



EnterWorks 10.2 API Overview

Revised 10/05/2020

- **EnterWorks 10.2 Release Enable Suite:**
 - EnterworksInstallPackage_20200630_1605
 - EnterworksDBScripts_20200630_1605
 - EnterworksPostgreSQLDBScripts_20200630_1605
- **EnterWorks Patch:**
 - EnableServer_10.2_Enable_patch_20200630_1539
 - ew-linux-10.2-patch-20200630_2004
- **EnterWorks Builds:**
 - Enable 10.2.0 Build 20200630_1539
 - EnableServer 10.2 Build 20200630_1539

Winshuttle

19820 North Creek Pkwy #200

Bothell, WA 98011

© Winshuttle
19820 North Creek Pkwy #200
Bothell, WA 98011

1.888.242.8356 (Sales and General Information)
1.888.225.2705 (U.S. Support)
<http://www.enterworks.com>

EnterWorks® Classic Administration Guide

Copyright © 2020 EnterWorks, Inc., a Winshuttle Company. All rights reserved.

Law prohibits unauthorized copying of all or any part of this document. Use, duplication, or disclosure by the U.S. Government is subject to the restrictions set forth in FAR 52.227-14.

“EnterWorks” and the “EnterWorks” logo are registered trademarks and “Enable PIM”, “EnterWorks PIM”, “EnterWorks Process Exchange” and “EnterWorks Product Information Management” are trademarks of EnterWorks, Inc.

Windows, .NET, IIS, SQL Server, Word, and Excel are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Java and all Sun and Java based trademarks are trademarks or registered trademarks of the Oracle Corporation in the United States and other countries.

Oracle is a registered trademark and Oracle 10g is a trademark of Oracle Corporation.

Pentium is a registered trademark of Intel Corporation in the United States and other countries.

JBoss is a registered trademark of Red Hat, Inc.

All other trademarks and registered trademarks are the property of their respective holders.

All icons and graphics, with the exception of the "e." logo, were obtained from West Coast Icons and Design at <http://www.bywestcoast.com>. EnterWorks, Inc. retains copyrights for all graphics unless otherwise stated. All other trademarks and registered trademarks are the property of their respective holders.

This document is furnished for informational purposes only. The material presented in this document is believed to be accurate at the time of printing. However, EnterWorks Acquisition, Inc. assumes no liability in connection with this document except as set forth in the License Agreement under which this document is furnished.

Contents

1	Customer Support	4
2	Document Terminology	5
3	Document Conventions	6
4	EnterWorks API Services	7
5	Rest Services	7
5.1	Rest Service APIs	7
5.1.1	Code Sets	9
5.1.2	Group and Users.....	10
5.1.3	Items.....	12
5.1.4	Repositories.....	14
5.1.5	Services.....	15
5.2	Calling the Rest Services.....	16
5.3	Rest Service Sample Code	16
6	SOAP Services.....	17
6.1	Services.....	18
6.1.1	Metadata Retrieval.....	19
6.1.2	Repository/Item Management	21
6.1.3	Tools	23
6.2	SOAP Service Sample Code	24

1 Customer Support

EnterWorks provides a full spectrum of customer support. Check your maintenance contract for details about the level of support purchased. A customer identification number will be issued the first time customer support is contacted. Keep this number for future reference when using the EnterWorks customer support service.

How to reach us	Comments
On the Web: https://enterworkssupport.zendesk.com	Our knowledge base includes solutions to common issues and is available 24 hours a day, 7 days a week.
Create a Support Request Ticket: https://enterworkssupport.zendesk.com/hc/en-us/requests/new	You can generate a Support Request ticket at any time. The Winshuttle Support team addresses Support Request tickets during our normal business hours: • 5 AM – 5 PM Pacific Time • Monday-Friday
Postal mail: Winshuttle Customer Support Team 19820 North Creek Pkwy #200 Bothell, WA 98011 USA	Please include your preferred contact information, as well as a description of your request.

2 Document Terminology

This document uses the following terminology:

Convention	Usage
"EnterWorks" and "Enable"	The EnterWorks Enable product is now referred to simply as "EnterWorks". Some system components and images in this document may still retain the name "Enable".
"master" and "slave"	In computing environments, the term "master" has historically been used to describe a process that controls other processes. The controlled processes have been referred to as "slaves". Winshuttle no longer finds these terms acceptable. We are in the process of replacing them with "controller" and "workers" in our documentation, displayed content, and underlying architecture. In the meantime, for further information or instructions on changing the terms in is played content, please contact your EnterWorks account representative.

3 Document Conventions

This EnterWorks document uses the following typographic conventions:

Convention	Usage
pathnames	Pathnames are shown with backslashes, as for Windows systems.
Courier New font	Denotes sample code, for example, Java, IDL, and command line information. May be used to denote filenames and pathnames, calculations, code samples, registry keys, path and file names, URLs, messages displayed on the screen. If <i>italicized</i> and in angle brackets (< >), it denotes a variable.
Calibri Font (bold)	When used in body text, it denotes an object, area, list item, button, or menu option within the graphical user interface; or a database name or database-related object. (Examples: the Save button; the Product tab; the Name field; the SKU repository) Can also be used to denote text that is typed in a text box. (Example: Type “ trackingNo ” in the Name field)
Blue underlined text	Click these active links to view additional information, steps, pages, chapters, or online resources.

4 EnterWorks API Services

The following sections describe what are included in the EnterWorks Web Services API and how to use them.

The EnterWorks API Services provides an extension to allow third party applications, not just limited to Java based applications, to integrate with the EnterWorks system. It provides most of the runtime support for the PIM management functionality. However, it is not an alternative to the web browser based user interface because it lacks most of the administrative capabilities such as user, security, repository mapping management etc.

The Enterworks Product Information Manager architecture is based on the Java 2 Enterprise Edition (J2EE) platform, and as such makes use of several J2EE platform technologies such as Enterprise Java Beans (EJBs), Java Servlets, and Java Messaging Service (JMS). Custom applications developed for EnterWorks must be written in Java, but do not necessarily require knowledge of the internal usage of J2EE technologies employed by EnterWorks. The EnterWorks public API available to custom applications are written entirely in Java and do not require JMS programming skills or knowledge of EJBs or Java Servlets. These Java APIs are actually libraries of compiled codes that can be used in EnterWorks.

5 Rest Services

With the representational state transfer (REST) style architecture, requests and responses are built around the transfer of representations of resources. Resources are identified by global IDs that typically use a uniform resource identifier (URI). Client applications use HTTP methods (such as GET, POST, PUT, or DELETE) to manipulate the resource or collection of resources. Refer to the SOAP API Services above for additional information on available service details

Generally, a GET method is used to get or list the resource or collection of resources, POST is used to create, PUT is used to update or replace, and DELETE is for removing the resource.

For example, GET `http://host/context/user/12345` gets the representation of the user with the ID 12345. The response representation could be an XML or JSON that contains the detailed user information.

Spring 4 (with Oauth2 support) is the framework for the EnterWorks RESTful Web Service layer.

5.1 Rest Service APIs

Enterworks is currently developing Restful Web Services for use with the EnterWorks system. Following is the EnterWorks 10's basic set of APIs. For a detailed list of the currently available EnterWorks APIs, see the EnterWorks 10 API Details document.

EPIM API

EPIM API description

Created by Enterworks

Codeset API : Code Set Controller

Show/Hide | List Operations | Expand Operations

GET	/api/codesets	Get code sets
GET	/api/codesets/{codeSetId}	Get code set by ID

Group and user API : Admin Controller

Show/Hide | List Operations | Expand Operations

GET	/api/groups	Get user groups
GET	/api/groups/javaApi	Get user groups with subset of data
GET	/api/groups/with-id	Get user groups with subset of data and group ID
GET	/api/groups/{groupId}	Get user group by ID

Item API : Item Controller

Show/Hide | List Operations | Expand Operations

POST	/api/items	Search Item
POST	/api/xmlitems	Search Item returns XML string

Repository API : Repository Controller

Show/Hide | List Operations | Expand Operations

GET	/api/repositories	Get repositories
GET	/api/repositories/{repositoryId}	Get repository by ID

Service API : Rest Service Controller

Show/Hide | List Operations | Expand Operations

POST	/api/login	login
GET	/api/logout	logout

5.1.1 Code Sets

GET /api/codesets Get code sets

Implementation Notes
Retrieve a list of code set the user can access.

Response Class (Status 200)
OK

Model | Model Schema

```
[
  {
    "codeSetId": 0,
    "codeSetLevelArray": [
      {
        "charPerLevel": 0,
        "codeSetId": 0,
        "codeSetLevelId": 0,
        "fillerValue": "string",
        "levelName": "string",

```

Response Content Type:

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
name	Classification	Codeset name	query	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

GET /api/codesets/{codeSetId} Get code set by ID

Implementation Notes
Retrieve metadata information for a code set with specified id.

Response Class (Status 200)
OK

Model | Model Schema

```
{
  "codeSetId": 0,
  "codeSetLevelArray": [
    {
      "charPerLevel": 0,
      "codeSetId": 0,
      "codeSetLevelId": 0,
      "fillerValue": "string",
      "levelName": "string",
      "levelNum": 0
    }
  ]
}
```

Response Content Type:

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
codeSetId	10000	Code set id	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

5.1.2 Group and Users

GET
/api/groups
Get user groups

Implementation Notes
Retrieve the list of groups the specified user belongs to.

Response Class (Status 200)
OK

Model
Model Schema

```
[
  {
    "createdBy": 0,
    "creationDatetime": "2016-08-12T19:53:23.920Z",
    "description": "string",
    "groupId": 0,
    "lastUpdatedBy": 0,
    "lastupdateDatetime": "2016-08-12T19:53:23.920Z",
    "name": "string",
    "uuid": "string"
  }
]
```

Response Content Type
application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
name	Niklas	User's name	query	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

GET
/api/groups/javaApi
Get user groups with subset of data

Implementation Notes
Retrieve the list of groups the specified user belongs to.

Response Class (Status 200)
OK

Model
Model Schema

```
[
  {
    "createdBy": 0,
    "creationDatetime": "2016-08-12T19:53:23.926Z",
    "description": "string",
    "groupId": 0,
    "lastUpdatedBy": 0,
    "lastupdateDatetime": "2016-08-12T19:53:23.926Z",
    "name": "string",
    "uuid": "string"
  }
]
```

Response Content Type
application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
name	Niklas	User's name	query	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

GET
/api/groups/with-id
Get user groups with subset of data and group ID

Implementation Notes

Retrieve the list of groups the specified user belongs to.

Response Class (Status 200)

OK

Model
Model Schema

```
[
  {
    "createdBy": 0,
    "creationDatetime": "2016-08-12T19:53:23.932Z",
    "description": "string",
    "groupId": 0,
    "lastUpdatedBy": 0,
    "lastUpdateDatetime": "2016-08-12T19:53:23.932Z",
    "name": "string",
    "uuid": "string"
  }
]
```

Response Content Type
application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
name	Niklas	User's name	query	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

GET
/api/groups/{groupId}
Get user group by ID

Implementation Notes

Retrieve the group with specified id.

Response Class (Status 200)

OK

Model
Model Schema

```
[
  {
    "createdBy": 0,
    "creationDatetime": "2016-08-12T19:53:23.937Z",
    "description": "string",
    "groupId": 0,
    "lastUpdatedBy": 0,
    "lastUpdateDatetime": "2016-08-12T19:53:23.937Z",
    "name": "string",
    "uuid": "string"
  }
]
```

Response Content Type
application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
groupId	10000	group id	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

Try it out!

5.1.3 Items

POST /api/items Search Item

Implementation Notes

Performs item search based on the search conditions for a given repository.

Returns a list of items that match the search conditions and item attributes are ordered according to the projection specification parameter. The object also contains the total number of matched items and total number of returned items.

Search conditions are either "AND" or "OR" joined by setting the isANDIndicator to true or false. However, one can use the "ADDITIONAL_WHERE_CLAUSE" as one of the search conditions to support user defined complex search criteria. Text search clauses and relational clauses (on snapshot columns and status columns) can be mixed. See examples below.

By using the "start" and "end" parameters, one can control the number of items to be return. This can be used as batch retrieval or result paging mechanism. Maximum number of items is limited to 1000.

Response Class (Status 200)

OK

Model | Model Schema

```
{
  "itemList": [
    {
      "attrLastUpdateBy": 0,
      "attrLastUpdateDatetime": "string",
      "attrList": [
        {
          "name": "string",
          "value": "string"
        }
      ]
    }
  ]
}
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
-----------	-------	-------------	----------------	-----------

Authorization Bearer

Authorization token in the format of 'Bearer [token]'

header string

body

```
{
  "andindicator": true,
  "end": 0,
  "projections": [
    "string"
  ],
  "repositoryId": 0
}
```

Search condition

body

Model | Model Schema

```
{
  "andindicator": true,
  "end": 0,
  "projections": [
    "string"
  ],
  "repositoryId": 0,
  "savedSetId": 0,
  "searchConditions": [
    {

```

Parameter content type: application/json

Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

POST

/api/xmlitems

Search Item returns XML string

Implementation Notes

Performs item search based on the search conditions for a given repository.

Returns an XML document containing a list of items that match the search conditions and item attributes are ordered according the projection specification parameter. The XML also contains the total number of matched items and total number of returned items.

Search conditions are either "AND" or "OR" joined by setting the isANDIndicator to true or false. However, one can use the "ADDITIONAL_WHERE_CLAUSE" as one of the search conditions to support user defined complex search criteria. Text search clauses and relational clauses (on snapshot columns and status columns) can be mixed. See examples below.

By using the "start" and "end" parameters, one can control the number of items to be return. This can be used as batch retrieval or result paging mechanism. Maximum number of items is limited to 1000 if XML fields are retrieved. If projections are all relational attributes including status attributes, then the maximum number of items can be higher than 1000. It requires large amount of memory to prepare and transport big XML document through Web Services call, so one should be extremely careful about the amount of data the search can produce.

ePIM provides utility classes for easy XML manipulation. Client can use the com.enterworks.epim.webservice.ItemList class to map the XML doc back to Java for data manipulation.

Use getRepositoryAttributeList() to retrieve a list of attributes for the repository - these will be the attributes you may use in the itemData key-value pairs. See the Attribute class for further description of XML and non-XML attributes.

Response Class (Status 200)

Response Content Type:

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
body		Search condition	body	Model

Model Schema

```

{
  "andIndicator": true,
  "end": 0,
  "projections": [
    "string"
  ],
  "repositoryId": 0,
  "savedSetId": 0,
  "searchConditions": [
    {

```

Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

5.1.4 Repositories

GET
/api/repositories
Get repositories

Implementation Notes
Retrieve the list of repositories the current user can access.

Response Class (Status 200)
OK

Model | **Model Schema**

```
[
  {
    "damUseInd": 0,
    "dataPoolUserName": "string",
    "description": "string",
    "masterIndicator": true,
    "masterRepositoryId": 0,
    "name": "string",
    "ownerTradingPartnerGLN": "string",
    "ownerTradingPartnerName": "string",
  }
]
```

Response Content Type: application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
name		Repository name	query	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

GET
/api/repositories/{repositoryId}
Get repository by ID

Implementation Notes
Retrieve the repository by id.

Response Class (Status 200)
OK

Model | **Model Schema**

```
{
  "damUseInd": 0,
  "dataPoolUserName": "string",
  "description": "string",
  "masterIndicator": true,
  "masterRepositoryId": 0,
  "name": "string",
  "ownerTradingPartnerGLN": "string",
  "ownerTradingPartnerName": "string",
  "permissions": "string",
}
```

Response Content Type: application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
repositoryId	10000	Repository id	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
401	Unauthorized		
403	Forbidden		
404	Not Found		

5.1.5 Services

POST

/api/login

login

Response Class (Status 200)

OK

Model | Model Schema

```

{
  "externalSessionInfo": "string",
  "language": "string",
  "sessionId": 0,
  "userId": 0
}

```

Response Content Type

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string
name	(required)	name	query	string
pwd	(required)	pwd	query	string
allowexpire	(required)	allowexpire	query	string
language		language	query	string
externalSessionInfo		externalSessionInfo	query	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
201	Created		
401	Unauthorized		
403	Forbidden		
404	Not Found		

GET

/api/logout

logout

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Bearer	Authorization token in the format of 'Bearer [token]'	header	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	OK		
401	Unauthorized		
403	Forbidden		
404	Not Found		

5.2 Calling the Rest Services

The ePIM RESTful Web Service can be called in 2 different ways:

- **URL:** <http://localhost:8090/webcm/rest/groups>

It will return JSON objects for all groups in the format of:

```
[{      "name": "01_Peer_Training",
  "description": ""  },
{      "name": "22deb",
  "description": "e1"
}]
```

- **Java API:** `public GroupDTO[]
getGroupList(TokenDTO token) throws
RemoteException`

It will return a list of `GroupDTO` objects containing the same data as when calling from URL.

Base URL is : <http://localhost:8090/webcm/rest>

The following are sample endpoints with details for review:

- <http://localhost:8090/webcm/rest/groups> - to retrieve all groups, group by name, or group by group_id
- <http://localhost:8090/webcm/rest/codesets> - retrieve all codesets, codeset by name, or codeset by code_set_id
- <http://localhost:8090/webcm/rest/repositories> - retrieve all repositories, repository by name, or repository by repository_id
- <http://localhost:8090/webcm/rest/items> – to search for items using search conditions

5.3 Rest Service Sample Code

These endpoints can be easily accessed via java with the following:

```
SessionDTO session = null;

token = restFacade.login("epim-client", "epim-client",
userName, password, null, null);

CodeSetDTO[] objList = restFacade.getCodeSetList(token);

CodeSetDTO codeSet = restFacade.getCodeSet(token, id);
```

```
GroupDTO[] objList = restFacade.getGroupList(token);
GroupDTO obj = restFacade.getGroupById(token, id);
RepositoryDTO[] repoList = restFacade.getRepoList(token);
RepositoryDTO obj = restFacade.getRepositoryByName(token,
name);
String xmlItem = restFacade.searchItemBySavedSet(token,
repo, savedSet, searchConditions, sortConditions,
isANDIndicator, projections, start, end)
```

6 SOAP Services

The interface provides the venue by which independent systems communicate with each other. The EnterWorks interface is programming language independent and Operating System independent.

The EnterWorks Web Services APIs are described in the `pim.wsdl` (Web Service Description Language). The EnterWorks installer already bundles the necessary java class files during the installation; thus, there is no need to re-generate the java client from the WSDL file. However, this is quite acceptable to do so. See the “How to Use” section for details on how to use the pre-bundled java classes. Non-Java applications can use the WSDL file as the basis to generate their corresponding client access commands in their native languages.

The core interface in the EnterWorks Web Services API is called “Server”. Others are simply supporting data types that are used to either support the SOAP protocol or used as information holders.

The supporting data types are as follows:

- `AttrCodeSetMapping`
- `AttrData`
- `Attribute`
- `CodeSet`
- `CodeSetDetail`
- `EJBServer`
- `EnableTreeNode`
- `Group`
- `Item`
- `ItemHistory`
- `ItemHistoryDiff`

- ItemList
- ItemState
- KeyValuePair
- Media
- MediaGroup
- MediaReference
- Package-frame
- Package-summary
- Package-tree
- Repository
- RepositoryGroup
- RepositoryViewMapping
- SearchCondition
- Server
- ServerSoapBindingImpl
- Session
- Subscription
- TextSearchConfig
- TextSearchConfigAttr
- TradingPartner
- TransactionStatus
- UserPreference

6.1 Services

The EnterWorks Web Services API are divided into three major functional groups:

- Metadata retrieval
- Repository/item management
- Tools

6.1.1 Metadata Retrieval

- `createSubscription` - Creates a subscription in the server by a given `Subscription` object.
- `deleteSubscription` - Deletes a subscription object from the database.
- `exportRepository` - Perform export for a particular repository with specified `userpreference`.
- `getCodeSet` - Retrieves the metadata information for the code set.
- `getCodeSetList` - Retrieves a list of code sets the user can access.
- `getCodeSetDetail` - Retrieves the code set detail including code, code description, parent code, and other information.
- `getLowerLevelCodeSetDetail` - Retrieves the lower level code set detail for a particular code set and a code.
- `getCustomerList` - Retrieves a list of customers. The return list construction depends on the user's access rights.
- `getCustomerListBySupplierGLN` - Retrieves a list of customers for a particular supplier.
- `getSupplierList` - Retrieves a list of suppliers. The return list construction depends on the user's access rights.
- `getGroupList` - Retrieves the list of groups the current user belongs to.
- `getRepositoryByName` - Retrieve the repository by name.
- `getRepositoryList` - Retrieves the list of repositories the current user can access.
- `getRepositoryAttributeList` - Retrieves the attribute list for a particular repository.
- `getUserPreferenceList` - Retrieves the user preferences the current user can access.
- `getUserPreferenceAttributeList` - Retrieve attributes for a particular user preference.
- `getJobStatus` - Retrieves the job status.
- `getCodeSetDetailByCode` - Retrieve code set detail including code, code description and parent code etc.

- `getCodeSetDetailByName` - Retrieve code set detail including code, code description and parent code etc.
- `getAttrCodeSetMapping` – Retrieves the attribute and code set mapping for a particular repository.
- `getCustomerList` - Retrieves a list of customers.
- `getCustomerListBySupplierGLN` - Retrieves a list of customers for a particular supplier.
- `getEnableTree` - Retrieves `enable` tree nodes.
- `getItemHistory` - Returns a single `ItemHistory` object based on its `itemHistoryId` (obtained from `getRepositoryItemHistoryList()`).
- `getItemHistoryAttrDiffList` - Returns a list of `ItemHistoryDiff` objects for two `ItemHistory` objects.
- `getItemHistoryList` - Returns an array of `ItemHistory` objects for the specified `repositoryId` and `itemId` based on the date and `userLogin` search parameters.
- `getLowerLevelCodeSetDetailByName` - Retrieve lower level code set detail for a particular code set and a code.
- `getMediaContent` - Retrieves media content for a particular media object identified by media ID.
- `getMediaGroupList` – Retrieves Media Groups.
- `getMediaReferenceList` - Retrieves references for a particular media object identified by media ID.
- `getRepositoryGroupList` - Retrieves the repository group list.
- `getRepositoryListByGroupName` - Retrieves a list of repository objects for a particular repository group.
- `getSubscriptionList` - To retrieve an array of `Subscription` objects.
- `getSupplierList` - Retrieves a list of suppliers.
- `getTextSearchConfigList` - Retrieves a list of `TextSearchConfig` objects for a particular repository.
- `getUserList` - Retrieve the list of users the current user has access to see.

- `getUserPreferenceByName` - Retrieve user preference by name.
- `getViewMappingList` - Retrieves an array of `RepositoryViewMapping` objects for a repository.
- `subscribeSubscription` - To subscribe the subscription.
- `searchCodeSetDetail` - Search the `codeSet` detail using search string and search method and return an array of `CodeSetDetail` Objects.
- `setDataPoolSyncOutStatus` - Update sync out history status based on message data pool response message identifier.
- `setDataPoolSyncOutStatusByFile` - Update sync out history status based on data pool response message file name.
- `syncInRepository` - Perform sync in operation for a particular repository.
- `unsubscribeSubscription` - To remove the subscription.
- `updateSubscription` - Updates a subscription object with the given `Subscription` object.
- `updateSubscriptionState` - Updates the subscription state of a given subscription object.
- `validateRepositoryOption` - Perform validation for a particular repository with specified `optionCode`.

6.1.2 Repository/Item Management

- `abortJob` - Abort a job with the given `jobId`.
- `createMedia` - Creates the media object.
- `createMediaByURL` - Creates media object using URL.
- `deleteItem` - Deletes a particular item.
- `deleteMedia` - Deletes a media object identified by media ID.
- `getItem` - Retrieves the item detail.
- `getItemState` - Retrieves the item state.
- `createItem` - Creates a new item for a particular master repository.
- `delistItem` - Mark item delist. `DeleteItem` - Deletes a particular item.

- `updateItem` - Updates the item data
- `updateItems` - Updates items with search conditions.
- `searchItem` - Performs item search based on the search conditions for a given repository.
- `searchItemByConfig` - Performs item search based on their configurations.
- `searchItemByKeyword` - Performs item search using keywords.
- `searchItemByRepository` - You can search items in a particular repository based on simple search conditions such as "AND" or "OR" joined.
- `searchItemByUserPreference` - You can search items for a particular user preference.
- `searchMedia` - Performs media object search.
- `setItemState` - Updates the item state.
- `setDataPoolResponseStatus` – Updates the item status based on data pool response message identifier.
- `setDataPoolReponseStatusByFile` – Updates the item status based on data pool response message file name.
- `setDataPoolResponseStatusByPrimKey` - Update item status based on data pool response message primary key.
- `setDataPoolReponseStatusByTransaction` - Updates the item status based on data pool response message identifier and transactions.
- `validateRepository` - Performs validation for a particular repository.
- `syncOutRepository` - Performs the sync out operation for a particular repository.
- `createCodeSetDetail` - Creates a code set detail record for a particular code set.
- `deleteCodeSetDetail` - Delete a code detail record for a particular code set.
- `isValidParent` - Will interrogate the `hasErrorInd` of the parent item and all of its linked children with the given link relationship and return the results.

- `updateCodeSetDetail` - Updates a code set detail record for a particular code set.
- `updateItemsByKeywordSearch` - Update items using keyword using text search.
- `updateItemTriggerOption` - Updates item data and status.
- `updateMedia` - Updates the media object.
- `updateMediaByURL` – Updates the media object by URL.
- `updateMediaContent` - Updates media content only without the associated meta data.
- `updateMediaContentByURL` - Updates media content only, without the associated metadata, using content URL.
- `updateMediaMetaData` - Updates the media metadata only and not the content.
- `getParentRepositoryLinkList` - Retrieves an array of `RepositoryLink` objects for a parent repository.
- `getChildRepositoryLinkList` - Retrieves an array of `RepositoryLink` objects for a child repository.

6.1.3 Tools

- `Login` - Logs into e-PIM server to obtain a valid session.
- `Logout` - Logs out from e-PIM server and terminates user session.
- `ChangeUserPassword` - Changes the password for the login user.
- `loginByGID` - Log into EnterWorks server using global ID.

The method by which these available Web service functions can be invoked is provided in the EnterWorks Integrated Web Services Javadoc, which provides the necessary documentation for EnterWorks Web Services. The Web Services Javadoc is the generated zip file containing the API documentation of the EnterWorks automated resources accessed via the Internet. It is extracted from the comments in the Java source code and made available in HTML format.

For more information on Java Web Services methods and calls, particularly metadata retrieval, repository/item management and tools, refer to the `Enable Web Services`

Javadoc. To access the Javadoc, navigate to the directory below and locate the `wsdoc` file:

```
C:\Enterworks\Enable\documents
```

6.2 SOAP Service Sample Code

Java source codes are compiled into Java Class files that can be loaded to execute Methods needed to perform the different functions and capabilities of the EnterWorks application, particularly the Web Services component. Presented below is the sample code or script that executes some EnterWorks methods described above.

The sample below shows how to perform sync in, sync out, and a validation method call using the Enable 7 Web Services API:

```
package com.enterworks.Enable.test.webservice;

import java.util.ArrayList;
import com.enterworks.Enable.test.webservice.KeyValuePair;
import com.enterworks.Enable.test.webservice.Repository;

/* Demonstrate sync in, sync out and validation method calls using Enable Web
services API
*/
public class PimWSCClientSample {
    private static String hostURL =
        "http://sullely:8888/webcm/services/server";

    private static String usage = "Usage: java PimWSCClient syncout|validation
<repository name>";

    public static void main(String[] args) {
        com.enterworks.Enable.test.webservice.ServerSoapBindingStub binding = null;
        if (args.length < 2) {
            System.err.println(usage);
            return;
        }
        if (args[0].equalsIgnoreCase("syncout")) {
            System.out.println("Sync out repository:" + args[1]);
        } else if (args[0].equalsIgnoreCase("validation")) {
            System.out.println("Validation repository:" + args[1]);
        } else {
            System.err.println(usage);
            return;
        }
        try {
            // login first

            com.enterworks.Enable.test.webservice.ServerServiceLocator loc = new
                com.enterworks.Enable.test.webservice.ServerServiceLocator();
            binding = (com.enterworks.Enable.test.webservice.ServerSoapBindingStub)
```

```

    loc.getServer(new java.net.URL(hostURL));
    com.enterworks.Enable.test.webservice.Session mySession = login(binding);

    // validate result

    System.err.println("login session:" +
        mySession.getSessionId());
    System.err.println("login user:" + mySession.getUserId());

    // retrieve repository list

    Repository[] repList = null;
    Repository rep = null;
    repList = binding.getRepositoryList(mySession);

    if (repList == null) {
        System.out.println("No repository to retrieve.");
        return;
    }
    for (int i = 0; i < repList.length; i++) {
        if (args[1].equalsIgnoreCase(repList[i].getName())) {
            System.out.println("Find matching repository:" + args[1]);
            rep = repList[i];
            break;
        }
    }
    if (rep == null) {
        System.err.println("No matching repository found.");
        return;
    }
    if (args[0].equalsIgnoreCase("syncout")) {
        binding.syncOutRepository(mySession, rep, false);
    } else if (args[0].equalsIgnoreCase("validation")) {
        binding.validateRepository(mySession, rep);
    } else {
        System.err.println(usage);
        return;
    }

    //logout

    binding.logout(mySession);
    } catch (javax.xml.rpc.ServiceException jre) {
        if (jre.getLinkedCause() != null) jre.getLinkedCause().printStackTrace();
        throw new junit.framework.AssertionFailedError("JAX-RPC
        ServiceException caught: " + jre);
    } catch (Exception e) {
        System.err.println(e.getMessage());
    }
    }
    /*
    * common login wrapper routine.

```

```

    */
    public static com.enterworks.Enable.test.webservice.Session
login(com.enterworks.Enable.test.webservice.ServerSoapBindingStub binding) {
    try {
        com.enterworks.Enable.test.webservice.Session value = null;
        value = binding.login("system", "system", true, null);
        return value;
    } catch (Exception e) {
        System.out.println("Can't login." + e.getMessage());
    }
    return null;
}
/* login */
/*
 * Update the item attribute values.
 * Please refer to the java doc for API description.
 */
public static void updateItem() {
    com.enterworks.Enable.test.webservice.ServerSoapBindingStub binding = null;
    try {
        // login first

        com.enterworks.Enable.test.webservice.ServerServiceLocator loc = new
com.enterworks.Enable.test.webservice.ServerServiceLocator();
        binding = (com.enterworks.Enable.test.webservice.ServerSoapBindingStub)
loc.getServer(new java.net.URL(hostURL));
        com.enterworks.Enable.test.webservice.Session mySession = login(binding);

        // validate result

        System.err.println("login session:" +
mySession.getSessionId());
        System.err.println("login user:" + mySession.getUserId());
        java.lang.String value = null;

        // should do repository retrieval instead of doing the following
        // this code assumes the repository is known.

        Repository rep = new Repository();
        rep.setRepositoryId(1);
        rep.setMasterIndicator(true);
        ArrayList l = new ArrayList();

        // again, should retrieve attribute list instead of hard code.

        KeyValuePair kvp1 = new KeyValuePair("F_11", "123");
        KeyValuePair kvp2 = new KeyValuePair("F_22", "abc");
        l.add(kvp1);
        l.add(kvp2);
        binding.updateItem(login(binding), rep, 1, (KeyValuePair[]) l.toArray(new
KeyValuePair[l.size()]));
        System.out.println("done.");
    }
}

```

```

    } catch (Exception e) {
        System.err.println(e.getMessage());
    }
}
/*
 * Update item data pool response.
 * Please read the java doc for API description.
 */
public static void updateItemStatus() {
    com.enterworks.Enable.test.webservice.ServerSoapBindingStub binding =
        null;

    try {

        // login first

        com.enterworks.Enable.test.webservice.ServerServiceLocator loc = new
com.enterworks.Enable.test.webservice.ServerServiceLocator();
        binding = (com.enterworks.Enable.test.webservice.ServerSoapBindingStub)
loc.getServer(new java.net.URL(hostURL));
        com.enterworks.Enable.test.webservice.Session mySession = login(binding);

        // validate result

        System.err.println("login session:" +
            mySession.getSessionId());
        System.err.println("login user:" + mySession.getUserId());
        java.lang.String value = null;

        // assume the message id is "UCCNET_RESPONSE1"
        // read java doc to see the actual status code meaning.

        binding.setDataPoolResponseStatus(login(binding),
            "UCCNET_RESPONSE1", 0, "Sync from UCCNet Ok.");
    } catch (Exception e) {
        System.err.println(e.getMessage());
    }
}
public static void updateItemResponse() {
    com.enterworks.Enable.test.webservice.ServerSoapBindingStub binding = null;

    try {

        // login first

        com.enterworks.Enable.test.webservice.ServerServiceLocator loc = new
com.enterworks.Enable.test.webservice.ServerServiceLocator();
        binding = (com.enterworks.Enable.test.webservice.ServerSoapBindingStub)
loc.getServer(new java.net.URL(hostURL));
        com.enterworks.Enable.test.webservice.Session mySession = login(binding);

        // validate result

```

```
System.err.println("login session:" +
    mySession.getSessionId());
System.err.println("login user:" + mySession.getUserId());

// should retrieve repository list first
// this code assume it exists.

Repository rep = new Repository();
rep.setRepositoryId(1010946);
rep.setMasterIndicator(true);

// see java doc for actual status code meaning

binding.setItemState(login(binding), rep, 1011019, 2, true, 3, true, 1, false, 0,
true);
} catch (Exception e) {
    System.err.println(e.getMessage());
}
}
}
```