API Profiles

An API Profile enables the creation of a data policy for a particular set of API Resources, generally in support of a specific usage scenario (such as for Nutrition or Special Education specialty applications).

The policy is expressed as a set of rules for explicit inclusion or exclusion of properties, references, collections, and/or collection items (based on Type or Ed-Fi Descriptor values) at all levels of a Resource. The Profile can then be assigned to API consumer applications in the Security Configuration Tool to ensure that all requests made by the consumer for Resources covered by the Profile's data policy use the Profile-specific content types, and thus are appropriately constrained.

Developers' Guide Contents

Find out more about how to develop platforms based on the Ed-Fi ODS / API v3.1:

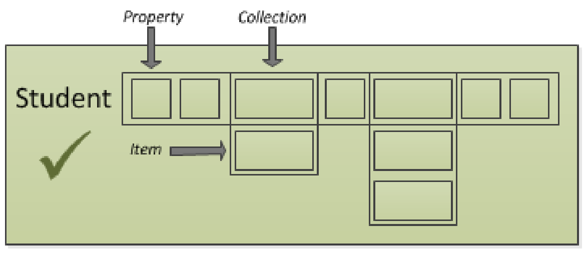


Figure 1. Depiction of the inclusion of all properties

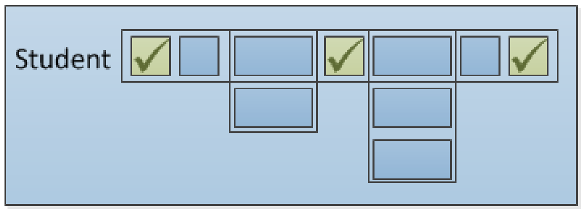


Figure 2. Depiction of the inclusion of specific properties

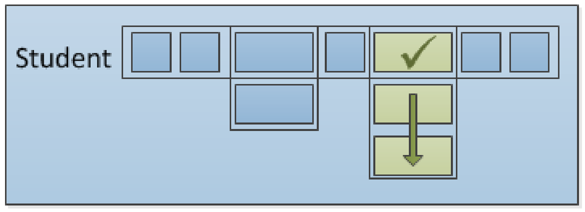


Figure 3. Depiction of the inclusion of a specific collection

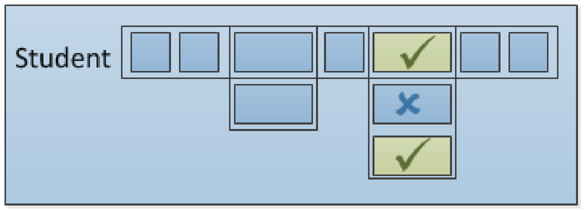


Figure 4. Depiction of the inclusion or exclusion of specific collection items

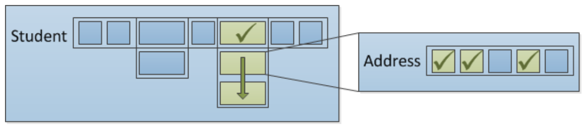
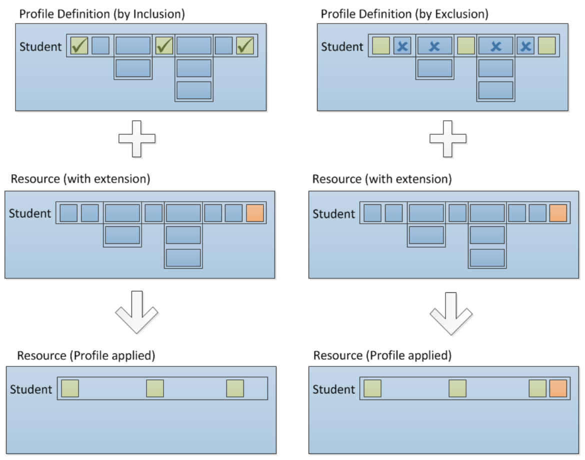


Figure 5. Depiction of the inclusion of specific properties on a child collection's items

Include-Only vs. Exclude-Only Strategies

Platform implementers can choose to configure access by inclusion or exclusion, but it's useful to understand the implications of using IncludeOnly vs. ExcludeOnly member selection modes when defining Profiles.

If the IncludeOnly value is used exclusively, a very rigid Profile definition will result. As new elements are added to the data model (either through an implementer extending the data model or when upgrading to a new version of the Ed-Fi Data Standard), none of these added data elements will be included. However, if the Profile is defined using ExcludeOnly then these other elements will be automatically included, resulting in a more flexible definition that will not necessarily require adjustments over time. The implications of these two approaches is depicted in the diagram below:


Profile Definition (by Inclusion)
Profile Definition (by Exclusion)
Student
Resource (with extension)
Resource (Profile applied)
Figure 6. Depiction of the implications on Profile flexibility using Inclusion vs. Exclusion

Profile Definition

The Profile Definition is expressed in XML in terms of the Resource model's members (not to be confused with the JSON representation), and is used by the code generation process in Visual Studio to create all the API artifacts necessary.

An XML schema is included in the Ed-Fi-ODS repository at Ed-Fi-ODS\Application\EdFi.Ods.CodeGen\Models\ProfileMetadata\Ed-Fi-ODS-API-Profiles.xsd. Additionally, a set of sample profiles can be found at Ed-Fi-ODS\Application\EdFi.Ods.Api.Models.SampleProfiles\Profiles.xml, and full set of test profile definitions can be found at Ed-Fi-ODS\Application\EdFi.Ods.Api.Models.TestProfiles\Profiles.xml. These definitions contain many different variations of profiles applied to API resources and serve as useful examples for developers.

A Profile Definition can consist of multiple Resources (e.g., School and Student):

```
<!-- Multiple resources -->
<Profile name="Test-Profile-Student-and-School-Include-All">
        <Resource name="School">
                <ReadContentType memberSelection="IncludeAll" />
                <WriteContentType memberSelection="IncludeAll" />
        </Resource>
        <Resource name="Student">
                <ReadContentType memberSelection="IncludeAll" />
                <WriteContentType memberSelection="IncludeAll" />
        </Resource>
</Profile>
```

Resources can be readable or writable only:

```
<!-- Readable Only Profile-->
<Profile name="Test-Profile-Resource-ReadOnly">
        <Resource name="School">
                <ReadContentType memberSelection="IncludeAll" />
        </Resource>
</Profile>

<!-- Writable Only Profile-->
<Profile name="Test-Profile-Resource-WriteOnly">
        <Resource name="School">
                <WriteContentType memberSelection="IncludeAll" />
        </Resource>
</Profile>
```

Resource members can be explicitly included based on the member selection:

```
<!-- Resource-level IncludeOnly -->
<Profile name="Test-Profile-Resource-IncludeOnly">
        <Resource name="School">
                <ReadContentType memberSelection="IncludeOnly">
                        <Property name="NameOfInstitution"
/>                               <!-- Inherited property -->
                        <Property name="OperationalStatusType"
/>                           <!-- Inherited Type property -->
                        <Property name="
CharterApprovalSchoolYearTypeReference" />          <!-- Property -->
                        <Property name="SchoolType"
/>                                      <!-- Type property -->
                        <Property name="
AdministrativeFundingControlDescriptor" />          <!-- Descriptor
property -->
                        <Collection name="EducationOrganizationAddresses"
memberSelection="IncludeAll"/> <!-- Inherited Collection -->
                        <Collection name="SchoolCategories"
memberSelection="IncludeAll" /> <!-- Collection -->
                </ReadContentType>
                <WriteContentType memberSelection="IncludeOnly">
                        <Property name="ShortNameOfInstitution"
/>                          <!-- Inherited property -->
                        <Property name="OperationalStatusType"
/>                           <!-- Inherited Type property -->
                        <Property name="WebSite"
/>                                         <!-- Property -->
                        <Property name="CharterStatusType"
/>                               <!-- Type property -->
                        <Property name="
AdministrativeFundingControlDescriptor" />          <!-- Descriptor
property -->
                        <Collection name="
EducationOrganizationInternationalAddresses" memberSelection="IncludeAll"
/> <!-- Inherited Collection -->
                        <Collection name="SchoolGradeLevels"
memberSelection="IncludeAll" /> <!-- Collection -->
                </WriteContentType>
        </Resource>
</Profile>
```

Resource members can be explicitly excluded based on the member selection:

```
<Profile name="Test-Profile-Resource-ExcludeOnly">
        <Resource name="School">
                <ReadContentType memberSelection="ExcludeOnly">
                        <Property name="NameOfInstitution"
/>                               <!-- Inherited property -->
                        <Property name="OperationalStatusType"
/>                           <!-- Inherited Type property -->
                        <Property name="
CharterApprovalSchoolYearTypeReference" />          <!-- Property -->
                        <Property name="SchoolType"
/>                                      <!-- Type property -->
                        <Property name="
AdministrativeFundingControlDescriptor" />          <!-- Descriptor
property -->
                        <Collection name="EducationOrganizationAddresses"
memberSelection="IncludeAll" /> <!-- Inherited Collection -->
                        <Collection name="SchoolCategories"
memberSelection="IncludeAll" /> <!-- Collection -->
                </ReadContentType>
                <WriteContentType memberSelection="ExcludeOnly">
                        <Property name="ShortNameOfInstitution"
/>                          <!-- Inherited property -->
                        <Property name="OperationalStatusType"
/>                           <!-- Inherited Type property -->
                        <Property name="WebSite"
/>                                         <!-- Property -->
                        <Property name="CharterStatusType"
/>                               <!-- Type property -->
                        <Property name="
AdministrativeFundingControlDescriptor" />          <!-- Descriptor
property -->
                        <Collection name="
EducationOrganizationInternationalAddresses" memberSelection="IncludeAll"
/> <!-- Inherited Collection -->
                        <Collection name="SchoolGradeLevels"
memberSelection="IncludeAll" /> <!-- Collection -->
                </WriteContentType>
        </Resource>
</Profile>
```

The same inclusion/exclusion rules apply to child collections (e.g., the School's addresses):

```
<!-- Child collection IncludeOnly/ExcludeOnly profiles -->
<Profile name="Test-Profile-Resource-BaseClass-Child-Collection-
IncludeOnly">
        <Resource name="School">
                <ReadContentType memberSelection="IncludeOnly">
                        <Collection name="EducationOrganizationAddresses"
memberSelection="IncludeOnly">
                                <Property name="City" />
                                <Property name="StateAbbreviationType" />
                                <Property name="PostalCode" />
                        </Collection>
                </ReadContentType>
                <WriteContentType memberSelection="IncludeOnly">
                        <Collection name="EducationOrganizationAddresses"
memberSelection="IncludeOnly">
                                <Property name="Latitude" />
                                <Property name="Longitude" />
                        </Collection>
                </WriteContentType>
        </Resource>
</Profile>
```

The data policy can contain filters on child collection items (e.g., only include Physical and Shipping addresses):

```
<!-- Child collection filtering on types and descriptors -->
<Profile name="Test-Profile-Resource-Child-Collection-Filtered-To-
IncludeOnly-Specific-Types-and-Descriptors">
        <Resource name="School">
                <ReadContentType memberSelection="IncludeOnly">
                        <Collection name="EducationOrganizationAddresses"
memberSelection="IncludeOnly">
                                <Filter propertyName="AddressType"
filterMode="IncludeOnly">
                                        <Value>Physical</Value>
                                        <Value>Shipping</Value>
                                </Filter>
                                <Property name="StreetNumberName" />
                                <Property name="City" />
                                <Property name="StateAbbreviationType" />
                        </Collection>
                </ReadContentType>
        </Resource>
</Profile>
```

In the example above, `GET` requests will only return Physical and Shipping addresses. If also applied to the `WriteContentType`, the caller will receive an error response if they attempt to write anything other than Physical or Shipping addresses.

Adding Profiles to the Ed-Fi ODS / API Solution

Refer to How To: Add Profiles to the Ed-Fi ODS / API Solution for details.