LDAP Schema Plugins

See also: Writing Registry Plugins and LDAP Provisioning Plugin

Plugin Requirements

The name of the Plugin should match the format `FooLdapSchema`.

The Plugin must define a public array `$attributes` that defines the schema, and a public function `assemblePluginAttributes` that assembles the attributes to write.

Defining the Schema

Each plugin defines one or more LDAP schemas (which presumably already exist on your LDAP server). The schema must be unique across all plugins *and* the core schemas that COmanage already maintains. That is, any given schema can only be managed by exactly one source. This means, for example, that you can't define the `inetOrgPerson` schema or its attributes in a plugin, because COmanage already defines it.

Each key in the `$attributes` array is the name of the schema. Each value is an array with two entries:

- `objectclass`: An array with the following keys:
  - `required`: Whether or not this objectclass is required. Currently, all plugin-defined schemas must be required.
- `attributes`: An array where each key defines the name of the attribute, and each value is an array with the following keys:
  - `required`: Whether or not this attribute is required for this objectclass
  - `multiple`: Whether or not this attribute permits multiple values
  - `extendedtype`: If the value for this attribute will be pulled from a model that supports extended types, the type of data to be pulled:
    - `address_types`
    - `email_address_types`
    - `identifier_types`
    - `telephone_number_types`
  - `defaulttype`: If the value for this attribute will be pulled from a model that supports extended types, the default type (cm_co_extended_types:name)

Currently, only schemas related to person attributes are supported. ie: these attributes will only be assembled for CO Person transactions, not CO Group transactions.

Example Schema Definition

```
public $attributes = array(
  'testPerson' => array(
    'objectclass' => array(
      'required'    => true
    ),
    'attributes' => array(
      'testDescription' => array(
        'required'    => true,
        'multiple'    => false
      ),
      'erpNumber' => array(
        'required'     => false,
        'multiple'     => false,
        'extendedtype' => 'identifier_types',
        'defaulttype'  => 'employeenumber'
      )
    )
  )
);
```

assemblePluginAttributes()

When the LDAP Provisioner is assembling an LDAP record, it will call the plugin's assemblePluginAttributes to do the work necessary to build the attributes for the plugin's schema. This function is passed an array of configured attributes and an array of provisioning data, and is expected to return an array of attributes.

The return array is keyed on the attribute name, with the attribute value being one of

- A simple string, for single valued attributes
- An array of strings, for multi valued attributes
- An empty array, if there is no value for the attribute (do *not* set the attribute to null), even if the attribute would otherwise be single valued

Sample Attribute Assembly

```
public function assemblePluginAttributes($configuredAttributes, $provisioningData) {
  $attrs = array();

  foreach($configuredAttributes as $attr => $cfg) {
    switch($attr) {
      case 'erpNumber':
        $attrs[$attr] = array();
        foreach($provisioningData['Identifier'] as $m) {
          if(isset($m['type'])
             && $m['type'] == $cfg['defaulttype']
             && $m['status'] == StatusEnum::Active) {
            $attrs[$attr] = $m['identifier'];
            break;
          }
        }
        break;
      case 'testDescription':
        $attrs[$attr] = "This is a test description";
        break;
      // else we don't know what this attribute is
    }
  }

  return $attrs;
}
```

Only Return Configured Attributes

`assemblePluginAttributes()` is expected to return an array whose keys correspond to the keys of `$configuredAttributes`. Any keys returned that are not present in `$configuredAttributes` will be ignored by the LDAP Provisioner when constructing the LDAP record.