# system.tag.browseConfiguration

This function is used in **Python Scripting.**

## Description

Browses a folder path or UDT and returns Tag configuration information for each Tag within the specified path. This can be used to view event scripts, alarms, as well as any other configurable attribute on a Tag.

## Client Permission Restrictions

This scripting function has no Client Permission restrictions.

## Syntax

**system.tag.browseConfiguration(path, recursive)**

- Parameters

  String path - The path that will be browsed, typically to a folder or UDT instance. Leave blank to browse the root folder. A Tag Provider may be specified as follows: "[TagProvider]". If the Tag Provider is omitted, client scoped calls will be made against the project's default provider. Gateway scoped calls must include a Tag Provider. When browsing UDTs, specifying the UDT path will browse the Tags inside the UDT. To browse the UDT instance, specify the parent of the instance.

  Boolean recursive - If true, will recursively search for Tags in folders. Each folder will return a 'tags' property containing the nested TagConfigurations in another list.

- Returns

  TagConfiguration[] - A list of TagConfiguration objects. Attributes on the object may be read by calling tag.get(propertyObject). A list of attributes with configuration information can be obtained by calling getProperties(). Only attributes with non-default values will appear in the attribute list.

- Scope

  All

If called in the gateway scope, a tag provider must be specified.

### TagConfiguration Method Summary

| Method | Description |
| --- | --- |
| getAlarms() | Returns a list of AlarmDefinition objects. Calling getProperties() on the object will return a list of non-default alarm properties. |
| getFolderPath() | Returns the path to the folder the Tag resides in. |
| getFullPath() | Returns the Tag path. |
| getName() | Returns the name of the Tag. |
| getParameters() | Returns a BasicPropertySet of UDT parameters. Returns an empty list if called on a Folder. Parameters and their values may be accessed with '.property' and '.value' |
| getProperties() | Returns a Java Hashmap of properties that have non-default values. Individual values may be specified by calling tag.get(propertyObject). |
| getSubTags() | If called on a folder and the recursive parameter is True, returns a list of TagConfiguration objects pertaining to the nested nodes. Returns an empty list if called on a UDT instance or recursive is False. |
| getTagType() | Returns the type of the node. Possible values are:<br>• OPC<br>• UDT_INST<br>• Folder<br>• DB<br><br>The DB type is unique in that it is a parent type to several other types. If the TagConfiguration has a property named expressionType, then it can be examined to determine its type. If it does not contain an expressionType property, then it is a Memory tag. The other types are listed below:<br>• SQL_Query<br>• Expression<br>• DERIVED |

**Code Examples**

**Code Snippet - Print Attributes From All Tags**

```
"""
This example will browse the root of the Tag Provider named
"default".
For each tag found, prints the non-default properties.
"""

# Specify the folder to browse. Modify this line to filter on a
specific folder.
folderPath = "[default]"

# Start a non-recursive browse, and stores the configuration
data in a variable
tagConfig = system.tag.browseConfiguration(folderPath, False)

# Iterate through the configuration data
for tag in tagConfig:

 # For each tag, retrieve a list of non-default properties. We
check each iteration of tagConfig
 # because each tag can potentially have a different
configuration
 propList = tag.getProperties()

 # Iterate through the list of properties
 for prop in propList:

  # Print the property name and value.
  print "Property '%s' has a value of '%s'" % (prop,
tag.get(prop))
```

**Code Snippet - Show Tag Types**

```
# Start a non-recursive browse, and stores the configuration
data in a variable
tagConfig = system.tag.browseConfiguration('[default]', False)

# Iterate through the configuration data
for tag in tagConfig:

 # Get the type of tag.
 tagType = tag.getTagType()

 # The DB tag type has multiple subtypes. To determine its real
type, we need to examine it further
 if str(tagType) == 'DB':

  # Look at the properties of the DB tag, and iterate over them
  propList = tag.getProperties()
  for prop in propList:

   # Once we find the expressionType, we can examine this
property to determine the actual tag type
   if str(prop) == 'expressionType':
    tagType = tag.get(prop)
    # If we found the type, then we don't need to check the rest
of the properties
    break

  # If we didn't break out, then we didn't find expressionType.
Thus, it's a memory tag
  else:
   tagType = 'Memory'

 print '%s has a tag type of %s' % (tag.getName(), tagType)
```