ServoMaster: Metadata: Rationale & Summary

Table of contents

1. Definition

The metadata is a way to obtain the information about the servo controller (or a servo), and, sometimes, adjust its behavior. The reason for existence of the metadata, as opposed to fixed method set API, is that it is not possible to foresee what features and/or properties may appear in the hardware of tomorrow, and this project's life is planned to be quite long.

Apparent overhead (string manipulation vs. direct method call) is in fact irrelevant because the calls supported by the metadata access mechanism are infrequent (how many times per second do you have to find out the controller's manufacturer's URL? Or set the terminal positions for the servo?), however, this mechanism offers great flexibility keeping the implementation overhead to bare minimum (just a glimpse: dictionary lookup plus an anonymous class serving as a request handler).

1.1. Features & Properties

Two flavors of metadata are supported: feature and property. The feature is something that is either present or absent. If it is supported, it can be switched on or off. The property is something that can be measured, described and possibly changed. An example of a feature: whether the controller is able to support the silent mode. An example of a property: how long the controller will stay inactive before going into silent mode.

1.2. Using features and properties

The feature or property identifier is normally a full or partial URL. The full URL points to the page containing the support documentation for this feature or property, for example, http://servomaster.sourceforge.net/meta/controller/precision. Since this is quite cumbersome, partial URLs will be accepted as well by the API calls. For this particular example, the identifier will look like `controller/precision`.

The full list of the metadata features and properties can be found on the navigation bar on the left.

1.3. Exceptions

`UnsupportedOperationException`	If the feature not supported or the property is not present, on both get and set operations.
`IllegalAccessError`	If the feature or property value can't be changed because it's read-only (immutable).

IOException	If there is a hardware related problem.
IllegalStateException	A special case. This exception will be thrown by some controller drivers (Phidgets in particular) if the device was not connected during the controller instantiation and hasn't been connected since. As soon as the device is detected, the metadata for it can be created and is accessible from that point on.

Table 1: Metadata Exceptions

2. Examples

2.1. Getting the set of available features & properties

```
ServoController sc = getController();

for ( Iterator i = sc.getMeta().getFeatures(); i.hasNext(); ) {

    System.out.println("Feature: " + i.next());
}
for ( Iterator i = sc.getMeta().getProperties(); i.hasNext(); ) {

    System.out.println("Property: " + i.next());
}
```

2.2. Setting the servo range

```
Servo s = getServo();

s.getMeta().setProperty("servo/range/min", "890");
s.getMeta().setProperty("servo/range/max", "2100");
```

2.3. Telling the controller to be quiet

```
ServoController sc = getController();

try {

    sc.setFeature("controller/silent", true);
    sc.setProperty("controller/silent", "5000");
```

```
} catch ( UnsupportedOperationException ex ) {

    // Nope, it doesn't support that
}
```

Note:

In this case the *feature* and the *property* names are the same, however, the meaning is different: the feature allows to enable or disable itself, whereas the property changes the operating parameters.