the front for the current Material.

Returns an object of type Material.

Python usage example

```
from bolinho_api.experiment import experiment_api

material_obj = experiment_api.get_material()

print(material_obj.name)
```

`def set_material(newValue) :`

Sets the current Material.

Receives an object of type Material

This function dumps the object to a JSON and sends it to the front end

Python usage example

```
from bolinho_api.experiment import experiment_api
from bolinho_api.jsClasses import Readings

current_material = Material(
    id=23,
    name="aço 22",
    batch="1",
    experimentArray=[1, 3, 2],
    supplier="Metalúrgica JOSÉ",
    extraInfo="Cilindro",
)

experiment_api.set_material(current_material)
```