

See each method's wiki page for full description.

Item (General)

- [CreateItem](#)(Type) - returns an item id
- [GetItem](#)(ItemId,Schema) - returns the item in the specified schema, default schema being METS
- [GetType](#)(ItemId) - returns an item's type
- [GetStatus](#)(ItemId) - gets the status of the item
- [SetStatus](#)(ItemId,Status,OverrideValidation) - sets the status of the item, executing validation; in some cases, it might make sense to override the validation and force a status update
- [GetRelationships](#)(ItemId) - returns the relationships (item and relationship type) for the given item
- [CreateRelationship](#)(ItemAId,ItemBId,RelationshipType) - creates the relationship between items a and b
- [DeleteRelationship](#)(ItemAId,ItemBId,RelationshipType) - deletes the relationship between items a and b
- [GetItemInformation](#)(ItemId) - returns the additional item information that the repository may store about the item (cataloging history, status, type, etc) in one packet
- [SetItemInformation](#)(ItemId,ItemInfo) - sets the additional item information in the repository, the contained information would be implementation-specific

Item (Descriptive Metadata)

- [SupportedSchemas](#)(ItemId) - returns array of schemas supported by the item
- [GetDescriptiveMetadata](#)(ItemId,Schema) - returns the descriptive metadata for an item in the requested schema
- [SetDescriptiveMetadata](#)(ItemId,DM,Schema) - sets the descriptive metadata for an item, returns the derived metadata form for the item
- [GetMetadataForm](#)(ItemId,DMR) - returns the derived metadata form for the item, either against the stored DMR or against the DMR that is optionally passed
- [GetMetadataProfileName](#)(ItemId) - returns the name of the MP for an item
- [IsDescriptiveMetadataEditable](#)(ItemId) - returns boolean for whether or not metadata can be edited, based on item type; for some types, DMR is derived (e.g., TEI)

Item (Component Management)

- [GetValidComponentTypes](#)(ItemId) - returns valid component types for the given item
- [GetComponents](#)(ItemId) - returns component map (containing appropriate component identifiers, labels, types, and possibly pointers)
- [AddComponent](#)(ItemId,ComponentMap) - add placeholder for component w/o the object itself, returns component map and component identifier
- [AddComponent](#)(ItemId,ComponentMap,Stream) - add component with its object stream, returns component map and component identifier
- [ReplaceComponent](#)(ItemId,ComponentMap,Stream) - replace component with new object stream, returns component map and component identifier
- [DeleteComponent](#)(ItemId,ComponentMap) - delete component from the repository

Item (Misc)

- [GetThumbnailURL](#)(ItemId) - return URL for item thumbnail

Component

- [GetThumbnail](#)(ItemId,ComponentId) - return item thumbnail as a stream
- [GetPreviewURL](#)(ItemId,ComponentId) - return a URL to a preview of the item with
- [GetComponentMetadata](#)(ItemId,ComponentId)
- [SetComponentMetadata](#)(ItemId,ComponentId,AMR)

Metadata Application Profile (MAP)

- [GetMetadataApplicationProfile](#)(MAPName) - returns the complete MAP for an item (schema reference, usage guideline reference, validation and MetadataFormDefinition)
- [GetValidationRules](#)(MAPName) - returns validation rules for a MAP
- [GetMetadataFormDefinition](#)(MAPName) - returns the MetadataFormDefinition for a MAP

Index

- [Index?](#)(params)

ResultSet

- [Find](#)(params)
- [Find](#)(params, returnSchema)

API Method Specification

The repository is modeled in a Resource Oriented Architecture. All repository entities are resources, and are thus available and manipulated through RESTful interfaces. The API Methods listed above are implemented as RESTful web services. Details about each of these web services and how they are addressable are available in the Bindings section for each method.

The repository has a base URI. It is referred to throughout the bindings sections as repositoryURI.

Error codes and conditions are not yet addressed in the binding.

Note:

- *Authentication will need to be addressed in the binding specification for the API methods.*