Otii Automation Toolbox

User manual

QOITECH

QOITECH

Otii Toolboxes

Elevate Otii instruments with additional software toolboxes for scripting capabilities and battery profiling, emulation, testing and validation.

Otii Toolboxes are additional software licenses for Otii instruments that can be purchased perpetual or as monthly and yearly subscription.

The two avaliable Otii Toolboxes are:

Otii Automation Toolbox

This license elevates the Otii software with scripting capabilities and opens for endless automated use cases.

Otii Battery Toolbox

This license enables capabilities for battery profiling, emulation, and cycling - everything needed to pick the right battery for your project.

QOITECH

Otii Automation Toolbox

The key to functional, regression, and benchmark testing. Ideal for continuous integration, tracking energy consumption from prototyping to production.

Scripting in any language using JSON API

- Otii Automation Toolbox makes Otii instruments programmable.
- Control your Otii Arc/Ace with the built-in TCP-server from any language or system that supports communication over TCP sockets, using a JSON-based API.
- Find an example of how to integrate Otii in Jenkins using the Python programming language in the Help section of the Otii application.
- Great for your continuous integration set-up, to keep track of your system's energy consumption throughout the whole development cycle, from prototyping to production.

Scripting statistics API

- The Otii core computational engine is designed to efficiently make calculations of large sets of data.
- The TCP-API is constantly being expanded with methods that will simplify your energy optimization tasks.
- Examples of API additions are recording_get_info that returns information about a recording, and recording_get_statistics that returns the minimum, maximum, average and energy consumed over a specified time range.

Packaged scripting modules available for Python, Matlab, C#, Java

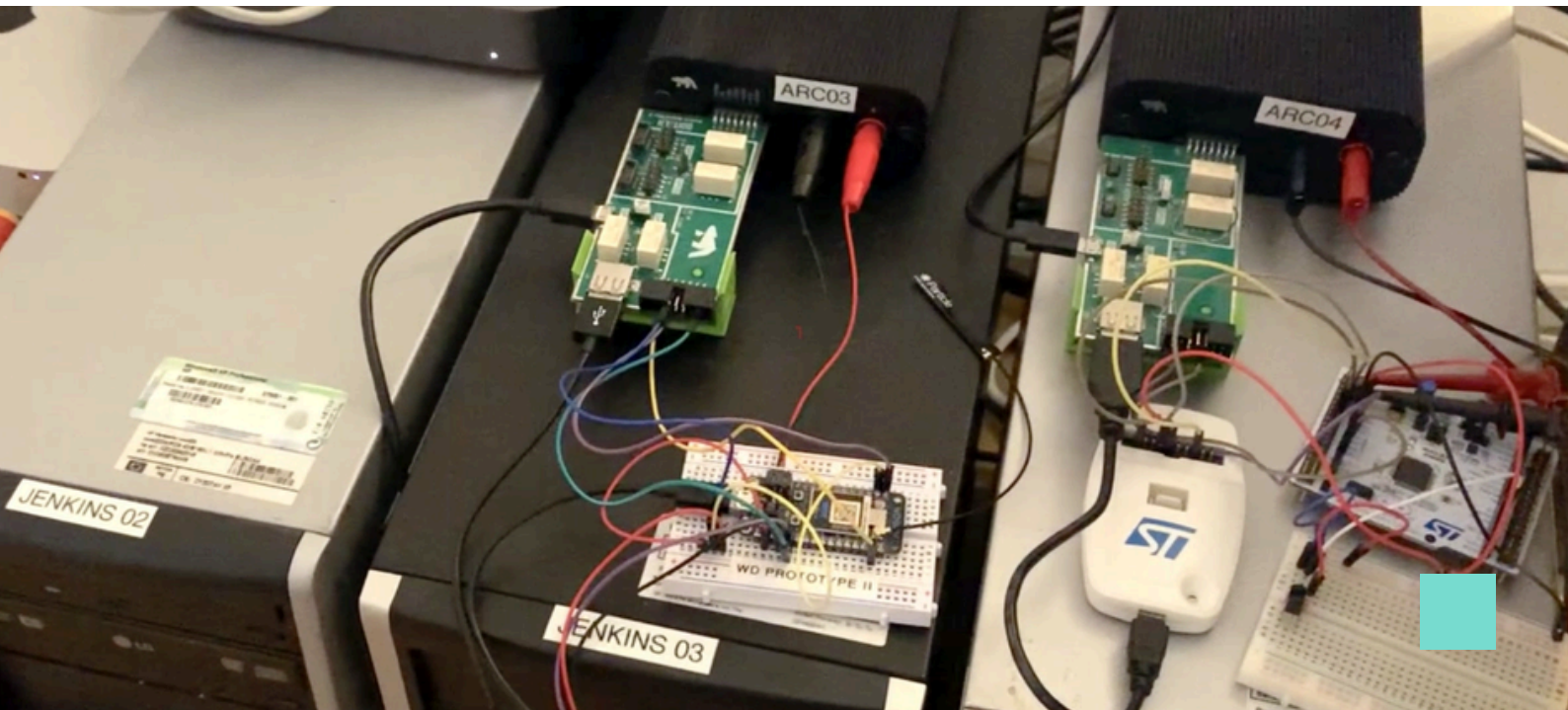
- Write your scripts using the JSON-based API or use our packaged scripting modules available for popular languages like Phyton, Java, C# and Matlab.
- Find the modules on Qoitech GitHub.

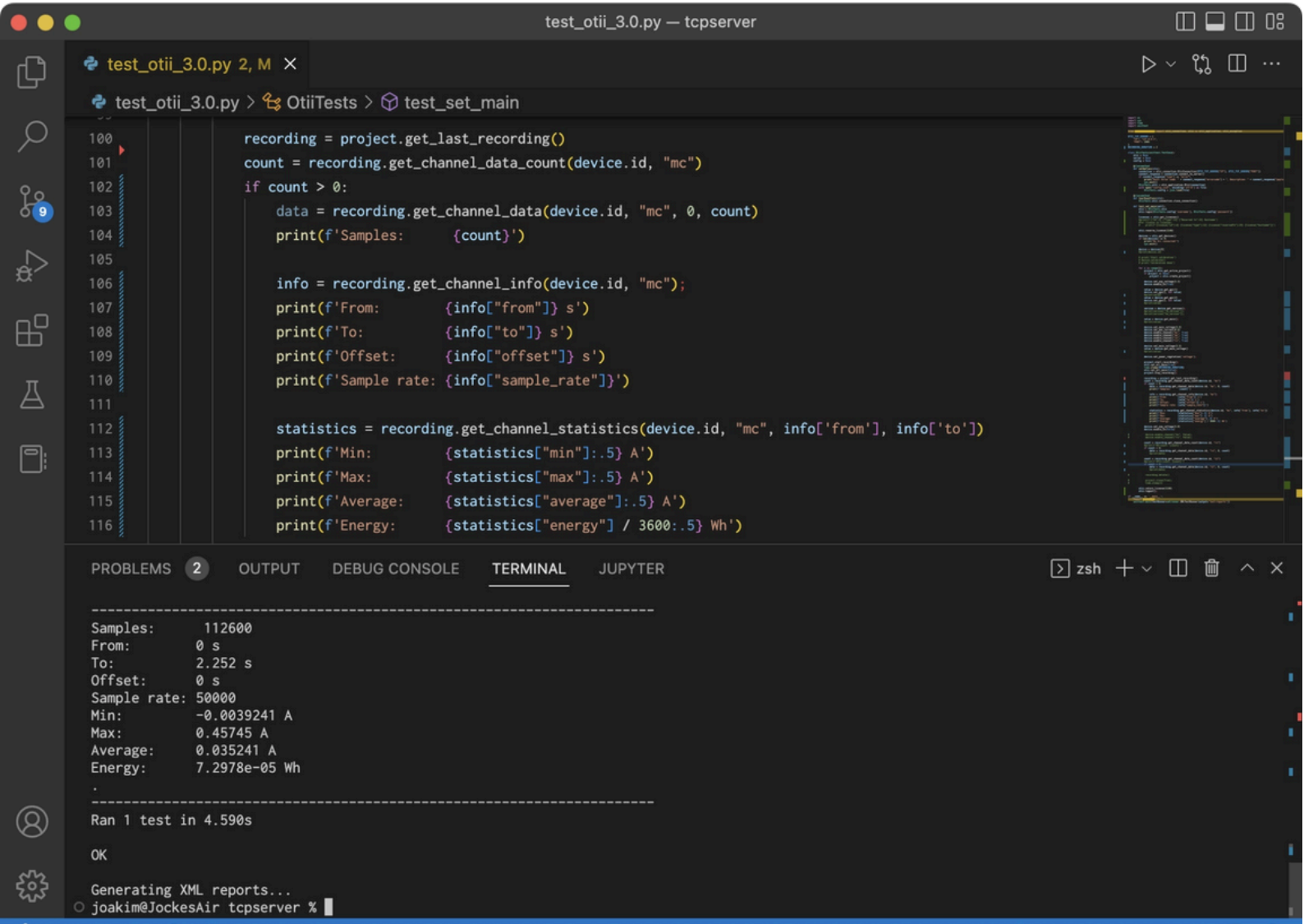
Command line tools

- The toolbox includes a command-line tool, that makes it possible to run test scripts without the UI.
- Run your script in your favorite continuous integration (CI) environment
- Create a test setup in a remote location
- The TCP API includes methods to log in, log out, and handle Otii licenses directly from the script, making it easier to include the script in an automated environment.

Shareable license

- Otii Automation Toolbox is shareable. The license can be assigned only at one user at the time. The license management is done in Otii User Management.
- The license can also be used in offline mode.






```java
try {
    // Remove any open project
    Project project = otii.getActiveProject();
    if (project != null) {
        project.close(true);
        project = null;
    }
    assertNull(project);

    Arc[] arcs = otii.getDevices();
    assertEquals(1, arcs.length);
    Arc arc = arcs[0];

    arc.setMainVoltage(3.0);
    arc.setExpVoltage(3.0);
    arc.setMaxCurrent(0.5);
    arc.setUartBaudrate(115200);
    arc.enableUart(true);
    arc.enableExpPort(true);
```

Java

```matlab
try
    % Configure first available device
    devices = otii.devices();
    assert(~isempty(devices.payload.devices), 'No available devices');
    device = otii.get_device(devices.payload.devices(1));
    device.enable_channel('mc');
    device.enable_channel('mv');

    % Create a new project if needed
    if ~otii.has_project()
        fprintf('Creating new project\n');
        otii.create_project();
    end

    % Save the project
    otii.save_project('Testar.otii', true, true, @(progress) disp(progress));

    % Record for a few seconds
    otii.start_recording();
    otii.set_all_main(true);
    pause(2);
    otii.set_all_main(false);
    otii.stop_recording();

    % Read recorded data
    recordings = otii.list_recordings();
    recording = recordings(2);

    count = otii.get_channel_data_count(device.device_id, recording.recording_id, 'mc');
    data = otii.get_channel_data(device.device_id, recording.recording_id, 'mc', 0, 1000);

    timestamp = 0:0.25:249.75;
    plot(timestamp, data);

    % Save the project
    otii.save_project('Testar.otii', true, true, @(progress) disp(progress));

catch ME
    disp(ME);
end
```

MATLAB

```python
def setup_otii():
    connection = otii_connection.OtiiConnection(OTII_TCP_SERVER["IP"
    connect_response = connection.connect_to_server()
    if connect_response["type"] == "error":
        print("Exit! Error code: " + connect_response["errorcode"]
        sys.exit()
    otii = otii_application.Otii(connection)

    devices = otii.get_devices()
    if len(devices) == 0:
        print("No Arc connected!")
        sys.exit()
    devices = [device for device in devices if device.name == ARC_NA
    if len(devices) != 1:
        print("Expected to find exactly 1 device named {0}, found {1
        sys.exit()
    arc = devices[0]

    arc.set_range("high")
    arc.set_main_voltage(3.3)
    arc.set_exp_voltage(3.3)
    arc.set_max_current(0.5)
    arc.set_adc_resistor(0.2)
    arc.set_uart_baudrate(115200)
    arc.enable_uart(True)
    arc.enable_exp_port(True)
    arc.enable_5v(True) # The switch board is powered by the Otii +5V pin

    return otii, arc
```

Python

Otii Automation Toolbox

All these features require an Automation Toolbox License.

This toolbox features:

- Scripting in any language using the Otii TCP Server API
- Packaged scripting modules available from Github for Python, C#, Java and Matlab
- Otii Server, a headless server that easily can be integrated in test setups

Explore features sets

Otii TCP Server	Otii 3 Desktop	Otii Server

Otii TCP Server

It is possible to control Otii from another application using the Otii TCP Server API.

Using this API you can control Otii from any application that includes support for standard TCP sockets.

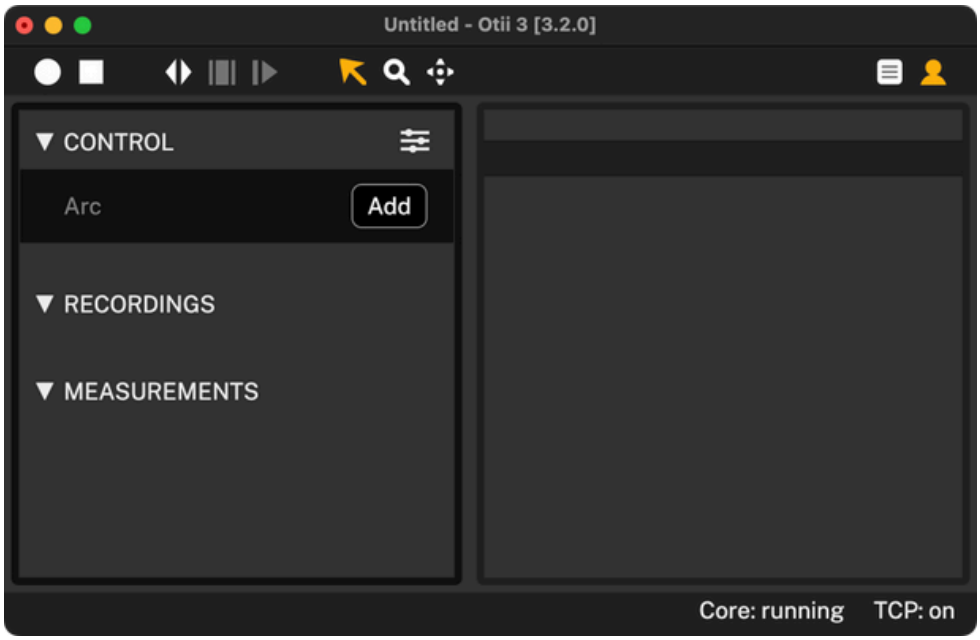
Otii server is available in both the Otii 3 Desktop App and in the Otii Server application. You can not run both at the same time on the same computer.

At Qoitech's Github page you will also find wrappers for a few popular languages like Python, C#, Java & Matlab.

Otii 3 Desktop App

If you have reserved an Automation Toolbox License the TCP-server is automatically started when you start the Otii dekstop application.

You can see the status of the TCP-server in the lower right corner of the application.



Otii 3 Desktop App | TCP status

You now can control the application using an external script.

Otii Server

Command Lines Interface

You can also start the TCP server using otii_server, the command line version of Otii.

The server is packaged within the application and to find where it is located on your system, select Help › Find otii_server in Otii 3.

If you run otii_server with the --help option you will see all available options:

```
# otii_server --help

Options:
  -V, --version                output the version number
  -v, --verbose                verbose output
  -c, --config <config-file>   configuration file
  -h, --help                   display help for command
```

To start a server run the command:

```
./otii_server
```

The server is by default listening on host address 127.0.0.1 and port 1905. If you want to change the default settings, or need to add settings for a proxy, create a config.json file:

```
{
    "tcpserver": {
        "host": "127.0.0.1",
        "port": 1907
    },
    "proxy": {
        "host": "192.168.1.1",
        "port": "8080",
        "auth": {
            "username": "johndoe",
            "password": "mypassword"
        }
    }
}
```

and start the server with:

```
./otii_server --config config.json
```

The server will run until you stop it by pressing Ctrl-C.

In a future release Otii will ship with tools that will make it possible to run otii_server as a Windows Service, or a daemon in Ubuntu and macOS.

User management

User management | Desktop application

Read more about logging in and reserving licenses in the Otii 3 Desktop App here:

- User management

User management | Desktop application

It is possible to use the TCP-API directly in the test script for user management:

```
#!/usr/bin/env python3
'''
If you want the script to login and reserve a license automatically
add a configuration file called credentials.json in the current folder
using the following format:

    {
        "username": "YOUR USERNAME",
        "password": "YOUR PASSWORD"
    }

Alternatively you can set the environment variables OTII_USERNAME and OTII_PASSWORD.

'''
from otii_tcp_client import otii_client

# Connect and login to Otii 3
client = otii_client.OtiiClient()
with client.connect() as otii:
    # INSERT TEST CODE HERE
```

The client.connect() is used to connect, login and reserve licenses. It works like this:

- If the TCP server isn't already logged in, the credentials will be read from a credentials.json file in the current directory, or from the environment variable OTII_USERNAME and OTII_PASSWORD
- If there is no Automation Toolbox license reserved and there is a license available, it will automatically be reserved.

- By default the system will only try to reserve an Automation Toolbox license. If you need to reserve another Toolbox as well add all the licenses you need to the licenses parameter:

```
client.connect(licenses = [ 'Automation', 'Battery' ])
```

- When disconnecting from Otii all licenses that were implicitly reserved will be returned, and if the connect method logged in to the system, it will log out again.

- If you want to manually reserve a specific license, you use an empty array for the licenses parameter:

```
#!/usr/bin/env python3
from otii_tcp_client import otii_client

# Connect and login to Otii 3
client = otii_client.OtiiClient()
with client.connect(licenses = []) as otii:
    # List all licenses
    licenses = otii.get_licenses()
    for license in licenses:
        print(f'{license["id"]:4d} {license["type"]:12} {license["reserved_to"]:15} {lice

    # Reserve a license
    otii.reserve_license(licenses[0])

    # INSERT TEST CODE HERE

    # Return license
    otii.return_license(licenses[0])
```

User management | Python tool

You can do the user management from the command line using our python module. You install the module with:

```
python3 -m pip install otii_tcp_client
```

You can then use the otii_control tool for user management:

```
python3 -m otii_tcp_client.otii_control --help

usage: otii_control.py [-h] {login,logout,list-licenses,reserve-license,return-license} .

Otii Control

options:
  -h, --help            show this help message and exit

commands:
  {login,logout,list-licenses,reserve-license,return-license}
    login               Log in to Qoitech server
    logout              Log out from Qoitech server
    list-licenses       List all available licenses
    reserve-license     Reserve license
    return-license      Return license
```

And here is an example bash script:

```
#!/usr/bin/env bash

python3 -m otii_tcp_client.otii_control login --username johndoe --password mypassword
python3 -m otii_tcp_client.otii_control reserve-license --id 1234
./my_test.py
python3 -m otii_tcp_client.otii_control return-license --id 1234
python3 -m otii_tcp_client.otii_control logout
```

Resources

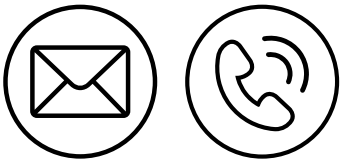
Otii Ace Pro datasheet

Otii Arc Pro datasheet

Otii Battery Toolbox documentation

Otii Automation Toolbox documentation

Otii Product Suite documentation



Contact us

To discuss how Otii Product Suite can best suit your team/s and how to tailor it for optimal utilization and cost efficiency, please contact our sales team at

sales@qoitech.com

SE +46 46 261 50 50

For technical questions and support, please contact us at

support@qoitech.com

We’re here to help!

The Qoitech Team

QOITECH

qoitech.com/otii

Qoitech creates the most effective solutions for energy consumption analysis and optimization, battery life analytics and prediction, and testing of batteries and energy harvesters, applicable across all industries. Founded as subsidiary of Sony, Qoitech is a Swedish company with global reach.