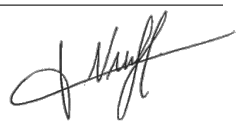
1.5 pyexiv2.xmp



This module provides the classes `XmpTag` and `XmpValueError` and the following five functions to handle the XMP parser and name spaces.

`pyexiv2.xmp.`**`initialiseXmpParser`**`()`
Initialise the xmp parser.

Calling this method is usually not needed, as encode() and decode() will initialize the XMP Toolkit if necessary.

This function itself still is not thread-safe and needs to be called in a thread-safe manner (e.g., on program startup).

`pyexiv2.xmp.`**`closeXmpParser`**`()`
Close the xmp parser.

Terminate the XMP Toolkit and unregister custom namespaces.

Call this method when the XmpParser is no longer needed to allow the XMP Toolkit to cleanly shutdown.

`pyexiv2.xmp.`**`register_namespace`**`(`*name*, *prefix*`)`
Register a custom XMP namespace.

Overriding the prefix of a known or previously registered namespace is not allowed.

Arguments:

- *name* str() The name of the custom namespace (ending with a /), typically a URL (e.g. http://purl.org/dc/elements/1.1/)
- *prefix* str() The prefix for the custom namespace (keys in this namespace will be in the form Xmp.{prefix}.{something})

Raises:

- *ValueError* – if the name doesn't end with a /
- *KeyError* – if a namespace already exist with this prefix

`pyexiv2.xmp.`**`unregister_namespace`**`(`*name*`)`
Unregister a custom XMP namespace.

A custom namespace is identified by its name, not by its prefix.

Attempting to unregister an unknown namespace raises an error, as does attempting to unregister a builtin namespace.

Arguments:

- *name* str() The name of the custom namespace (ending with a /), typically a URL (e.g. http://purl.org/dc/elements/1.1/)

Raises:

- *ValueError* – if the name doesn't end with a /
- *KeyError* – if the namespace is unknown or a builtin namespace

`pyexiv2.xmp.`**`unregister_namespaces`**`()`
Unregister all custom XMP namespaces.

Builtin namespaces are not unregistered.

This function always succeeds.

`class` `pyexiv2.xmp.`**`XmpTag`**