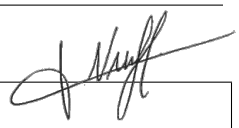

```
>>> tag.value = datetime.datetime.today()

>>> metadata.write()
```

Similarly to reading the value of a tag, one can set either the *raw_value* or the `value` (which will be automatically converted to a correctly formatted byte string by pyexiv2).

You can also add new tags to the metadata by providing a valid key and value pair (see exiv2's documentation for a list of valid XMP tags):

```
>>> key = 'Xmp.xmp.Label'
>>> value = 'A beautiful picture.'
>>> metadata[key] = pyexiv2.XmpTag(key, value)
```

As a handy shortcut, you can always assign a value for a given key regardless of whether it's already present in the metadata. If a tag was present, its value is overwritten. If the tag was not present, one is created and its value is set:

```
>>> metadata[key] = value
```

Accessing to the tags of type *XmpSeq <Property>Detail*. Example with *Xmp.plus.Licensor* wich is type *XmpSeq LicensorDetail*:

```
>>> base = "Xmp.plus.Licensor"
>>> # Always chek if the tag already exists
>>> try:
>>>     seq = data[base]
>>> except KeyError:
>>>     # Tag not set, create one. Note the value [""]
>>>     tag = pyexiv2.xmp.XmpTag(base, [""])
>>>     data[base] = tag
>>> key = "".join([base, "[1]/plus:LicensorID"])
>>> datum = pyexiv2.xmp.XmpTag(key, "https://iptc.org")
>>> data[key] = datum
>>> key = "".join([base, "[1]/plus:LicensorName"])
>>> datum = pyexiv2.xmp.XmpTag(key, "John Doe")
>>> data[key] = datum
>>> key = "".join([base, "[1]/plus:LicensorCountry"])
>>> datum = pyexiv2.xmp.XmpTag(key, "USA")
>>> data[key] = datum
>>> key = "".join([base, "[1]/plus:LicensorCity"])
>>> datum = pyexiv2.xmp.XmpTag(key, "Washington")
>>> data[key] = datum
```

If you need to write custom metadata, you can register a custom XMP namespace:

```
>>> pyexiv2.xmp.register_namespace('http://example.org/foo/', 'foo')
>>> metadata['Xmp.foo.bar'] = 'baz'
```

Note that a limitation of the current implementation is that only simple text values can be written to tags in a custom namespace.

A custom namespace can be unregistered. This has the effect of invalidating all tags in this namespace for images that have not been written back yet:

```
>>> pyexiv2.xmp.unregister_namespace('http://example.org/foo/')
```