DigitalMeasures

Web Services Platform Version 4

Updated 2016-10-08

Concepts and Getting Started

Concepts and Getting Started

Introduction

Digital Measures supports interoperability and automation via web services, including visibility into your data source's configuration, data import and export, user setup and management, and security role assignment. Each service resource supports a similar grammar and style, so the knowledge gained implementing a web service client for one resource transfers easily to others.

Web Services

Web services allow computer systems to collaborate across networks, operating systems, and programming languages. Data is generally transferred between systems as XML, but some services may respond with binary data formats when appropriate. For more information on web services, please see the following resource:

- http://en.wikipedia.org/wiki/Web_Service
  Wikipedia article describing web services in general, with focus on W3C web services.

REST-style Services

Digital Measures uses document-based, REST-style web service endpoints that are easy to understand and test. You may be familiar with SOAP web services that can require platform support, complex toolkits, and code generation. Our resources are straightforward and easy, focusing on simple technologies like HTTP, XML, and RelaxNG. Resource data is accessible via simple HTTP GET commands to human-readable URLs, allowing convenient testing within your web browser. Our writeable web services respond to HTTP POST, PUT, and DELETE methods to create, update, and delete resources, and most provide a friendly validation resource to test your request before making changes.

For more information on REST web service architecture, please see the following resources:

- http://en.wikipedia.org/wiki/Representational_State_Transfer
  Wikipedia article describing the underlying theory of Representational State Transfer
- http://rest.blueoxen.net/cgi-bin/wiki.pl?FrontPage
  RESTWiki site with in-depth discussion of REST principles, plus links to external resources
- http://www.oreilly.com/catalog/9780596529260/
  Richardson, Leonard and Sam Ruby. RESTful Web Services. Sebastopol, CA: O'Reilly Media, Inc., 2007.
  Excellent coverage of consuming and designing REST web services

Additional Documentation

- http://www.w3.org/TR/xlink/
  XLink is used by many resources to indicate URLs for related data.
- http://www.oasis-open.org/committees/relax-ng/tutorial.html
  RelaxNG is an XML schema language that formally defines the expected structure of a XML document, and is considered more flexible and easy to use than W3C XML Schema. RelaxNG schemas are available for all Digital Measures resources.

Documentation Style

Anywhere a resource URL expects substitution of a value from a Service Grammar element, the following convention will be used: `{GrammarValue}`

No correctly formed URL may contain either `{` or `}` characters; URL encoding would substitute character codes for both.

Metadata

Some web service resources provide metadata in a separate XML Namespace from that of the main document representation. Metadata elements and attributes are provided for developer reference and debugging purposes only. *Note: The presence, name and values of metadata may change without notice between web service releases. Web service clients should not store these values, nor depend on their presence.*

Case Sensitivity

All URLs, element names, and enumerated values in RelaxNG schemas are case-sensitive.

Resource Testing

A beta environment is available for web service client development and testing purposes. To use the beta environment, change all Resource URLs from:

```
www.digitalmeasures.com/login/service/v4
```

to:

```
beta.digitalmeasures.com/login/service/v4.
```

Note: the beta database is refreshed from production each Saturday at 5:00 a.m. US/Central; any data that is changed in beta will be reverted when a refresh occurs.

Resource Misuse

Digital Measures strives to build robust and tolerant software, however we cannot always anticipate how other developers may use our web services. Web services usage in excess of 10,000 requests per day per client is concerning, and may result in Digital Measures reaching out to pursue more efficient means of meeting the given need. It is rare, however, that clients reach this level of activity since batching of requests and caching of data can be used to diminish the number of calls required. It is also important to note that web services requests to Digital Measures for the purposes of integrating with another data source on campus should be made outside of business hours (7 am - 7 pm CT). If we determine that your organization's web service usage is causing a detriment to the system, we may temporarily disable your web service accounts until we are able to contact your Digital Measures Administrator.

Resources which work with potentially large data sets have explicitly limits defined. If you encounter the error `"Request too large: 21829 records. Please narrow the scope of your request and try again."` you may need to make multiple smaller requests. For example, if you encounter the error working with all Scheduled Teaching data for the past five years for everyone in your organization, you could:

- Make one request per year for everyone on your campus
- Make one request for the past five years per college or department

Please see specific resources for exact limits and workarounds.

Updated Documentation

The most recent version of this documentation is always available from the Documentation Resource.

Service Grammar

Digital Measures' web services share common grammar when constructing resource URLs.

- **{Date}**
  An ISO 8601 ("yyyy-MM-dd") formatted date. For example, "2007-10-13".
- **{DateQuery}**
  A URL query string that specifies a combination of "start" and/or "end" dates when calling the Data Query Resource. For example, "start=2007-10-13" includes records that occur after the specified date, while "end=2007-10-13" includes records that occur before. Specifying both, as "start=2007-10-13&end=2007-10-13", returns only records that occur on the specified date.
- **{EntityKey}**
  An identifier that represents a data collection entity, which is represented to end users as a single data collection page. Entity identifiers correspond with XML elements returned by the Data Query Resource. Available **{EntityKey}** values may be retrieved from the Schema Entity List Resource.
- **{EntityKeys}**
  A delimited collection of **{EntityKey}** strings. Entities are delimited by comma characters. Available **{EntityKey}** values may be retrieved from the Schema Entity List Resource.
- **{fullTextSearch}**
  A URL query string that specifies a search value to find matching records. For example, "fullTextSearch=%22mobile+phones%22" matches records containing the phrase "mobile phones." Searches analyze the entire record contents, including file attachments, and are not able to target specific fields.
- **{IndexEntryKey}**
  A value on an **{IndexKey}** axis that may be used to limit data retrieved. For example, the college index may allow values such as "Business", "Engineering", etc. Note that a single user may have multiple values for an index; i.e. a faculty member that belongs to both the Department of "Mathematics" and the Department of "Statistics". Available **{IndexEntryKey}** values may be retrieved from the Schema Index List Resource.
- **{IndexKey}**
  An axis that may be used to limit data retrieved. Common indexes include "COLLEGE", "DEPARTMENT", etc. Available **{IndexKey}** values may be retrieved from the Schema Index List Resource.
- **{IndexKeyEntryKeys}**
  A delimited collection of **{IndexKey}** and **{IndexEntryKey}** pairs. **{IndexKeys}** and Entry Keys are separated by colon characters, and multiple Key/Value pairs are delimited by commas characters. For example, to specify everyone from the Business College as well as everyone from the Mathematics Department, the **{IndexKeyEntryKeys}** string of "COLLEGE:Business,DEPARTMENT:Mathematics" would be used. Available **{IndexKey}** and **{IndexEntryKey}** values may be retrieved from the Schema Index List Resource.
- **{RoleKey}**
  Unique identifier for a security role. A security Role specifies a set of capabilities that may be applied to one or more users. Available **{RoleKey}** values may be retrieved from the Role List Resource.
- **{SchemaKey}**
  Unique identifier for a data source. Available **{SchemaKey}** values may be retrieved from the Schema List Resource.
- **{Username}**
  Unique identifier for a user. This is the identifier users enter to log into Digital Measures. This is often the user's email address without the domain name or a unique identifier assigned by your organization. Available **{Username}** values may be retrieved from the User List Resource.
- **{UserIdentifierType}**
  Unique identifier for a user, configured by Digital Measures for your organization. A user may have a unique user identifier value for a user identifier type, allowing you to integrate your user identities into our

web services easily. Available **{UserIdentifierType}** values and associated user identifier values may be retrieved from the User List Resource.

- **{UserSchemaKey}**
  Unique identifier for a user's relationship to a data source. Available **{UserSchemaKey}** values for a User may be retrieved from the User Schema List Resource.

Getting Started

First Steps

1. Ask your Digital Measures administrator to request a Service Account with appropriate privileges.
2. Use your web browser to explore the read-only web services. Start by downloading the most recent version of this document from the Documentation Resource. Next, view the Schema List Resource to find your Schema Key(s), which is used by many other service resources.
3. Consider installing the RESTClient Firefox Extension (a simple interface for testing readable and writeable REST web services), available at https://addons.mozilla.org/en-US/firefox/addon/restclient/.
4. Find appropriate HTTP Connection sample code for your development platform.
5. If you will be creating a web application based on web service data, choose an appropriate technology to transform XML data into HTML for your development platform. XSLT is one common option available on many platforms; using XPath to navigate a Document Object Model is another.
6. Read the documentation for the resources you plan on consuming. The following sample use cases provide a starting point for a variety of tasks.
7. If you use resources that modify data, please submit your data to the corresponding Validate resource prior to submitting to the primary resource. Validation resources provide detailed error messages without modifying data. Primary resources fail immediately on detecting any error, and revert any changes that had previously been processed.
8. Build and test your application against the beta environment; once tested and debugged, switch your resource URLs to point to production.

Example: Faculty Directory Web Site

1. Retrieve a list of colleges and/or departments as stored by Digital Measures from the Schema Index Resource.
   `GET /login/service/v4/SchemaIndex/{SchemaKey}`
2. Transform the returned index/entry lists into HTML, and include a link to a user list page for each College or Department, including the **`{IndexKey}`** and **`{EntryKey}`** in the URL or query string.
3. Retrieve a list of users for a college or department from the User List Resource.
   `GET /login/service/v4/User/{SchemaKey}/{IndexKey}:{EntryKey}`
4. Transform the returned user list into HTML, and include a link to a user detail page for each User, including the **`{Username}`** in the URL or query string.
5. Retrieve user data for an individual user from the Schema Data Query Resource, requesting data from as many data collection screens as needed. **`{EntityKey}`** values for data collection screens may be retrieved from the Schema Entity Resource, and cached or hard-coded.
   `GET /login/service/v4/SchemaData/{SchemaKey}/USERNAME:{Username}/{EntityKeys}`
6. Transform the returned user data into HTML, optionally integrating data from other university information systems.
7. Optional: Provide an HTML form to search for users by partial first and/or last name. Users can be searched via the User List Resource.
   `GET /login/service/v4/User/{SchemaKey}?firstName=F&lastName=Flintstone`

Example: Centralized Account Creation

1. For reference, retrieve a list of users currently stored in the system from the User List Resource.
   `GET /login/service/v4/User/`
2. Create a new user by sending a correctly formatted XML request for user creation to the User Create Resource.
   `POST /login/service/v4/User/`

```
<User username="{Username}">
    <FirstName>{FirstName}</FirstName>
    <MiddleName>{MiddleName}</MiddleName>
    <LastName>{LastName}</LastName>
    <Email>{EmailAddress}</Email>
    <LocalAuthentication>{Password}</LocalAuthentication>
</User>
```

3. Grant the new user access to a Schema by sending a correctly formatted XML request for user creation to the User Schema Create Resource. *Note: The exact XML required will vary based on your data collection screen configuration; consult the* User Schema Create RelaxNG Resource *for full details.*
   `POST /login/service/v4/UserSchema/USERNAME:{Username}`

```
<{SchemaKey}>
    <ADMIN>
        <AC_YEAR>{AcademicYear}</AC_YEAR>
        <ADMIN_DEP>
            <DEP>{Department}</DEP>
        </ADMIN_DEP>
    </ADMIN>
</{SchemaKey}>
```

4. Grant the new user access to a security role by sending a correctly formatted XML request for user creation to the User Role Create Resource. Note: Not all security roles require a Permission entity; consult the Role List Resource or the User Role Create RelaxNG Resource for full details.
   `POST /login/service/v4/UserRole/USERNAME:{Username}`

```
<{RoleKey}>
    <Permission permissionKey="{PermissionKey}"/>
</{RoleKey}>
```

Example: Automated Data Import

1. For reference, or when updating data, retrieve existing data using the Schema Data Query Resource.
   `GET /login/service/v4/SchemaData/{SchemaKey}`
2. Transform your existing data into the format specified by the Schema Data RelaxNG Resource, which describes the same format returned by the Schema Data Query Resource.
3. Send the data for import to the Schema Data Validate Resource.
   `POST /login/service/v4/SchemaData:validate/{SchemaKey}`

```
<Data>
    <Record username="{Username1}">
        <PCI>
            <OPHONE1>{AreaCode1}</OPHONE1>
            <OPHONE2>{Exchange1}</OPHONE2>
            <OPHONE3>{Line1}</OPHONE3>
        </PCI>
    </Record>
    <Record username="{Username2}">
        <PCI>
            <OPHONE1>{AreaCode2}</OPHONE1>
            <OPHONE2>{Exchange2}</OPHONE2>
            <OPHONE3>{Line2}</OPHONE3>
        </PCI>
    </Record>
</Data>
```

4. Resolve any validation errors reported, by correcting invalid values, or by asking your Digital Measures Administrator to request screen revisions necessary to accommodate your data.
5. Send the data for import to the Schema Data Import Resource.
   `POST /login/service/v4/SchemaData/{SchemaKey}`

```
<Data>
    <Record username="{Username1}">
        <PCI>
            <OPHONE1>{AreaCode1}</OPHONE1>
            <OPHONE2>{Exchange1}</OPHONE2>
            <OPHONE3>{Line1}</OPHONE3>
        </PCI>
    </Record>
    <Record username="{Username2}">
        <PCI>
            <OPHONE1>{AreaCode2}</OPHONE1>
            <OPHONE2>{Exchange2}</OPHONE2>
            <OPHONE3>{Line2}</OPHONE3>
        </PCI>
    </Record>
</Data>
```

Example: Security Role Management

1. Retrieve a list of users currently stored in the system from the User List Resource.
   `GET /login/service/v4/User/`
2. Transform the returned user list into HTML, and include a link to a user detail page for each user, including the **{Username}** in the URL or query string.
3. Retrieve existing security role assignments for a user from the User Role List Resource.
   `GET /login/service/v4/UserRole/USERNAME:{Username}`
4. Transform the returned user roles into HTML, and include a link to a user role editing for each College or Department, including the **{Username}** and **{RoleKey}** in the URL or query string.
5. Retrieve existing security role assignment details for a user from the User Role Item Resource.
   `GET /login/service/v4/UserRole/USERNAME:{Username}/{RoleKey}`
6. Retrieve the possible permissions for the security role from the Role Item Resource.
   `GET /login/service/v4/Role/{RoleKey}`
7. Transform the returned user role details into an HTML form, and include list of possible permissions for the security role for editing.
8. Update the existing security role assignments with the selected permissions using the User Role Update Resource.
   `PUT /login/service/v4/UserRole/USERNAME:{Username}/{RoleKey}`
9. To create a new association between a user and a security role, call the User Role Create Resource.
10. To delete a user's existing association with a security role, call the User Role Delete Resource.

Accounts and Privileges

A special service account is required for web service access; normal user accounts will not work. If you have not been provided service account credentials, please ask your Digital Measures Administrator to request that an account be created. Please indicate which service resources privileges you wish to use. Privileges can always be increased later, but should your service account ever be compromised, minimal privileges may decrease potential damage.

Service Account Privileges

- **Schema Resource:** Read Access - *Metadata about your data collection screen configurations*
- **Data Resources:**
  - Read Access - *Retrieve data already stored by Digital Measures*
  - Write Access - *Import data into Digital Measures from another source, or update data already stored*
  - *Delete Access - Delete records stored by Digital Measures*
- **User Resources:**
  - Read Access - *View existing user accounts*
  - Write Access - *Create or modify user accounts and data collection access*
- **Role Resources:**
  - Read Access - *View details of security role configuration and user role assignments*
  - Write Access - *Add or remove user role assignments*

A couple of notes regarding these resources:

- If you choose to include Read or Write access to the User Resource, we recommend also including Read or Write for the Role Resource, as accounts cannot be created fully without the ability to assign security roles.
- While Schema Resources is listed as an "option" above, it will automatically be included with each service account setup.

Existing Service Accounts

Existing service accounts from before Version 3 of the web services (i.e. those calling `/report/service/query.do` and `/report/service/queryCriteria.do` URLs) will only have:

- Schema Resources: Read Access
- Data Resources: Read Access

Existing service accounts using Version 3 web services will only have:

- Schema Resources: Read Access
- Data Resources: Read Access
- User Resources: Read Access

You may request additional privileges as needed; please ask your Digital Measures Administrator to request specific privileges to be added to your existing account.

Connecting

All service endpoints use HTTP Basic Authentication and **require** an HTTPS connection to protect both your data and account credentials in-transit. All modern development platforms provide a library for HTTP/HTTPS data retrieval. All service endpoint URLs are relative to Digital Measure's server at https://www.digitalmeasures.com/ for the Production environment, and https://beta.digitalmeasures.com/ for the Beta environment.

As XML is a fairly verbose format, GZip data compression may be used to make large data transfers faster. If your HTTP client can accept compressed data, please send an `Accept-Encoding` HTTP header to enable response compression. Also, all writeable service endpoints support receiving GZip compressed data; please specify a `Content-Encoding` HTTP header when request compression is in use.

If your platform does not have sample code in Appendix A, and you implement a working solution, please consider sending a code snippet to Digital Measures for inclusion in future revisions of this documentation.

Documentation Resources

Documentation Resources provide read-only information regarding Digital Measures Web Services. Currently, only a single resource is supported; future releases may include additional resources.

Web Services Documentation Resource

```
GET /login/service/v4/Documentation
```

Download the most recent version of this documentation.

Schema Resources

A Schema is a representation of the set of data collection screens used by your organization. Your organization may have a single schema if you are working with Digital Measures at an institutional level. Your organization may have multiple schemas if you are using more than one data collection solution, or several units within your organization are working with Digital Measures separately.

Schema resources provides read-only metadata about your data collection screen configurations, including the **{SchemaKey}** unique identifier that is used as part of many other resource requests, and links to several other resources. This resource is primarily informational for the web service client developer.

Schema List Resource

```
GET /login/service/v4/Schema
```

Specifies the available Schemas and **{SchemaKey}** values, along with XLink references to additional resources.

- **/Schemas/Schema/@schemaKey**
  String unique identifier for a Schema, referenced as **{SchemaKey}** from other resources.
- **/Schemas/Schema/@text**
  Human-readable instrument name, appropriate for display. This name is not guaranteed to be unique, as it is dependent on your organization's requested configuration.

Schema List RelaxNG

```
GET /login/service/v4/Schema:list-relaxng
```

Formally defines the document available via the Schema List Resource.

Schema Entity Resources

A Schema Entity corresponds to a single data collection screen. Schema Entity resources provides read-only metadata about your data collection screens, including the **{EntityKey}** unique identifier that is used as part of many other resource requests, and links to several other resources.

Schema Entity List Resource

`GET /login/service/v4/SchemaEntity/{SchemaKey}`

Specifies the available data Entities and **{EntityKey}** values for a Schema. Entities are grouped into `Views` that indicate the visibility of Entities to users. All users can see all Entities in the `"Common"` View; other Views may exist if your institution has customized their Schema for specific colleges or departments.

- **`/Entities/View/@text`**
  Human readable View name.
- **`/Entities/View/Entity/@entityKey`**
  String unique identifier for an Entity, referenced as **{EntityKey}** from other resources.
- **`/Entities/View/Entity/@text`**
  Human-readable Entity name, appropriate for display. This corresponds with the title of a data collection page within your Schema.

Schema Entity List RelaxNG

`GET /login/service/v4/SchemaEntity:list-relaxng/{SchemaKey}`

Formally defines the document available via the Schema Entity List Resource.

Schema Index Resources

Schema Indexes are used to search records based on pre-defined criteria. Indexed data is primarily stored in one or more questions within a Schema. Default indexes include college, department, and username.

If the default indexes are not sufficient, please ask your Digital Measures Administrator to request that a new index be created. An index may be created on any data collection question, but are most useful when the question pertains to the user, not to an individual data record within the system. For example, indexing a question regarding the user's Faculty/Staff status would be appropriate, while indexing a question regarding the date of a presentation would generally not be appropriate.

Advanced indexes may use data from multiple questions within a single data collection screen, or may use conditional logic to reformat question data. Generally, if the index criteria can be expressed using basic XPath syntax, Digital Measures can construct an index for your use.

Note: Indexes are also used by the Digital Measures web application for reporting criteria. Any index created for web service usage will be available when running custom or ad-hoc reports.

Schema Index List Resource

```
GET /login/service/v4/SchemaIndex/{SchemaKey}
          [/{IndexKeyEntryKeys}]
```

Specifies the available Indexes as **{IndexKey}** and **{IndexEntryKey}** values, along with XLink references to additional resources.

If an optional set of **{IndexKeyEntryKeys}** is specified, the returned Index Entry lists will be restricted by the supplied values. This feature could be used to retrieve lists of "DEPARTMENT" and "USER" Entries found for members of "COLLEGE:Business".

- **/Indexes/Index/@indexKey**
  String unique identifier for an Index, referenced as **{IndexKey}** from other resources.
- **/Indexes/Index/@text**
  Human-readable Index name, appropriate for display.
- **/Indexes/Index/IndexEntry/@entryKey**
  String unique identifier for an Index Entry value, referenced as **{IndexEntryKey}** from other resources.
- **/Indexes/Index/IndexEntry/@text**
  Human-readable Index Entry name, appropriate for display.

Example: All Indexes
All indexes and entries for the university:

```
GET /login/service/v4/SchemaIndex/INDIVIDUAL-ACTIVITIES-University
```

Example: Subset for Index Entry
Indexes and entries for the College of Business:

```
GET /login/service/v4/SchemaIndex/INDIVIDUAL-ACTIVITIES-University
          /COLLEGE:Business
```

Example: Subset for Index Entries
Indexes and entries for the College of Business and the College of Engineering:

```
GET /login/service/v4/SchemaIndex/INDIVIDUAL-ACTIVITIES-University
          /COLLEGE:Business,COLLEGE:Engineering
```

Schema Index List RelaxNG

```
GET /login/service/v4/SchemaIndex:list-relaxng/{SchemaKey}
```

Formally defines the document available via the Schema Index List Resource.

Schema Data Resources

Schema Data represents data stored by Digital Measures for one or more users. Schema-level data access is designed for batch or bulk operations, such as querying across all users in a department, or importing new data records for an entire university. For access to a single user's data, please consider the User Data Resources instead.

Schema Data Query Resource

```
GET /login/service/v4/SchemaData/{SchemaKey}/{IndexKeyEntryKeys}
          [/{EntityKeys}]
          [?{DateQuery}&{fullTextSearch}]
GET /login/service/v4/SchemaData/{SchemaKey}/{EntityKeys}
          [?{DateQuery}&{fullTextSearch}]
```

Retrieve stored data for one or more users. The data may be limited by pairs of **{IndexKeys}** and **{IndexEntryKey}** values.

Specifying an optional set of **{EntityKey}** values limits the types of data returned to specific data collection screens.

The optional **{DateQuery}** argument can limit records to a specified date range.

The optional **{fullTextSearch}** argument can limit records to those that match a search value.

- **/Data/Record**
  Contains all data stored on behalf of a user in the requested Schema and Entities.
- **/Data/Record/dmd:IndexEntry**
  Metadata describing the **{IndexKeys}** and **{IndexEntryKey}** values that may be used to access the data contained by the record.
- **/Data/Record/{EntityKey}/@dmd:lastModified**
  Last date and time the data contained was modified. *Note: This date may not be the last time the record was edited by the owner of the data; Manage Data functionality or Schema design changes may also have affected the record's content and last modified dates.*

This resource is restricted to 20,000 **Entity** records per request. If you encounter a "Request too large" error, please narrow the scope of your request by using more narrow **{IndexKeyEntryKeys}**, specifying fewer **{EntityKeys}**, or specifying a smaller **{DateQuery} or {fullTextSearch}** (see Resource Misuse for more information). *Note: This resource may not be used to download all of your data; please use the* Schema Data Backup Resource *instead.*

Example: All Entity Data

Personal Contact Information for all users from the University data:

```
GET /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-University
          /PCI
```

Example: Entity Data in Date Range
Presentations for all users from the University data, that occurred with the past year:

```
GET /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-University
          /PRESENT?startDate=2007-01-01&endDate=2007-12-31
```

Example: All Entity Data with Index Entries
Mathematics and Statistics Department's Personal Contact Information from the University data:

```
GET /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-University
          /DEPARTMENT:Mathematics,DEPARTMENT:Statistics/PCI
```

Example: User's Entity Data
Fred Flintstone's Personal Contact Information and Administrative Data - Yearly records from the Business college data. *Note: This case may be better addressed with User Data Resources.*

```
GET /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-Business
          /USERNAME:FFlintstone/PCI,ADMIN
```

Example: Entity Data Matching a Search
Presentations for all users from the University data, that contain the phrase "mobile phones":

```
GET /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-
University/PRESENT?fullTextSearch=%22mobile+phones%22
```

Schema Data RelaxNG Resource

Formally defines the document available via the Schema Data Query Resource and accepted by the Schema Data Import Resource.

```
GET /login/service/v4/SchemaData:relaxng/{SchemaKey}
```

Schema Data Import Resource

```
POST /login/service/v4/SchemaData/{SchemaKey}
          [/{IndexKeyEntryKeys}]
          [/{EntityKeys}]
```

Import data for one or more users, using the same format returned from the Schema Data Query Resource as formally defined by the Schema Data RelaxNG Resource.

To import data, you must first transform your data to match that required by the RelaxNG schema. If the source data is XML, XSLT may be an appropriate transformation tool. Other formats may require a custom conversion script/program or the use of a more advanced tool like Stylus Studio.

You are not required to send data for every field on a screen; only certain fields must be specified:

- Required fields.
- Fields that comprise the entity's primary key.
- Fields that determine the availability of other fields. *Note: Not all organizations make use of this functionality.* Some schemas contain fields that are configured to only be available under specified conditions. For example:
    - A university with multiple colleges, some of which require additional fields be available for accreditation purposes. A schema may be configured so that such fields are only available for members of those colleges.
    - An Intellectual Contributions screen with a link to the Journals Instrument have some fields dependent on the Contribution Type selected. When a Journal type is selected, additional fields become available to capture the related Journal name. Likewise, when a non-Journal contribution

type is selected, additional fields become available to capture additional information about the publication.

Such fields must be specified when importing data into the fields that would initially be hidden.

Non-required fields may be omitted from an import request if values are not part of your source data; simply do not include an XML element for the field. Empty or blank values can be explicitly imported by sending an empty XML element; either `<MNAME/>` or `<MNAME></MNAME>` specify that existing data in the field should be removed.

To insert data, simply `POST` the transformed XML data to the Schema Data Import Resource URL.

To update existing data, two methods may be considered:

If you need to view the existing data prior to updating it, use the Schema Data Query Resource to download the existing data and make the necessary modifications. Once ready, simply `POST` the modified data to the Import Resource. As long as the `"@id"` attribute values are still in place, the Import Resource will find the corresponding records and perform the update.

If the data update is not dependant on existing data, you may choose to construct an XML document as if inserting data. However, prior to updating data in this manner, Digital Measures will need to configure a Primary Key for each Entity you wish to update. This Primary Key may consist of one or more data fields already populated with data; please ask your Digital Measures Administrator to submit a request to configure a Primary Key, with an appropriate list of fields. *Note: If the specified Primary Key does not actually uniquely identify stored data, the Import Resource will fail while performing the update. All changes already processed will safely be rolled back, but the update will not complete. Please choose a Primary Key carefully.* Any fields referenced by the Primary Key must be included in the data sent to the Import Resource, so that the Primary Key may be matched. *Note: If a Primary Key exists for an Entity,* ***and*** *`"@id"` attribute values from the Query Resource are included, the `"@id"` attributes will be used to find the corresponding records in preference to any Primary Key values. This is the only supported mechanism to update fields that are part of a Primary Key.*

Regardless of the update method used, any data records that cannot be matched to an existing record will be created rather than updated. *Note: If you are updating only a portion of your existing records, you are not required to send all records, only those that have changed.*

Pairs of **`{IndexKeys}`** and **`{IndexEntryKey}`** values may be specified on the URL to indicate data are only present for users in a specific department or college. Specifying one or more **`{EntityKey}`** values indicates data is only present for those entities. If the POST data do not match these optional restrictions, the update will be prevented from completing. This capability may be useful as a safety mechanism to prevent unintended data from being stored.

This resource is restricted to 20,000 **`Entity`** records per request. This number includes new or updated **`Entity`** records, *and* **`Entity`** records loaded when looking for Primary Key matches. If you encounter a `"Request too large"` error, please break your import requests into smaller documents (see Resource Misuse for more information).

Example: Import New Records

Insert two new Intellectual Contribution records into Fred Flintstone's data:

```
POST /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-Business
        /USERNAME:FFlintstone
```

```
<Data>
    <Record username="FFlintstone">
        <INTELLCONT>
            <CONTYPE>Book, Edited</CONTYPE>
            <TITLE>Brontosaurus Crane Service Manual</TITLE>
            <!-- ... -->
        </INTELLCONT>
        <INTELLCONT>
            <CONTYPE>Journal Article</CONTYPE>
            <TITLE>Optimization of Brontosaurus Based Mining Operations</TITLE>
```

```
            <!-- ... -->
        </INTELLCONT>
    </Record>
</Data>
```

Example: Update Records by Id
Update Fred Flintstone's and Wilma Flintstone's Scheduled Teaching - Mean Course Evaluation Score based on results from the Query Resource:

```
GET /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-Business
        /DEPARTMENT:Accounting/SCHTEACH
```

Modify the results of the Query Resource to contain the updated data, and send to the Data Import resource. *Note: namespaces and metadata returned by the Query Resource may be included when updating data, and are simply ignored.*

```
POST /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-Business
```

```
<Data xmlns:dmd="http://www.digitalmeasures.com/schema/data-metadata" dmd:date="2007-09-27">
    <Record userId="12345" username="FFlintstone" dmd:id="10002">
        <SCHTEACH id="21235" dmd:lastModified="2007-03-23T15:23:56">
            <MEAN_EVAL>3.25</MEAN_EVAL>
        </SCHTEACH>
    </Record>
    <Record userId="56232" username="wflintstone" dmd:id="10352">
        <SCHTEACH id="62345" dmd:lastModified="2007-03-23T15:28:39">
            <MEAN_EVAL>3.45</MEAN_EVAL>
        </SCHTEACH>
    </Record>
</Data>
```

Example: Update Records by Primary Key
Update Fred Flintstone's and Wilma Flintstone's Scheduled Teaching - Mean Course Evaluation Score based on a Primary Key of Year, Term, Course Prefix, Number and Section:

```
POST /login/service/v4/SchemaData/INDIVIDUAL-ACTIVITIES-Business
```

```
<Data>
    <Record username="FFlintstone">
        <SCHTEACH>
            <TYY_TERM>2007-2008</TYY_TERM>
            <TYT_TERM>Fall</TYT_TERM>
            <COURSEPRE>ACCT</COURSEPRE>
            <COURSENUM>101</COURSENUM>
            <SECTION>1</SECTION>
            <MEAN_EVAL>3.25</MEAN_EVAL>
        </SCHTEACH>
    </Record>
    <Record username="wflintstone">
        <SCHTEACH>
            <TYY_TERM>2007-2008</TYY_TERM>
            <TYT_TERM>Fall</TYT_TERM>
            <COURSEPRE>ACCT</COURSEPRE>
            <COURSENUM>302</COURSENUM>
            <SECTION>8</SECTION>
            <MEAN_EVAL>3.45</MEAN_EVAL>
        </SCHTEACH>
    </Record>
</Data>
```

Schema Data Validate Resource

```
POST /login/service/v4/SchemaData:validate/{SchemaKey}
        [/{IndexKeyEntryKeys}]
        [/{EntityKeys}]
```

Perform a "dry run" on data to be imported. This resource is nearly identical to Schema Data Import Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the Schema Data RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.
- All processing steps are performed except for saving the data changes. This includes validation of Primary Key references.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

Schema Data Backup Resource

```
GET /login/service/v4/SchemaData:backup/{SchemaKey}
```

Download a complete backup of your Schema Data, refreshed on a weekly basis. Backups are returned as a Zip file containing a CSV data for each Entity in the system, suitable for importing into a relational database system.

Backups are not enabled by default; if you require weekly backups, please ask your Digital Measures Administrator to submit a task request to enable this functionality.

Note: All data stored by Digital Measures is replicated to a disaster recovery data center, and additionally backed up each night to three geographically-dispersed locations. Backup provided by this service are intended for integration or data mining purposes, not as the primary backup of data.

Schema Data Delete Resource

```
POST /login/service/v4/SchemaData:delete/{SchemaKey}/[{IndexKeyEntryKeys}]
[?{DateQuery}]
```

Permanently delete data records for one or more users, using the format formally defined by the Schema Data Delete RelaxNG Resource. *Note: This resource **cannot** be used to delete individual Dynamic Sub-Answer (DSA) rows, like Authors on Intellectual Contributions.*

Pairs of **{IndexKeys}** and **{IndexEntryKey}** values may be specified on the URL to indicate data should only be deleted from users in a specific department or college. The **{DateQuery}** argument may be specified on the URL to indicate data should only be deleted for records within the specified date range. Deleting by a date range will only delete records within that date range. Non-dated records will not be deleted when a date range is supplied. Combining these optional restrictions may be useful as a safety mechanism to prevent unintended data from being deleted.

The service responds with a list of the IDs of the deleted records. Any ID values specified but not found will be listed as missing.

```
<Success>
   <SCHTEACH>
      <item id="298357987">
                  <deleted/>
      </item>
      <item id="298357235">
         <deleted/>
      </item>
      <item id="11980933">
         <missing/>
      </item>
   </SCHTEACH>
</Success>
```

This resource is restricted to deleting 20,000 records per **Entity** per request. If you encounter a `Request too large` error, please narrow the scope of your request by using more narrow **{IndexKeyEntryKeys}**, specifying fewer Entities, specifying fewer IDs, or specifying a smaller **{DateQuery}** (see Resource Misuse for more information).

Example: Delete Entity Data

Scheduled Teaching records.

```
POST /login/service/v4/SchemaData:delete/INDIVIDUAL-ACTIVITIES-University
```

```
<Data>
   <SCHTEACH>
      <all/>
   </SCHTEACH>
</Data>
```

Example: Delete Entity Data by IDs

Scheduled Teaching records by ID.

```
POST /login/service/v4/SchemaData:delete/INDIVIDUAL-ACTIVITIES-University
```

```
<Data>
   <SCHTEACH>
      <item id="298357987"/>
      <item id="298357235"/>
   </SCHTEACH>
</Data>
```

Example: Delete User's Entity Data

Fred Flintstone's Scheduled Teaching records.

```
POST /login/service/v4/SchemaData:delete/INDIVIDUAL-ACTIVITIES-
University/USERNAME:FFlintstone
```

```
<Data>
   <SCHTEACH>
      <all/>
   </SCHTEACH>
</Data>
```

Schema Data Delete RelaxNG Resource

```
GET /login/service/v4/SchemaData:delete-relaxng/{SchemaKey}
```

Formally defines the document accepted by the Schema Data Delete Resource.

Schema Data Delete Validate Resource

```
POST /login/service/v4/SchemaData:delete-
validate/{SchemaKey}/[{IndexKeyEntryKeys}][?{DateQuery}]
```

Perform a "dry run" on data to be deleted. This resource is nearly identical to Schema Data Delete Resource, except for the data is not deleted.

User Resources

User Resources provide read and write access to user accounts stored by Digital Measures.

User List Resource

```
GET /login/service/v4/User[/{SchemaKey}[/{IndexKeyEntryKeys}]]
          [?firstName=firstNameSearch&lastName=lastNameSearch]
```

Presents a list of users currently stored in the system. If the optional **{SchemaKey}** is specified, only users who currently may store data in that Schema are returned. If optional **{IndexKeyEntryKeys}** are specified, only users with the specified Index Entry values are returned. If optional name search values are specified, only users that have names starting with the specified substring are returned.

- **/Users/User/@username**
  Unique login identifier for each user. This is the identifier users enter to log into Digital Measures. This is often the user's email address without the domain name or a unique identifier assigned by your organization.
- **/Users/User/@{UserIdentifierType}**
  Any user identifier values for the user will be present as attributes on the User element. If a user does not have a user identifier value for a user identifier type configured for your organization, no user identifier attribute will be present.
- **/Users/User/Item/@xlink:href**
  Resource URL to view details of this user account from the User Item Resource.

Example: All Users
Retrieve all users:

```
GET /login/service/v4/User
```

Example: All Users for a Schema
Retrieve users with access to University data:

```
GET /login/service/v4/User/INDIVIDUAL-ACTIVITIES-University
```

Example: All Users for a Schema and Index Entry
Retrieve users with access to University data in the College of Business

```
GET /login/service/v4/User/INDIVIDUAL-ACTIVITIES-University/COLLEGE:Business
```

Example: Users by Name
Retrieve users with access to University data and with a last name starting with "F":

```
GET /login/service/v4/User/INDIVIDUAL-ACTIVITIES-University?lastName=F
```

Example: Users by Name and Index Entry
Retrieve users with access to University data in the College of Business and with a last name starting with "Flintstone" and a first name starting with "F":

```
GET /login/service/v4/User/INDIVIDUAL-ACTIVITIES-University/COLLEGE:Business
          ?lastName=Flintstone&firstName=F
```

User List RelaxNG Resource

```
GET /login/service/v4/User:list-relaxng
```

Formally defines the document available via the User List Resource.

User Item Resource

```
GET /login/service/v4/User/USERNAME:{Username}
```

Presents the details of a single user's account, including authentication mechanism, and links to security role and data schema resources.

- **/User/@username**
  Unique login identifier for each user. This is the identifier users enter to log into Digital Measures. This is often the user's email address without the domain name or a unique identifier assigned by your organization.
- **/User/@enabled**
  Indicates if the user account is currently enabled. Disabled accounts may not log in, but any data is preserved for later reporting.
- **/User/@{UserIdentifierType}**
  Any user identifier values for the user will be present as attributes on the User element. If a user does not have a user identifier value for a user identifier type configured for your organization, no user identifier attribute will be present.
- **/User/LocalAuthentication**
  || /User/LDAPAuthentication
  || /User/PortalAuthentication
  Indicates the authentication mechanism assigned to the user.
- **/User/dmu:Schemas/@xlink:href**
  Resource URL to view the user's associations with schemas for data collection via User Schema Resource.
- **/User/dmu:Roles/@xlink:href**
  Resource URL to view the user's associations with security roles via the User Roles Resource.

User Item RelaxNG Resource

```
GET /login/service/v4/User:item-relaxng
```

Formally defines the document available via the User Item Resource.

User Create Resource

```
POST /login/service/v4/User
```

Create a new user account without access to any Schemas or Security Roles, using the same format as returned by the User Item Resource, and formally defined by the User Create RelaxNG Resource.

Example: Create User Account
Create a new user Fred Flintstone with username "FFlintstone":

```
POST /login/service/v4/User/
```

```
<User username="FFlintstone">
    <FirstName>Frederick</FirstName>
    <MiddleName>J</MiddleName>
    <LastName>Flintstone</LastName>
    <Email>fflintstone@bedrock.edu</Email>
    <LocalAuthentication>wilma123</LocalAuthentication>
</User>
```

The service responds with the user's newly created resource URL:

```
<dmu:Success xmlns:xlink="http://www.w3.org/1999/xlink">
    <Updated xlink:type="simple" xlink:href="/login/service/v4/User/USERNAME:FFlintstone"/>
</dmu:Success>
```

User identifier values may be used in addition to the username attribute if your organization has set up user identifier types through Digital Measures. This allows you to provide additional custom unique user identifying values for a user. For example, if your organization has a **{UserIdentifierType}** of "bannerId", you may specify the user using the `@bannerId` attribute:

```
<User username="FFlintstone" bannerId="323">
...
</User>
```

User Create RelaxNG Resource

```
GET /login/service/v4/User:create-relaxng
```

Formally defines the document accepted by the User Create Resource.

User Create Validation Resource

```
POST /login/service/v4/User:create-validate
```

Perform a "dry run" on a user to be created. This resource is nearly identical to the User Create Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the User Create RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.
- All processing steps are performed except for saving the user changes.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

User Update Resource

```
PUT /login/service/v4/User/USERNAME:{Username}
```

Update an existing user account, using the same format as returned by the User Item Resource, and formally defined by the User Update RelaxNG Resource.

Example: Update User Details
Update Wilma Slaghoople's last name, email address, and username:

```
PUT /login/service/v4/User/USERNAME:WSlaghoople
```

```
<User username="WFlintstone">
    <LastName>Flintstone</LastName>
    <Email>wflintstone@bedrock.edu</Email>
</User>
```

The service responds with the user's updated resource URL:

```
<dmu:Success xmlns:xlink="http://www.w3.org/1999/xlink">
    <Updated xlink:type="simple" xlink:href="/login/service/v4/User/USERNAME:WFlintstone"/>
</dmu:Success>
```

Example: Update Custom User Identifier
Update Barney Rubble's bannerId:

```
PUT /login/service/v4/User/USERNAME:BRubble
```

```
<User bannerId="999"/>
```

Example: Disable User Account
Disable Barney Rubble's account:

```
PUT /login/service/v4/User/USERNAME:BRubble
```

```
<User enabled="false"/>
```

User Update RelaxNG Resource

```
GET /login/service/v4/User:update-relaxng
```

Formally defines the document accepted by the User Update Resource. *Note: This resource is similar to but not identical to the User Item RelaxNG Resource; the differences allow partial specification of updated data by indicating many elements are "optional".*

User Update Validation Resource

```
PUT /login/service/v4/User:update-validate/USERNAME:{Username}
```

Perform a "dry run" on a user to be updated. This resource is nearly identical to the User Update Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the User Update RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.
- All processing steps are performed except for saving the user changes.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

User Delete Resource

```
DELETE /login/service/v4/User/USERNAME:{Username}
```

Permanently delete a user and all associated data.

Note: Deletion of users is not recommended in most scenarios, as the historical accuracy of reports may be affected. If the user has left your organization and you simply wish to block further use, consider disabling the account per the User Update Resource example. This resource is most appropriately used to delete records created by mistake or for testing purposes.

User Schema Resources

A User Schema resource links a User account to a set of data collection screens. A User Schema relationship must be established before any data may be stored for a user.

A User Schema resource also contains a partial representation of the user's data. The representation is based on the location of indexed questions, as described under Schema Index Resources. The partial data representation will also include date questions when an indexed question occurs on a data collection screen that captures yearly records. A user's index entries are drawn from the user's most recent yearly record.

User Schema Resources provide read and write access to User Schema relationships.

User Schema List Resource

```
GET /login/service/v4/UserSchema/USERNAME:{Username}
```

Presents a list of Schemas currently configured for a specified User.

- **`/UserSchemas/UserSchema/@userSchemaKey`**
  Identifier for a User's relationship to a Schema, referenced as **`{UserSchemaKey}`** by other resources.
  Note: A **`{UserSchemaKey}`** *is unique only in conjunction with a Username.*
  Note: A **`{UserSchemaKey}`** *is sometimes similar to the* **`{SchemaKey}`** *of the data collection screens being referenced, but this is not guaranteed. Please only use* **`{UserSchemaKey}`** *values that are retrieved from this resource; do not be tempted to use a hard-coded* **`{SchemaKey}`** *instead.*
- **`/UserSchemas/UserSchema/@xlink:href`**
  Resource URL to view details of this User Schema relationship from the User Schema Item Resource.

User Schema List RelaxNG Resource

```
GET /login/service/v4/UserSchema:list-relaxng/
```

Formally defines the document available via the User Schema List Resource.

User Schema Item Resource

```
GET /login/service/v4/UserSchema/USERNAME:{Username}/{UserSchemaKey}
```

Presents the details of the specified user's relationship with a Schema, including **`{IndexKey}`** and **`{IndexEntryKey}`** pairs, and a partial representation of the user's data.

- **`/{SchemaKey}/dmd:IndexEntry`**
  List of **`{IndexKey}`** and **`{IndexEntryKey}`** pairs associated with this User Schema relationship.
- **`/{SchemaKey}/{EntityKey}`**
  Partial representation of user's data stored in the Schema.

User Schema Item RelaxNG Resource

```
GET /login/service/v4/UserSchema:item-relaxng/
```

Formally defines the document available via the User Schema Item Resource.

User Schema Create Resource

```
POST /login/service/v4/UserSchema/USERNAME:{Username}
```

Create a new User Schema relationship, using the same format as returned by the User Schema Item Resource, and formally defined by the User Schema Create RelaxNG Resource.

Example: Create User Schema Relationship
Grant Barney Rubble access to the University data collection screens:

```
POST /login/service/v4/UserSchema/USERNAME:BRubble
```

```
<INDIVIDUAL-ACTIVITIES-University>
    <ADMIN>
        <AC_YEAR>2007-2008</AC_YEAR>
        <ADMIN_DEP>
            <DEP>Management</DEP>
        </ADMIN_DEP>
    </ADMIN>
</INDIVIDUAL-ACTIVITIES-University>
```

User Schema Create RelaxNG Resource

```
GET /login/service/v4/UserSchema:create-relaxng
```

Formally defines the document accepted by the User Schema Create Resource.

User Schema Create Validation Resource

```
POST /login/service/v4/UserSchema:create-validate/USERNAME:{Username}
```

Perform a "dry run" on user schema relationship to be created. This resource is nearly identical to the User Schema Create Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the User Create RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.
- All processing steps are performed except for saving the user schema changes.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

User Schema Update Resource

```
PUT /login/service/v4/UserSchema/USERNAME:{Username}/{UserSchemaKey}
```

Update an existing User Schema relationship, using the same format as returned by the User Schema Item Resource, and formally defined by the User Schema Update RelaxNG Resource.

Updates to the partial representation of a user's data are expected, but in some cases may result in some data being removed from the user's records. For example, assume your data collection screens are configured to capture multiple departments **and** the percentage of time dedicated to that department for each user. Fred Flintstone has been reassigned from the Management and Marketing departments to the Management and Economics departments. His records include values for the percentage of time dedicated fields, represented in the following Schema Data Query Resource fragment:

```
    <!-- ... -->
    <ADMIN>
        <AC_YEAR>2007-2008</AC_YEAR>
        <ADMIN_DEP>
            <DEP>Management</DEP>
            <PERCENT_DEDICATION>80</PERCENT_DEDICATION>
        </ADMIN_DEP>
        <ADMIN_DEP>
            <DEP>Marketing</DEP>
            <PERCENT_DEDICATION>20</PERCENT_DEDICATION>
        </ADMIN_DEP>
    </ADMIN>
    <!-- ... -->
```

When the update is performed, Fred's Management fields are unchanged, and no data is removed. However, he has left the Marketing department for the Economics department, so all fields related to the Marketing department are removed, and a new set of fields are created for the Economics department. The result is the loss of his percentage of time dedicated field values for the Marketing department, resulting in the following fragment:

```
    <!-- ... -->
    <ADMIN>
```

```
        <AC_YEAR>2007-2008</AC_YEAR>
        <ADMIN_DEP>
            <DEP>Management</DEP>
            <PERCENT_DEDICATION>80</PERCENT_DEDICATION>
        </ADMIN_DEP>
        <ADMIN_DEP>
            <DEP>Economics</DEP>
            <PERCENT_DEDICATION/>
        </ADMIN_DEP>
    </ADMIN>
    <!-- ... -->
```

In most cases, this type of data removal is desired and expected; if this is not the case for your update, please request assistance from Digital Measures.

Example: Update Index Entries for User
View Fred Flintstone's existing Administrative Data:

```
GET /login/service/v4/UserSchema/
        USERNAME:FFlintstone/INDIVIDUAL-ACTIVITIES-University
```

```
<INDIVIDUAL-ACTIVITIES-University>
    <dmd:IndexEntry indexKey="DEPARTMENT" entryKey="Management" text="Management"/>
    <dmd:IndexEntry indexKey="DEPARTMENT" entryKey="Marketing" text="Marketing"/>
    <ADMIN>
        <AC_YEAR>2007-2008</AC_YEAR>
        <ADMIN_DEP>
            <DEP>Management</DEP>
        </ADMIN_DEP>
        <ADMIN_DEP>
            <DEP>Marketing</DEP>
        </ADMIN_DEP>
    </ADMIN>
</INDIVIDUAL-ACTIVITIES-University>
```

Change Fred Flintstone's departments to Management and Economics for 2007-2008:

```
PUT /login/service/v4/UserSchema/
        USERNAME:FFlintstone/INDIVIDUAL-ACTIVITIES-University
```

```
<INDIVIDUAL-ACTIVITIES-University>
    <ADMIN>
        <AC_YEAR>2007-2008</AC_YEAR>
        <ADMIN_DEP>
            <DEP>Management</DEP>
        </ADMIN_DEP>
        <ADMIN_DEP>
            <DEP>Economics</DEP>
        </ADMIN_DEP>
    </ADMIN>
</INDIVIDUAL-ACTIVITIES-University>
```

User Schema Update RelaxNG Resource

```
GET /login/service/v4/UserSchema:update-relaxng
```

Formally defines the document accepted by the User Schema Update Resource.

User Schema Update Validation Resource

```
POST /login/service/v4/UserSchema:update-validate/
        USERNAME:{Username}/{UserSchemaKey}
```

Perform a "dry run" on a user schema relationship to be updated. This resource is nearly identical to the User Schema Update Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the User Create RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.

- All processing steps are performed except for saving the user schema changes.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

User Schema Delete Resource

```
DELETE /login/service/v4/UserSchema/USERNAME:{Username}/{UserSchemaKey}
```

Permanently delete a user's relationship with a schema and all associated data.

Note: Deletion of user-schema relationships is not recommended in most scenarios, as the historical accuracy of reports may be affected. If the user has left your organization and you simply wish to block further logins by this user, consider disabling the account per the User Update Resource example*. This resource is most appropriately used to delete records created by mistake or for testing purposes.*

Role Resources

A role resource defines a set of security capabilities that may be assigned to one or more users.

Roles are composed of capabilities which grant access to a set of functionality related to a single Schema. Some capabilities allow additional levels of configuration. For example, the Custom Reports capability by itself only allows a user to view the Custom Reports menu item; Reports must be added to the capability to allow a user to run custom reports.

All roles have a scope, which indicates the level of data access the role may grant. Common scopes include university, college, department, and self. Assignment of which specific college or department a user may work with is controlled by Permissions, assigned via User Role Resources.

Some roles contain capabilities that Digital Measures considers inappropriate or dangerous for customers to manage directly. Examples include the capabilities that mark a user as an organization's Digital Measures Administrator, or grant access to web services. These Roles are available for viewing via the Role Item Resource, but not assignment via User Role Resources. To request changes to a user's assignment to an immutable Role, please ask your Digital Measures Administrator to submit a task request.

Role Resources provide read-only access. This resource is primarily informational for the web service client developer interested in assigning Users to Roles via User Role Resources.

Role List Resource

```
GET /login/service/v4/Role
```

Presents a list of Roles stored by Digital Measures.

- **`/Roles/Role/@roleKey`**
  Unique identifier for role, referenced as **`{RoleKey}`** by other resources.
- **`/Roles/Role/@text`**
  Name of role intended for display.
- **`/Roles/Role/Item/@xlink:href`**
  Resource URL to view details of role from the Role Item Resource.
- **`/Roles/Role/Users/@xlink:href`**
  Resource URL to view all Users currently assigned to role from the Role User List Resource.

Role List RelaxNG Resource

```
GET /login/service/v4/Role
```

Formally defines the document available via the Role List Resource.

Role Item Resource

```
GET /login/service/v4/Role/{RoleKey}
```

Presents the details of the specified Role, including all capabilities and a list of available Permissions, when available.

- **`/Role/@text`**
  Name of the role intended for display.
- **`/Role/@schemaKey`**
  Identifies which Schema the role applies to.
- **`/Role/@scope`**
  Indicates the level of data access the role is intended to assign.
- **`/Role/Capabilities`**
  Identifies the security capabilities that the role grants.

- **/Role/dmr:Permissions**
  Provides a list of possible **{PermissionKey}** values that may be used in conjunction when assigning this role via the User Role Resources. Not all scopes allow assignment of permissions; specifically, the organization scope grants access to all data without restriction, and the self scope grants access only to the assigned user's data.

Role Item RelaxNG Resource

```
GET /login/service/v4/Role:item-relaxng[/{SchemaKey}]
```

Formally defines the document available via the Role Item Resource. A **{SchemaKey}** may optionally be specified to include capabilities and scopes that may be Schema-dependent.

Role User List Resource

```
GET /login/service/v4/RoleUser/{RoleKey}
```

Presents a list of users assigned to the specified Role.

- **/RoleUsers/User/@username**
  Username of user assigned to the specified role.
- **/RoleUsers/User/@{UserIdentifierType}**
  Any user identifier values for the user will be present as attributes on the User element. If a user does not have a user identifier value for a user identifier type configured for your organization, no user identifier attribute will be present.
- **/RoleUsers/User/UserItem/@xlink:href**
  Resource URL to view details of this Role from the Role Item Resource.
- **/RoleUsers/User/UserRole/@xlink:href**
  Resource URL to view details of this User's assignment to this Role from the User Role Item Resource.

Role User List RelaxNG Resource

```
GET /login/service/v4/RoleUser:list-relaxng
```

Formally defines the document available via the Role User List Resource.

User Role Resources

A User Role resource links a user account to a security role to grant authorization and data access.

A User Role resource may contain one or more permissions that restrict a user's access to data; the role's scope determines the availability of data permissions. A scope of "organization" allows access to all data without restriction, and hence does not allow a specific permission to be assigned. A scope of "self" allows access only to the user's own data. Other scopes, such as "college" or "department", require that at least one permission is assigned, specifying which the college or department the user should be able to access.

Multiple roles may be assigned to a user. Consider a user who is acting both as an individual faculty member, and as a department chair. The user is assigned a "faculty" role containing capabilities needed by all faculty members, such as running a few basic reports on the user's own data. The user is also assigned a "department chair" role that grants data access to all users in the department, and contains more advanced capabilities, such as the ability to manage data, and to run a larger set of reports. What data set can the user report on?

- If a report is assigned as part of a single security role, the user can access data with the permissions assigned to that security role.
  In this example, the "department chair" role may include to a yearly department-level status report; the user may run the report for any user in the department. Similarly, the "faculty" role may include a "vita" report; the user may run the report only on the user's own data.
- If a report is assigned as part of multiple security roles, the user can access data with permissions from the most permissive security scope.
  In this example, both the "department chair" and "faculty" roles may include a publications list report; the user may run the report for any user in the department.
- If a report is assigned as part of multiple security roles with the same role scope, data permissions are accumulated.
  In this example, consider if the user was acting as chair of multiple departments; the user may run the report for any user in either department.

User Role List Resource

```
GET /login/service/v4/UserRole/USERNAME:{Username}
```

Presents a list of Roles stored assigned to the specified user account.

- **`/UserRoles/UserRole/@text`**
  Name of role intended for display.
- **`/UserRole/UserRole/@xlink:href`**
  Resource URL to view details of this User Role from the User Role Item Resource.

User Role List RelaxNG Resource

```
GET /login/service/v4/UserRole:list-relaxng
```

Formally defines the document available via the User Role List Resource.

User Role Item Resource

```
GET /login/service/v4/UserRole/USERNAME:{Username}/{RoleKey}
```

Presents the details of the specified User Role relationship, including assigned permissions when available.

- **`/{RoleKey}/dmr:Role/@xlink:href`**
  Resource URL to view details of this role from the Role Item Resource.
- **`/Role/Permission/@permissionKey`**
  Specifies the data permissions that should be applied to the role for the user.

User Role Item RelaxNG Resource

```
GET /login/service/v4/UserRole:item-relaxng
```

Formally defines the document available via the User Role Item Resource.

User Role Create Resource

```
POST /login/service/v4/UserRole/USERNAME:{Username}
```

Create a new User Role relationship, using the same format as returned by the User Role Item Resource, and formally defined by the User Role Create RelaxNG Resource.

Example: Create User Role Relationship
Grant Fred Flintstone the Department Chair role for the Economics department:

```
POST /login/service/v4/UserRole/USERNAME:FFlintstone
```

```
<INDIVIDUAL-ACTIVITIES-Business-DEPARTMENTCHAIR>
    <Permission permissionKey="Economics"/>
</INDIVIDUAL-ACTIVITIES-Business-DEPARTMENTCHAIR>
```

User Role Create RelaxNG Resource

```
GET /login/service/v4/UserRole:create-relaxng
```

Formally defines the document accepted via the User Role Create Resource. *Note: This resource is currently identical to the* User Role Item RelaxNG Resource.

User Role Create Validate Resource

```
POST /login/service/v4/UserRole:create-validate/USERNAME:{Username}
```

Perform a "dry run" on user role relationship to be created. This resource is nearly identical to the User Role Create Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the User Role Create RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.
- All processing steps are performed except for saving the user schema changes.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

User Role Update Resource

```
PUT /login/service/v4/UserRole/USERNAME:{Username}/{RoleKey}
```

Update a new User Role relationship, altering the Permissions assigned to a user, using the same format as returned by the User Role Item Resource, and formally defined by the User Role Update RelaxNG Resource.

Example: Modify User Role Relationship
Change Fred Flintstone's permission for the Department Chair role to Management:

```
PUT /login/service/v4/UserRole/USERNAME:FFlintstone/INDIVIDUAL-ACTIVITIES-Business
```

```
<INDIVIDUAL-ACTIVITIES-Business-DEPARTMENTCHAIR>
    <Permission permissionKey="Management"/>
</INDIVIDUAL-ACTIVITIES-Business-DEPARTMENTCHAIR>
```

User Role Update RelaxNG Resource

```
GET /login/service/v4/UserRole:update-relaxng
```

Formally defines the document accepted via the User Role Update Resource. *Note: This resource is currently identical to the* User Role Item RelaxNG Resource.

User Role Update Validate Resource

```
PUT /login/service/v4/UserRole:update-validate/USERNAME:{Username}/{RoleKey}
```

Perform a "dry run" on user role relationship to be updated. This resource is nearly identical to the User Role Update Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the User Role Update RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.
- All processing steps are performed except for saving the user schema changes.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

User Role Delete Resource

```
DELETE /login/service/v4/UserRole/USERNAME:{Username}/{RoleKey}
```

Permanently delete a user's relationship with a security role. *Note: A useable user account must remain assigned to at least one role.*

User Batch Resources

User Batch resources provide a consolidated tool for creating and updating multiple user accounts, user schema relationships, and user role relationships. The resource automatically determines if a specified resource should be created or updated based on the presence of existing records.

The batch resources are a convenient way to set up multiple user accounts and privileges in a single request, and are best suited for setting up a number of new users at the beginning of a term or year. The non-batch user resources are more appropriate for day-to-day management of individual users and privileges.

User Batch Save Resource

`POST /login/service/v4/UserBatch[/{SchemaKey}]`

Creates or updates one or more users using the same format returned from the List Resource, as formally defined by the User Batch RelaxNG Resource. If the optional **`{SchemaKey}`** is specified, the XML document may only contain Schema entries for that Schema, excluding others the service account may be able to access.

- **`/Users/User`**
  Accepts user records similar to those required by the User Update Resource, with the addition of `UserSchemas` and `UserRoles` elements detailed below. User identifier values are able to uniquely identify a user in this resource, in the same format as the User Update Resource.
- **`/Users/User/UserSchemas`**
  Optionally contains user-schema relationships that should be created or updated for the current user.
- **`/Users/User/UserSchemas/{UserSchemaKey}`**
  Accepts user-schema relationship records similar to those required by the User Schema Update Resource.
- **`/Users/User/UserRoles`**
  Optionally contains user-role relationships that should be created or updated for the current user.
- **`/Users/User/UserRoles/{RoleKey}`**
  Accepts user-role relationship records similar to those required by the User Role Update Resource.

This resource is restricted to 20,000 **`Entity`** records per request, which are contained by **`{UserSchemaKey}`** elements. This number includes new or updated **`Entity`** records, *and* **`Entity`** records loaded when looking for Primary Key matches. If you encounter a "`Request too large`" error, please break your import requests into smaller documents (see Resource Misuse for more information).

User Batch RelaxNG Resource

`GET /login/service/v4/UserBatch:save-relaxng[/{SchemaKey}]`

Formally defines the document accepted via the User Batch Save Resource, optionally limited to only support Schemas and Roles for the specified **`{SchemaKey}`**, excluding others the service account may be able to access.

User Batch Validate Resource

```
POST /login/service/v4/UserBatch:save-validate[/{SchemaKey}]
```

Perform a "dry run" on user batch data. This resource is nearly identical to the User Batch Save Resource, except for the following differences:

- Data are not initially validated against the RelaxNG schema produced by the User Batch Save RelaxNG Resource. Instead, a more forgiving interval validation mechanism is used that categorizes and reports all errors found, instead of a single error per request.
- All processing steps are performed except for saving the user schema changes.
- If no other errors are found, the requested data is validated against the RelaxNG schema as a final check.

Appendix A: Sample HTTP Connection Code

Java: Jakarta HttpClient

http://jakarta.apache.org/httpcomponents/httpclient-3.x/

HttpClient supports high-level connection abstractions, including Basic Authentication, and is recommended instead of the Java platform's java.net.HttpURLConnection support.

```
/* Use PostMethod, PutMethod, or DeleteMethod when appropriate */
GetMethod method = new GetMethod("https://www.digitalmeasures.com/login/service/v4/...");
method.addRequestHeader("Accept-Encoding", "gzip");

/* Uncomment when sending data via PostMethod or PutMethod
method.addRequestHeader("Content-Encoding", "gzip");
ByteArrayOutputStream compressedData = new ByteArrayOutputStream();
OutputStream compressor = null;
try
{
    compressor = new GZIPOutputStream(compressedData);
    compressor.write(dataBytes, 0, dataBytes.length);
}
finally
{
    if(compressor != null)
        compressor.close();
}
method.setRequestBody(new ByteArrayInputStream(compressedData.toByteArray()));
*/

HttpClient client = new HttpClient();
client.getState().setAuthenticationPreemptive(true);
client.getState().setCredentials("Digital Measures", "www.digitalmeasures.com",
        new UsernamePasswordCredentials(username, password));
InputStream in = null;
try
{
    int statusCode = client.executeMethod(method);
    in = new GZIPInputStream(method.getResponseBodyAsStream());
    // TODO: Handle HTTP status code and response data here
}
finally
{
    if(in != null)
        in.close();
    method.releaseConnection();
}
```

Additional Information

Java: java.net.HttpURLConnection

http://java.sun.com/j2se/1.3/docs/api/java/net/HttpURLConnection.html

`HttpURLConnection` provides basic low-level HTTP access. Basic Authentication is supported by adding an `Authorization` header to the request.

```
URL url = new URL("https://www.digitalmeasures.com/login/service/v4/...");
HttpURLConnection connection = (HttpURLConnection) url.openConnection();

String credentials = new BASE64Encoder().encode((username + ":" + password).getBytes());
connection.setRequestProperty("Authorization", "Basic " + credentials);
connection.setRequestProperty("Accept-Encoding", "gzip");

/* Uncomment when sending data via POST or PUT
connection.setDoOutput(true);
connection.setRequestMethod("PUT");
connection.setRequestProperty("Content-Encoding", "gzip");
*/

connection.connect();

/* Uncomment when sending data via POST or PUT
OutputStream out = null;
try
{
    out = new GZIPOutputStream(connection.getOutputStream());
    out.write(dataBytes, 0, dataBytes.length);
}
finally
{
    out.close();
}
*/

InputStream in = null;
try
{
    int statusCode = connection.getResponseCode();
    in = new GZIPInputStream(connection.getInputStream());
    // TODO: Handle HTTP status code and response data here
}
finally
{
    if(in != null) in.close();
}
```

Additional Information

.NET: System.Net.HttpWebRequest

http://msdn2.microsoft.com/en-us/library/system.net.httpwebrequest.aspx

Microsoft's `System.Net` namespace contains built-in support for Basic Authentication.

```
string uri = "https://www.digitalmeasures.com/login/service/v4/...";

CredentialCache credentialCache = new CredentialCache();
credentialCache.Add(
        new Uri("https://www.digitalmeasures.com"),
        "Basic",
        new NetworkCredential(username, password));


HttpWebRequest request = (HttpWebRequest) WebRequest.Create(uri);
request.AllowAutoRedirect = true;
request.PreAuthenticate = true;
request.Credentials = credentialCache;

request.AutomaticDecompression = DecompressionMethods.GZip;

/* Uncomment to send data
request.Method = "POST";
request.ContentType = "text/xml";
request.ContentLength = contentByteArray.Length;
using(Stream requestData = request.GetRequestStream())
{
    requestData.Write(contentByteArray, 0, contentByteArray.Length);
}
*/

HttpWebResponse response = null;
try
{
    response = (HttpWebResponse) request.GetResponse();
    HttpStatusCode statusCode = response.StatusCode;


    using(Stream responseData = response.GetResponseStream())
    {
        // TODO: Handle HTTP status code and response data here
    }
}
catch(IOException e)
{
    Debug.WriteLine(e.Message);
}
finally
{
    if(response != null)
        response.Close();
}
```

PHP: cURL

http://www.php.net/curl

PHP supports HTTPS and Basic Authentication via a lightweight wrapper around `libcurl`; the following code requires `libcurl` 7.17.1 or greater.

```
$curl = curl_init();

curl_setopt_array($curl, array(
    CURLOPT_URL            => 'https://www.digitalmeasures.com/login/service/v4/...',
    CURLOPT_USERPWD        => $user . ':' . $password,
    CURLOPT_ENCODING       => 'gzip',
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_RETURNTRANSFER => true));

$responseData = curl_exec($curl);

if(curl_errno($curl))
{
    $errorMessage = curl_error($curl);
    // TODO: Handle cURL error
}
else
{
    $statusCode = curl_getinfo($curl, CURLINFO_HTTP_CODE);
    // TODO: Handle HTTP status code and response data
}

curl_close($curl);
```