

# GPX module

The GPX module is a Geotools plugin to read, and eventually to write, Global Positioning System exchange (GPX) format files. The module implements a Geotools DataStore responsible for files with the `.gpx` extension. The module has been added to Geotools in the summer of 2007.

## The GPX format

The GPS exchange format is an XML based file format used for data interchange by GPS consumer grade equipment and software. The core features of the GPX format are:

- Fixed Coordinate System: geographic
- Fixed CRS: WGS84
- Fixed Data Dictionary: waypoints, tracks or routes
- XML format

The coordinates are all stored as geographic coordinates related to the WGS84 coordinate referencing system. For example, a waypoint might be defined as:

```
<wpt lat="39.921055008" lon="3.054223107"/>.
```

The different structures described, which make up the "Data Dictionary", are limited to waypoints, which are point locations, tracks, which are ordered sequences of points, and routes, which are like tracks but differ conceptually: usually tracks are records of an actual journey and routes describe a potential itinerary.

Because the GPX format is file based, the format is not optimized for random access, that is the format can only be read and written linearly. The format therefore does not have the advantages of a database.

More information on this file format can be found on the [GPX wikipedia page](#).

## Module architecture

The module can be split into two distinct parts.

The first part of the module is independent of Geotools; this part parses the `.gpx` file and creates an in-memory representation. The module assumes that all GPX files can fit into memory since GPX files are generally used to describe short trips or excursions.

The second part of the module provides the actual interface to the Geotools library. This part defines:

- a `FileDataStoreFactorySpi`
- a `DataStore`
- a `FeatureReader` for waypoints
- a `FeatureReader` for tracks.

## Proposed packages

<code>org.geotools.data.gpx</code>	GeoTools interface implementations
<code>org.geotools.data.gpx.memory</code>	data beans and the file parser code
<code>org.geotools.data.gpx.test</code>	JUnit test cases

## Design decisions

The initial implementation during the summer of 2007 provides a subset of the full possible functionality (providable by a DataStore, and the gpx format). Two main restrictions are that the DataStore is read only and the module only interprets waypoints and tracks. The module's development started as part of an other project, where fast development was more important than full functionality. That's why the following design issues in the initial implementation of the module, could be reconsidered:

- The parser uses XppReader, which is part of the [XStream](#) project, and which in turn uses [XPP](#), a pull parser. [*pro: have experience in it, con: external dependency*]
- The data container bean currently stores the geometry, name, description, comment and date values from the file. [*these were just enough for the other project, and as I saw, these would almost always be enough*]
- The DataStore when created, creates two FeatureClasses, which describe the same features every time: a point class, and a track class.
- The DataStore parses the GPX file in the constructor, so if a parser error occurs, the creation of the DataStore itself fails. [*pro: prevents deferred errors*]
- The FeatureReader implementations convert the data beans to Feature instances. A waypoint's geometry becomes a Point with lat/lon/elevation, and a track's becomes a MultiLineString with lat/lon/elevation coordinates.
- The structure of the module is minimalistic in the sense that it depends highly on the default implementations of the interfaces. [*for the sake of fast development, and that is what default implementations is there for. As this module assumes to work with small data, this should cause no performance issues.*]

## Development directions

The module's development continues with the followings (not ordered):

- The XML parsing code should be switched over to the Geotools parsers, this reduces external dependencies and increases consistency between modules.
- The data beans can be extended, to support every possible gpx attributes.
- The FeatureClass instances might be changed to static objects, if that is a valid option. [*I need support in deciding this*]
- The date field of the features might be stored as the M value of the geometries' coordinates. [*just an idea, I have no pro/contra for that*]
- Support for gpx route feature class.
- Write mode: a FeatureWriter implementation has to be written for every feature class, that converts the Feature objects back to the data beans. This way the memory representation of the data could be modified. Flushing the in-memory data back to the file could be implemented as a non-standard call on GpxDataStore. [*Or is there a standard way of the flushing? I need support in this too.*]