What is Ingeniamotion?

Ingeniamotion is a library that works over ingenialink and aims to simplify the interaction with Ingenia's drives.

How it works?

All ingeniamotion functionalities works through the MotionController class. So, first of all we should instantiate a MotionController object.

```
from ingeniamotion import MotionController
mc = MotionController()
```

Now, `mc` is our MotionController instance.

Then, we should connect some servos.

```
# In case we want to connect only one servo
mc.communication.connect_servo_eoe("192.168.2.22", "eve-net_1.7.0.xdf")

# In case we want to connect more servos
mc.communication.connect_servo_eoe("192.168.2.22", "eve-net_1.7.0.xdf",
                                   alias="servo_one")
mc.communication.connect_servo_eoe("192.168.2.23", "eve-net_1.7.0.xdf",
                                   alias="servo_two")
# The "alias" field will allow to reference these servos in the future.
# The "alias" can be whatever we want to use as identifier.
```

Now, the servos are ready and we can work with them.

We then can apply some configurations:

```
# If we have only one servo
mc.configuration.release_brake()
# By default it uses the axis 1
```

Or we can execute some tests or calibrations:

```
mc.tests.digital_halls_test(servo="servo_one", axis=1)

mc.tests.commutation(servo="servo_two", axis=1)
```

MotionController namespaces

MotionController functionalities are group in the following namespaces.

Communication

This namespace has all the basic communication functions with the servo: connect, read or write a register, load firmware, etc.

Configuration

Here we will find functions to configure the servo: load or save configuration, configure limits, feedbacks, brake settings, etc.

Motion

In this namespace we will find all the functions that will help us to move the servos.

Capture

This namespace will help us to work with monitoring and similar features.

Info

Functions to get register information from dictionary.

Errors

Namespace to manage drive errors and get errors data.

Tests

The functions of this namespace will help us to lunch some tests for the commissioning process.

Common exceptions

```
KeyError: "Servo 'default' is not connected"
```

This KeyError exception is raised whenever we use a function that interacts with the drive but no drive is connected.

```
TypeError: 'NoneType' object is not subscriptable
```

This TypeError exception is raised when we provide a function with the wrong servo axis number.

```
ingenialink.exceptions.ILError
```

This ingenialink.exceptions.ILError exception is raised when the drive gets abruptly disconnected.