

# MappedAttributeDefinition

The Mapped attribute definition performs a many-to-many mapping from source attributes values according to a series of mapping statements. If an input value matches no mapping rule, then a default can be provided.

## Schema Name and Location

This `xsi:type` is defined by the `urn:mace:shibboleth:2.0:resolver` namespace<sup>3.3</sup>, the schema for which can be located at <http://shibboleth.net/schema/idp/shibboleth-attribute-resolver.xsd>

Prior to V3.3 supplied plugins were defined by a schema type in the `urn:mace:shibboleth:2.0:resolver:ad` namespace, the schema for which is located at <http://shibboleth.net/schema/idp/shibboleth-attribute-resolver-ad.xsd>. This is still supported, but every element or type in the old namespace has an equivalently named (but not necessarily identical) version in the `urn:mace:shibboleth:2.0:resolver` namespace. The use of the `urn:mace:shibboleth:2.0:resolver` namespace also allows a relaxation of the ordering requirements of child elements to reduce strictness.

## Attributes

Any of the [common attributes](#) can be specified.

## Child Elements

Any of the [common child elements](#) can be specified. In addition the following elements are supported:

Name	Cardinality	Description
<code>&lt;DefaultValue&gt;</code>	0 or 1	Describes the action to be taken if <i>any</i> input value does not match one of the <code>&lt;ValueMap&gt;</code> <code>SourceValue</code> elements that follow. If the <code>passThru</code> attribute is set to "true", then the input value is passed through untouched, otherwise the contents of the element are used as the default value.
<code>&lt;ValueMap&gt;</code>	1 or more	See below.

## <ValueMap>

The `<ValueMap>` element supports two child elements:

Name	Cardinality	Description
<code>&lt;ReturnValue&gt;</code>	1	The content is used as the output value if any of the source values match. Replacement rules from <a href="#">java.util.regex.Matcher.replaceAll(java.lang.String)</a> apply.
<code>&lt;SourceValue&gt;</code>	1 or more	<p>The content is matched against each input value and if it matches, then the mapped <code>&lt;ReturnValue&gt;</code> is output. The mapping process depends on the value of the <code>partialMatch</code> attribute, which defaults to false.</p> <p>If set to true, then the source value is checked for containment within the input value, and if so, the return value replaces the original.</p> <p>If set to false, then the source value is a <a href="#">Java Regular Expression</a> applied to the input value, and if a match, then the return value is applied as a replacement, with all matches replaced by the return value.</p> <p>The <code>ignoreCase</code> attribute (default false) allows case insensitive comparisons to be made. This is only valid for regular expression matching (i.e., if <code>partialMatch</code> is false).</p>

## Example

```
<AttributeDefinition id="mapped" xsi:type="Mapped">
  <InputAttributeDefinition ref="uid" />
  <DefaultValue passThru="true"/>
  <ValueMap>
    <ReturnValue>return1</ReturnValue>
    <SourceValue>sou.+rc.+e1</SourceValue>
    <SourceValue partialMatch="true">fred</SourceValue>
    <SourceValue ignoreCase="true">Ignore.+Case.+When.+Comparing</SourceValue>
  </ValueMap>
  <ValueMap>
    <ReturnValue>return1</ReturnValue>
    <SourceValue>source2</SourceValue>
  </ValueMap>
  <ValueMap>
    <ReturnValue>some_string_to_add_before_value:$1</ReturnValue>
    <SourceValue>(.)</SourceValue>
  </ValueMap>
  <AttributeEncoder xsi:type="SAML2String" name="https://example.org/example/name" friendlyName="Mapped"
  encodeType="false" />
</AttributeDefinition>
```