activereader

Release 0.0.2

Aaron Schroeder

Oct 28, 2022

API REFERENCE:

CHAPTER

ONE

ACTIVEREADER.GPX MODULE

.gpx file reader architecture.

See also:

GPX schema
Official documentation for GPX elements and file structure.

Garmin's GPX trackpoint extension schema
XML file describing the schema for Garmin's additional GPX trackpoint elements.

activereader.gpx.Gpx(lxml_elem)	Represents an entire .gpx file object.
activereader.gpx.Track(lxml_elem)	An ordered list of trackpoints describing a path.
activereader.gpx.Segment(lxml_elem)	Holds a list of trackpoints which are logically connected in order.
activereader.gpx.Trackpoint(lxml_elem)	Represents a single data sample corresponding to a point in time.

class activereader.gpx.**Gpx**(*lxml_elem*)

Bases: *ActivityElement*

Represents an entire .gpx file object.

TAG = 'gpx'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

classmethod from_file(*file_obj*)

Initialize a Gpx element from a file-like object.

Parameters
file_obj (*str, bytes, io.StringIO, io.BytesIO*) – File-like object. If str, either filename or a string representation of XML object. If str or StringIO, the encoding should not be declared within the string.

Returns
An instance initialized with the `_Element` that was read in.

Return type
Gpx

See also:

https://lxml.de/tutorial.html#the-parse-function

property start_time

Timestamp at start of recording.

See also:

Timestamps

Type
datetime.datetime

property tracks

All element descendents with tag name "trk".

Type
list of *Track*

property segments

All element descendents with tag name "trkseg".

Type
list of *Segment*

property trackpoints

All element descendents with tag name "trkpt".

Type
list of *Trackpoint*

property routepoints

All element descendents with tag name "rtept".

Type
list of Routepoint

class activereader.gpx.**Track**(*lxml_elem*)

Bases: *ActivityElement*

An ordered list of trackpoints describing a path.

TAG = 'trk'

XML tag name of the element.

If an instance of the class is initialized from a lxml.etree._Element with any other tag name, a TypeError will be raised.

property segments

All element descendents with tag name "trkseg".

Type
list of *Segment*

property trackpoints

All element descendents with tag name "trkpt".

Type
list of *Trackpoint*

class activereader.gpx.**Segment**(*lxml_elem*)

Bases: *ActivityElement*

Holds a list of trackpoints which are logically connected in order.

To represent a single GPS track where GPS reception was lost, or the GPS receiver was turned off, start a new Track Segment for each continuous span of track data.

TAG = 'trkseg'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

property trackpoints

All element descendents with tag name "trkpt".

Type
list of *Trackpoint*

class activereader.gpx.**Trackpoint**(*lxml_elem*)

Bases: *ActivityElement*

Represents a single data sample corresponding to a point in time.

The most granular of data contained in the file.

TAG = 'trkpt'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

property time

Timestamp when trackpoint was recorded.

See also:

Timestamps

Type
datetime.datetime

property altitude_m

Elevation of ground surface in meters above sea level.

Type
float

property hr

Heart rate.

Type
int

property cadence_rpm

Cadence in RPM.

See also:

Cadence

Type
int

property lat

Latitude in degrees N (-90 to 90).

Type
float

property lon

Longitude in degrees E (-180 to 180).

Type
float

CHAPTER

TWO

ACTIVEREADER.TCX MODULE

.tcx file reader architecture.

Originated in heartandsole back in the day.

See also:

Garmin's TCX schema
XML file describing the schema for TCX files.

Garmin's ActivityExtension schema
XML file describing Garmin's extensions to the TCX schema.

activereader.tcx.Tcx(lxml_elem)	Represents an entire .tcx file object.
activereader.tcx.Activity(lxml_elem)	TCX files representing a run should only contain one Activity.
activereader.tcx.Lap(lxml_elem)	Represents one bout from {start/lap} -> {lap/stop}.
activereader.tcx.Track(lxml_elem)	In a running TCX file, there is typically one Track per Lap.
activereader.tcx.Trackpoint(lxml_elem)	Represents a single data sample corresponding to a point in time.

class activereader.tcx.**Tcx**(*lxml_elem*)

Bases: *ActivityElement*

Represents an entire .tcx file object.

TAG = 'TrainingCenterDatabase'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

classmethod from_file(*file_obj*)

Initialize a Tcx element from a file-like object.

Parameters
file_obj (*str, bytes, io.StringIO, io.BytesIO*) – File-like object. If str, either filename or a string representation of XML object. If str or StringIO, the encoding should not be declared within the string.

Returns
An instance initialized with the `_Element` that was read in.

Return type
Tcx

See also:

https://lxml.de/tutorial.html#the-parse-function

property activities

All element descendents with tag name "Activity".

Type
list of *Activity*

property laps

All element descendents with tag name "Lap".

Type
list of *Lap*

property tracks

All element descendents with tag name "Track".

Type
list of *Track*

property trackpoints

All element descendents with tag name "Trackpoint".

Type
list of *Trackpoint*

class activereader.tcx.**Activity**(*lxml_elem*)

Bases: *ActivityElement*

TCX files representing a run should only contain one Activity.

Contains one or more *Lap* elements.

TAG = 'Activity'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

property start_time

Timestamp for activity start time.

See also:

Timestamps

Type
datetime.datetime

property sport

Activity sport.

Restricted to "Running", "Biking", or "Other" according to Garmin's TCX file schema.

Type
str

property device

Device brand name.

Type
str

property device_id

Device ID - specific to the individual device.

Type
int

property laps

All element descendents with tag name "Lap".

Type
list of *Lap*

property tracks

All element descendents with tag name "Track".

Type
list of *Track*

property trackpoints

All element descendents with tag name "Trackpoint".

Type
list of *Trackpoint*

class activereader.tcx.**Lap**(*lxml_elem*)

Bases: *ActivityElement*

Represents one bout from {start/lap} -> {lap/stop}.

There is at least one lap per activity file, created by the *start* button press and ended by the *stop* button press. Hitting the *lap* button begins a new lap. Hitting the pause button stops data recording, but the same lap resumes after the pause.

Made up of 1 or more *Track* in the XML structure.

TAG = 'Lap'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

property start_time

Timestamp of lap start.

See also:

Timestamps

Type
datetime.datetime

property total_time_s

Total lap time, in seconds, as reported by the device.

This is timer time, not elapsed time; it does not include any time when the device is paused.

See also:

Starts, stops, and pauses

Type
float

property distance_m

Total lap distance, in meters, as reported by the device.

See also:

Distance

Type
float

property max_speed_ms

The maximum speed achieved during the lap as reported by the device, in meters per second.

Type
float

property avg_speed_ms

The average speed during the lap as reported by the device, in meters per second.

Type
float

property hr_avg

average heart rate during the lap as reported by the device.

Type
float

property hr_max

maximum heart rate during the lap as reported by the device.

Type
float

property cadence_avg

average cadence during the lap as reported by the device, in RPM.

See also:

Cadence

Type
float

property cadence_max

maximum cadence during the lap as reported by the device, in RPM.

See also:

Cadence

Type
float

property calories

Calories burned during the lap as approximated by the device.

Type
float

property tracks

All element descendents with tag name "Track".

Type
list of *Track*

property trackpoints

All element descendents with tag name "Trackpoint".

Type
list of *Trackpoint*

class activereader.tcx.**Track**(*lxml_elem*)

Bases: *ActivityElement*

In a running TCX file, there is typically one Track per Lap.

As far as I can tell, in a running file, Tracks and Laps are one and the same; when a Lap starts or ends, so does its contained Track.

Made up of 1 or more *Trackpoint* in xml file.

TAG = 'Track'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

property trackpoints

All element descendents with tag name "Trackpoint".

Type
list of *Trackpoint*

class activereader.tcx.**Trackpoint**(*lxml_elem*)

Bases: *ActivityElement*

Represents a single data sample corresponding to a point in time.

The most granular of data contained in the file.

TAG = 'Trackpoint'

XML tag name of the element.

If an instance of the class is initialized from a `lxml.etree._Element` with any other tag name, a TypeError will be raised.

property time

Timestamp when trackpoint was recorded.

See also:

Timestamps

Type
datetime.datetime

property lat

Latitude in degrees N (-90 to 90).

Type
float

property lon

Longitude in degrees E (-180 to 180).

Type
float

property altitude_m

Elevation of ground surface in meters above sea level.

Type
float

property distance_m

Cumulative distance from the start of the activity, in meters.

See also:

Distance

Type
float

property hr

Heart rate.

Type
int

property speed_ms

Speed in meters per second.

Type
float

property cadence_rpm

Cadence in RPM.

See also:

Cadence

Type
int

CHAPTER

THREE

ACTIVEREADER.BASE MODULE

Provides the tools to build classes corresponding to XML elements in GPX and TCX files.

class activereader.base.**ActivityElement**(*lxml_elem*)

Base class for creating compositions with _Element.

TCX and GPX files are both XML documents that follow their own particular schema. ActivityElement subclasses provide access to the underlying XML elements' data, attributes, descendent elements.

Parameters

lxml_elem (_Element) – XML element object that has been read in by the lxml.etree package. The tag name of the element must match the *TAG* attribute of this class, or else a TypeError will be raised.

TAG = 'element'

XML tag name of the element.

If an instance of the class is initialized from a lxml.etree._Element with any other tag name, a TypeError will be raised.

classmethod _add_attr_properties(***property_keys_types*)

Add properties to cls that access specific keys using *get_attr()*.

Parameters

- **cls** – Class to add the properties to.
- ****property_keys_types** – kwargs mapping property_name : tuple(key, data_type)
  - property_name will be added to cls.
  - key is the attribute name within the contained lxml element whose text will be retrieved.
  - data_type is the type that the subelement's text will be converted to. A python type, or datetime.datetime to read a timestamp using dateutil.parser.isoparse().

Examples

```
>>> class MyElement(ActivityElement):
...   [...]
>>> MyElement._add_attr_properties(start_time=('StartTime', datetime.datetime))
```

classmethod _add_data_properties(***property_paths_types*)

Add properties to cls that access specific paths using *get_data()*.

Parameters

- **cls** – Class to add the properties to.
- ****property_paths_types** – kwargs mapping property_name : tuple(path, data_type)
  - property_name will be added to cls.
  - path is the tag name or path of the desired subelement of the contained lxml element whose text will be retrieved
  - data_type is the type that the subelement's text will be converted to. A python type, or datetime.datetime to read a timestamp using `dateutil.parser.isoparse()`.

Examples

```
>>> class MyElement(ActivityElement):
...   [...]
>>> MyElement._add_data_properties(time=('Time', datetime.datetime))
```

See also:

pandas.core.accessor.PandasDelegate._add_delegate_accessors

classmethod _add_descendent_properties(***descendent_classes*)

Add properties to cls that return all descendents matching a path.

Parameters

- **cls** – Class to add the properties to.
- ****descendent_classes** – kwargs mapping property_name : descendent class.
  - property_name will be added to cls.
  - All values must be a subclass of *ActivityElement*.

Examples

```
>>> class MyElement(ActivityElement):
...   [...]
>>> MyElement._add_descendent_properties(subelements=MySubElement)
```

get_attr(*key*, *conv_type=<class 'str'>*)

Retrieve data using contained lxml element's `get()` and convert its type.

Parameters

- **key** (*str*) – The attribute name within the contained lxml element.
- **conv_type** (*data type*) – Data type that the attribute text will be converted to. Python type, or datetime.datetime to read a time using `dateutil.parser.isoparse()`. Defaults to `str`.

Returns

The retrieved attribute data in the requested type, or None if the contained lxml element does not have an attribute named `key`.

Examples

```
>>> e = ActivityElement(etree.fromstring(
...   '<element StartTime="2021-02-26T19:51:07.000Z"></element>
... ))
>>> e.get_attr('StartTime', conv_type=datetime.datetime)
datetime.datetime(2021, 2, 26, 19, 51, 7, tzinfo=tzutc())
```

get_data(*path*, *conv_type=<class 'str'>*)

Retrieve data using the contained lxml element's `findtext()` and convert its type.

Parameters

- **path** (*str*) – The tag name or path of the desired subelement of the contained lxml element whose text will be retrieved. Text will be retrieved from the first matching subelement.
- **conv_type** (*data type*) – Data type that the element text will be converted to. Python type, or datetime.datetime to read a time using `dateutil.parser.isoparse()`. Defaults to `str`.

Returns

The retrieved data in the requested type, or None if no subelements match `path`.

Examples

Text can be converted to another Python type:

```
>>> e_lat = ActivityElement(etree.fromstring(
...   '<element><Position><LatitudeDegrees>40.0</LatitudeDegrees></Position></
↪element>'
... ))
>>> e_lat.get_data('Position/LatitudeDegrees', conv_type='float')
40.0
```

Or a timestamp can be read in:

```
>>> e_time = ActivityElement(etree.fromstring(
...   '<element><Time>2021-02-26T19:51:08.000Z</Time></element>'
... ))
>>> e_time.get_data('Time', conv_type=datetime.datetime)
datetime.datetime(2021, 2, 26, 19, 51, 8, tzinfo=tzutc())
```

class activereader.base.**XmlReader**(*filepath_or_buffer*, *ext='XML'*)

XmlReader provides an interface for reading in a XML file (eg GPX, TCX).

_get_data_from_filepath(*filepath_or_buffer*)

The method {reader}.from_file accepts four input types:

1. filepath (string-like)
2. bytes
3. file-like object (e.g. open file object, StringIO, BytesIO)
4. GPX string

This method turns (1) and (2) into (3) to simplify the rest of the processing. It returns input types (3) and (4) unchanged.

Raises FileNotFoundError if the input is a string ending in `.{ext}` but no such file exists.

Ref:
https://github.com/pandas-dev/pandas/blob/v1.5.1/pandas/io/json/_json.py#L837

_preprocess_data(*data*)

At this point, the data either has a `read` attribute (e.g. an open file object, a StringIO, or a BytesIO) or is a string that is a XML document. Any of these are acceptable inputs to `lxml.etree.parse()`, so this method does not change the data currently.

Ref:
https://github.com/pandas-dev/pandas/blob/v1.5.1/pandas/io/json/_json.py#L821

read()

Read the whole input into a `lxml.etree._Element`

activereader.base.**add_xml_attr**(***property_keys_types*)

Add properties to a class that access data using `get_attr()`.

This provides an alternative usage to directly calling *ActivityElement._add_attr_properties()* below a class definition.

Parameters
****property_keys_types** – kwargs mapping property_name : tuple(key, data_type)

- property_name will be added to the decorated class.
- key is the attribute name within the contained lxml element whose text will be retrieved.
- data_type is the type that the subelement's text will be converted to. A python type, or datetime.datetime to read a timestamp using `dateutil.parser.isoparse()`.

Returns
A class decorator.

Return type
callable

Examples

```
>>> @add_xml_attr(start_time=('StartTime', datetime.datetime))
... class MyElement(ActivityElement):
...   [...]
```

See also:

ActivityElement._add_attr_properties()

activereader.base.**add_xml_data**(***property_paths_types*)

Add properties to a class that access data using `get_data()`.

This provides an alternative usage to directly calling *ActivityElement._add_data_properties()* below a class definition.

Parameters

- ****property_paths_types** – kwargs mapping property_name : tuple(path, data_type)
- **class.** (*- property_name will be added to the decorated*) –

- **the** (*- path is the tag name or path of the desired subelement of*) – contained lxml element whose text will be retrieved
- **converted** (*- data_type is the type that the subelement's text will be*) – to. A python type, or datetime.datetime to read a timestamp using `dateutil.parser.isoparse()`.

Returns
A class decorator.

Return type
callable

Examples

```
>>> @add_xml_data(time=('Time', datetime.datetime))
... class MyElement(ActivityElement):
...   [...]
```

See also:

ActivityElement._add_data_properties()

activereader.base.**add_xml_descendents**(***descendent_classes*)

Add properties to a class that return all descendents matching a path.

This provides an alternative usage to directly calling *ActivityElement._add_descendent_properties()* below a class definition.

Parameters
****descendent_classes** – kwargs mapping property_name : descendent class.

- Each key will be a property name added to the decorated class.
- All values must be a subclass of *ActivityElement*.

Returns
A class decorator

Return type
callable

Examples

```
>>> @add_xml_descendents(subelements=MySubElement)
... class MyElement(ActivityElement):
...   [...]
```

See also:

ActivityElement._add_descendent_properties()

activereader.base.**create_attr_prop**(*key*, *conv_type=<class 'int'>*)

Add property inside an ActivityElement class definition that accesses data using *get_attr()*.

Parameters

- **key** (*str*) – The attribute name within the contained lxml element.

- **conv_type** (*data type*) – Data type that the element text will be converted to. Python type, or datetime.datetime to read a time using `dateutil.parser.isoparse()`. Defaults to `int`.

Returns
accessor for underlying lxml element's attribute data.

Return type
property

Examples

```
>>> class MyElement(ActivityElement):
...   my_prop = create_attr_prop('Time', datetime.datetime)
...   """Property docstring goes here (if desired)"""
...   [...]
```

`activereader.base.`**create_data_prop**(*path*, *conv_type=<class 'int'>*)

Add property inside an ActivityElement class definition that accesses data using *get_data()*.

Parameters

- **path** (*str*) – The tag name or path of the desired subelement of the contained lxml element whose text will be retrieved. Text will be retrieved from the first matching subelement.
- **conv_type** (*data type*) – Data type that the element text will be converted to. Python type, or datetime.datetime to read a time using `dateutil.parser.isoparse()`. Defaults to `int`.

Returns
accessor for the underlying lxml element's data.

Return type
property

Examples

```
>>> class MyElement(ActivityElement):
...   my_prop = create_data_prop('Time', datetime.datetime)
...   """Property docstring goes here (if desired)."""
...   [...]
```

`activereader.base.`**create_descendent_prop**(*descendent_class*)

Add descendent list accessor property inside an ActivityElement class definition.

The property returns all the current element's descendents matching the tag name defined in the descendent class.

Parameters
descendent_class (`ActivityElement`) – Must be a subclass of *ActivityElement*.

Returns
accessor for underlying lxml element's descendent list.

Return type
property

Examples

```
>>> class MyElement(ActivityElement):
...     my_prop = create_descendent_prop(MySubElement)
...     """Property docstring goes here (if desired)"""
...     [...]
```

CHAPTER

FOUR

NOTES ABOUT FILE DATA

4.1 Device data

4.1.1 Distance

What your device reports as distance is usually not strictly the distance between recorded GPS points. Most devices rely on a combination of GPS readings and accelerometer data, inferring the true distance travelled using a cool process known as *sensor fusion*. Since GPS readings have a bit of noise to them (see GPS section), the device's reported distance is typically a little shorter than the sum of GPS point-to-point distances (unless the GPS was malfunctioning badly or had no signal at all, in which case the distance is inferred from the accelerometer data alone).

4.1.2 Cadence

For whatever reason, activity files report running cadence in RPM. In running, we typically deal with cadence in strides per minute, which is twice the RPM value. I think the use of RPM is a relic of the fact that cyclists, who use RPM, tend to adopt new technology faster than runners. They were data-obsessed way before we strapped on our first GPS-enabled watch. They got here first, and they wanted cadence in RPM, and these file formats want to stay consistent.

4.1.3 Timestamps

Usually timestamp strings are timezone-aware, but that depends on how the input file is formatted. Files exported from Garmin Connect have timestamps in Coordinated Universal Time, but different services or devices may generate different timestamp formats. $dateutil.parser.isoparse$ processes the timestamp strings from the file.

4.2 TCX Files

4.2.1 Starts, stops, and pauses

Unfortunately, although TCX files create a new lap element when the lap button is pressed, they do not explicitly create an artifact when the pause button is pressed. Trackpoints are simply not recorded. The only definitive proof that the pause button was pressed is that the lap's `total_time_s()` is less than its elapsed time (the difference between subsequent laps' `start_time()`).

CHAPTER

FIVE

RELEASE NOTES

This is the list of changes to activereader between each release. For full details, see the commit logs.

5.1 Version 0.0

5.1.1 What's new in 0.0.2 (October 28, 2022)

These are the changes in activereader 0.0.2. See *Release notes* for a full changelog including other versions of activereader.

Enhancements

Reading route files

ActivityElement creation methods *Tcx.from_file* and *Gpx.from_file* can now read route files in addition to activity files. Previously, these methods were not reading in the data properly from these files, which are formatted slightly differently than activity files. The failure was quiet: the list of trackpoints would simply be empty, for example.

Create `XmlReader`

A common class that is used by *Tcx.from_file* and *Gpx.from_file*.

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

a

INDEX

Symbols

A

C

D

F

G

H

L

M

R

S

T

X