GIS File Tag Importer

Import tags, channels, nodes, and device points from GeoJSON, Shapefile, and KML/KMZ files.

Reference Solution Designer Settings and Tools Import Tags GIS File

Version 10.1.5+

The GIS File Tag Importer reads geographic feature files and produces one FrameworX tag per feature, with an optional device channel, node, and point linked to the tag. Each feature contributes its attributes as naming tokens so you build tag names, addresses, and descriptions from the source data.

Use this wizard when an operations team hands you a GeoJSON export of solar panels, a KML of water-treatment assets, or a Shapefile of pipeline segments, and you want FrameworX tags created for every asset without manual entry.

Supported File Formats

Format	Extensions	Geometry	Notes
GeoJSON (RFC 7946)	.geojson, .json	Point, LineString, Polygon	Streaming parse. Handles 100,000+ features.
Esri Shapefile	.shp with companion .dbf, .shx, .prj, or a .zip bundle	Point, LineString, Polygon	ZIP is extracted transparently. The .dbf is required.
KML / KMZ	.kml, .kmz	Point, LineString, LinearRing, Polygon, MultiGeometry	KMZ extraction is transparent. The first .kml inside the archive is used.

Coordinates are assumed to be WGS84 (latitude and longitude). Line and polygon geometry is reduced to its centroid. Elevation and Z values are ignored. Coordinate reprojection is not performed in 10.1.5.

Import Process

Open **Solution Import Tags GIS File**. The wizard has five steps.

Step 1. File Select

1. Click **Browse** and choose a file.
2. The parser is auto-detected from the extension. Step 1 runs a fast metadata scan and shows the feature count, distinct layer list, distinct attribute list, and coordinate system.
3. Review the summary. If the file is empty or unreadable, the error is shown here before you advance.

Step 2. Protocol

1. Select a **Protocol** from the list of registered device protocols.
2. Set the **Channel Name Pattern** and the **Node Name Pattern** using the naming tokens in the table below.
3. Set the **Station Address Pattern** for the node.

Step 3. Mapping

1. Set the **Tag Name Pattern**. Default is *{Name}*.
2. Choose the **Tag Type** (Analog, Digital, Text, and so on).
3. Map the **Description Attribute** to an attribute key from the source file.
4. Set the **Address Pattern** used to build each device point's address.
5. Pick **Access Type** (ReadOnly, ReadWrite, WriteOnly).
6. Set the **Update Existing** toggle. Default is on.
7. Optionally click **Save Profile** to reuse this configuration on future imports.

Step 4. Preview

The wizard shows the first 50 features with the resolved tag name, channel name, node name, and address. A duplicate-count banner warns if any resolved tag names already exist in the solution.

Step 5. Generate

Click **Start**. The wizard streams features from the source file into the device-creation pipeline and updates the counters live. Click **Cancel** to stop. Features already processed are kept.

Naming Tokens

Every pattern field (tag name, channel, node, address, description) supports the tokens below. Tokens are replaced per feature during generation.

Token	Source	Example
{Name}	Feature name. From *properties.name*, *properties.NAME*, or *properties.Name* for GeoJSON, from *Placemark.Name* for KML, from the *Name* DBF column for Shapefile.	Inverter_01
{Layer}	KML Folder or Document name, DBF layer inferred from the base filename, GeoJSON *properties.layer*.	Inverters
{Feature Type}	Optional sub-classification. Usually empty for GIS input.	
{Id}	Source-assigned unique ID if present (FID, OBJECTID, GeoJSON *properties.id*).	42
{Index}	Ordinal within the file, zero-based.	0, 1, 2, ...
{Lat}	Latitude, formatted to six decimals.	38.729420
{Lon}	Longitude, formatted to six decimals.	-9.139500
{Attr: key}	Value from *feature.Attributes[key]*. Empty string if the key is missing.	{Attr:device_id} INV-00042

Import Options

Option	Description	Default
Tag Name Pattern	Pattern used for each new tag name.	{Name}
Tag Type	Data type applied to created tags.	Analog
Description Attribute	Attribute key whose value populates the tag Description column.	(none)
Protocol	Device protocol used for the channel.	ModbusTCPIP
Channel Name Pattern	Channel name pattern. Channels are shared across features that resolve to the same name.	{Layer} _Channel
Node Name Pattern	Node name pattern. Nodes are shared across features with the same resolved node name.	{Name}
Station Address Pattern	Protocol-specific station address used on the node.	(empty)
Address Pattern	Address written to the device point.	{Attr:Address}
Access Type	Access mode for the device point.	ReadWrite
Update Existing	When on, re-importing a file with unchanged names updates existing rows. When off, rows with an existing name are skipped.	On

---

File Format Reference

GeoJSON

Source	Maps to
properties.id or root id	Feature Id
properties.name, properties.NAME, properties.Name	Feature Name (first match wins)
properties.layer, properties.Layer	Feature Layer (first match wins)

properties.* (all other keys)	Feature Attributes
Point coordinates	Longitude, Latitude
LineString, Polygon	Centroid of points, or centroid of outer ring

Nested objects under *properties* are JSON-stringified into a single attribute value. UTF-8 BOM is handled transparently.

Shapefile

Source	Maps to
.dbf columns	Feature Attributes
.prj content	Coordinate system, informational only
.shp geometry	Centroid of the geometry for every feature
Base filename	Feature Layer

Companion files (.dbf, .shx, .prj) must sit alongside the .shp in the same folder, or be packaged in a .zip. A missing .dbf is a hard error reported at Step 1.

KML and KMZ

Source	Maps to
Placemark.Name	Feature Name
Parent Folder.Name or Document.Name	Feature Layer
ExtendedData/Data elements	Feature Attributes
SchemaData (typed schema)	Feature Attributes
Point, LineString, LinearRing, Polygon, MultiGeometry	Centroid of the geometry

---

Tutorial

Import 20 solar inverters from GeoJSON and verify the tags appear in Data Explorer.

1. Open a solution in Designer.
2. Open the **Datasets** module once so ModbusTCPIP is registered as a protocol for this solution.
3. Choose **Solution Import Tags GIS File**.
4. Click **Browse**, select *solar_plant_inverters.geojson*, and click **Next**. The summary shows 20 features, one layer, and five attribute keys.
5. Pick **ModbusTCPIP**. Set **Channel Name Pattern** to *Inverters_Channel* and **Node Name Pattern** to *{Attr:device_id}*. Click **Next**.
6. Set **Tag Name Pattern** to *{Attr:device_id}*, **Tag Type** to *Analog*, **Description Attribute** to *inverter_model*, **Address Pattern** to *40001*. Click **Next**.
7. Review the preview grid. The first 50 rows show resolved names. Click **Next**.
8. Click **Start**. Wait for the counter to reach 20. Close the wizard.
9. Open **Edit Unified Namespace Tags**. The 20 new tags appear. Open **Edit Devices Points**. The 20 device points appear, linked to *Inverters_Channel*.

Edge Cases and Limitations

- **Multi-part geometry.** MultiPoint, MultiLineString, and MultiPolygon are reduced to a single centroid of the union. Topology is lost for these.
- **Coordinate reprojection.** All inputs are treated as WGS84. A non-WGS84 .prj is read but not acted on, so coordinates are used as given.
- **Null-island guard.** Features at exact (0,0) latitude and longitude are skipped by the display-generation pipeline. The device pipeline is unaffected and creates tags and points normally.
- **Hierarchical tag patterns.** Patterns such as *{Layer}/{Name}* are written as flat tag names in 10.1.5 without creating intermediate Asset Tree folders. Create the folder structure manually in **Asset Tree** if you need it.
- **Out-of-scope formats.** GeoPackage, File Geodatabase, WMS, WFS, and TopoJSON are not supported in 10.1.5.

Related Pages

- CSV files Tag Importer. The same pattern for CSV input.
- AutoCAD File Tag Importer. DXF and DWG input from engineering drawings.

- GIS/CAD Display Generator. Generate display pages from the same GIS or CAD files.

In this section...
