

INQUISIQ^{r3}

Unleash the Power of e-Learning

API Reference Manual

Revision 1.6

October 2011 Edition

© 2002-2011 ICS Learning Group

Disclaimer

ICS Learning Group makes no representations or warranties with respect to the contents or use of this manual, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. ICS Learning Group reserves the right to revise this publication and to make changes to its content at any time, without obligation to notify any person or entity of such revisions or changes.

Further, ICS Learning Group makes no representations or warranties with respect to any ICS Learning Group product, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. ICS Learning Group reserves the right to make changes to any and all parts of ICS Learning Group products at any time, without obligation to notify any person or entity of such changes.

Trademarks

Inquisiq, Inquisiq R3, Inquisiq Studio, Intelligent Streaming Video (ISV), and ICS Learning Group are registered trademarks of Interactive Communications Solutions Group, Inc.

Other brands and their products are trademarks or registered trademarks of their respective holders.

Product Support

If you have purchased a support package and have any questions during the use of Inquisiq R3™ that are not addressed in this guide, please visit our support site at:

<http://support.icslearninggroup.com/>

Or contact us at:

ICS Learning Group
8221 Ritchie Highway
Suite 303
Pasadena, MD 21122

<http://www.icslearninggroup.com>

Table of Contents

API Overview	4
Configuring API Access	5
Enabling Access in the Control Panel	5
API Account Configuration	6
The XML Request Format	7
The <serv_request> Element	7
The <head> Element	7
The <body> Element	8
The XML Response Format	9
The <serv_response> Element	9
The <head> Element	9
The <body> Element	10
Sample API Request Code	11
API Method Reference	13
Method: getUser	14
Method: getSSOToken	16
Method: getCourse	19
Method: getEnrollment	21
Method: saveUser	23
Method: saveEnrollment	28

API Overview

The Inquisiq API is provided as a value-add to enable customers to integrate their third-party applications with the LMS. Due to the range in variation in third-party applications, support for this API is limited:

- If the Inquisiq feature, method, dataset, etc. is not listed in this document, it is not available via the API.
- Per above, Reporting features/methods are not listed and, thus, not supported. We understand this is a popular request but the possibility for unfavorable system impact is too great if an improper or excessive request is made. There are other methods to obtain automated reports (i.e. automated subscriptions).
- Similar to any LMS customization requests, any actual coding assistance is not covered by support and must be scoped as a billable project (1 hour minimum).
- Inquisiq Support is with our direct customers. You may request us to work directly with your customer developers, support, or representatives, but such contact will be scoped as a billable project (1 hour minimum).

In sum, ICS will support clarification of what the API offers access to and verbal examples of how people use the API. However, any actual coding or detailed technical assistance will require a billable project to be initiated.

Configuring API Access


In order to access the API, it must be first enabled for your account and then configured. If the API icon on the Administrator Menu appears to be disabled, then it has not been enabled for your account and must be done so by the server administrator.

Please note that API functionality exists only in Inquisiq R3™ Build 2.10.0315 and later despite the inclusion of the API options in the Control Panel and Administrator Menu on all versions.

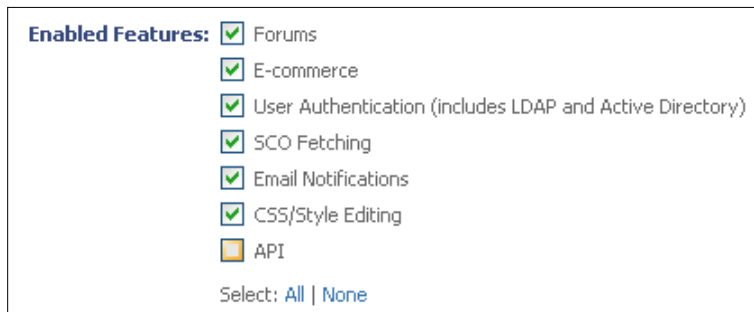
Enabling Access in the Control Panel

Log in to the Control Panel, click the Accounts icon and find the account for which you would like to enable the API.



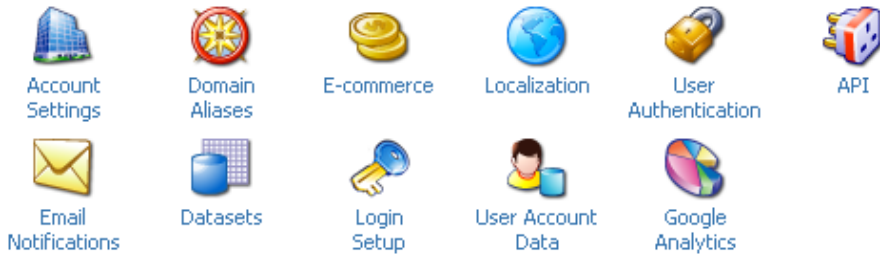
Click the View icon () and select the Settings tab.

Place a check in the API box under Enabled Features and click Save Changes. The API setup interface will now be available within the LMS account.



API Account Configuration

From the Administrator Menu, click the API icon to access the API configuration interface.



Mark the “On” selection and enter an Authorization Key. This key functions as a password for the API on your account; every API request made must include your account name and this key in order to be successfully executed.

*** API Access:** On Off

*** Authentication Key:** (10 character minimum)

Require HTTPS: When checked, only API requests made over SSL will be accepted.

URL: API requests should be send to this URL:
https://sample.inquisiq3.com/_gateway/api/

If your server has an SSL certificate installed, it is recommended that you check the Require HTTPS option. Enabling this option will ensure that both your API requests (containing your API authorization key) as well as the API responses (which can contain personal user data as well as passwords) are encrypted during transmission.

The API URL is noted on screen. The appropriate path for each method is noted in this document and should be combined with this URL to create the full URL for each request.

Click Save Changes.

The XML Request Format

Every API command request contains some common elements that encapsulate the request as well as elements that are used to identify the LMS account and security context of the request.

The <serv_request> Element

The root of the XML request should always be a <serv_request> element that contains a <head> element and a <body> element.

```
<serv_request>
  <head>
  ...
  </head>
  <body>
  ...
  </body>
</serv_request>
```

The <head> element will contain information about the request itself while the <body> element will contain the instructions for the API.

The <head> Element

The <head> element is required and must be the first element within the <serv_request> element. It should contain one <securityContext> element with the account and security information contained within.

```
<head>
  <securityContext>
  ...
  </securityContext>
</head>
```

Within the <securityContext> element, you should include an <account> element and a <key> element. The <account> element should contain the actual LMS account name. This is the login name that you selected when creating the LMS account and identifies your LMS on the server. The <account> element should appear as follows:

```
<account>www</account>
```

The <key> element should contain the security key that has been set up on the server to allow access to the API. This is not your account password. The <key> element should appear as follows:

```
<key>1234567890</key>
```

If you do not provide a correct account/key combination, you will not be able to access the API.

A complete <head> element would appear in the following format:

```
<head>
  <securityContext>
    <account>www</account>
    <key>1234567890</key>
  <securityContext>
</head>
```

The <body> Element

The <body> element is required and must be the second element within the <serv_request> element.

The <body> element is where the actual instructions for the API will reside and its content will vary depending on the API command that you are executing.

The XML Response Format

When the API has completed processing the XML Request, its response will consist of similarly formatted XML.

The <serv_response> Element

Just like the <serv_request> element, the <serv_response> element encapsulates the entire API response and will also contain a <head> element and a <body> element.

```
<serv_response>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</serv_response>
```

The <head> element will contain information regarding the status of the request and the <body> element will contain the actual response data.

The <head> Element

Within the <head> element you will find a <status> element. You should always check the <status> element to determine if the API processed your request. If no errors were encountered, the <status> element will contain the value "success"

```
<head>
  <status>success</status>
</head>
```

If errors occurred during the processing of the XML request, the <status> element will contain a value of "fail".

Following the <status> element, you may find an <items> element that contains information about the status of the request. When the request has failed, details regarding the specific errors encountered will be found here.

In some cases a successful XML request may also return status messages in the <items> element however this is not the norm.

```
<head>
  <status>fail</status>
  <items>
    ...
  </items>
</head>
```

Each error entry within the <items> element will be contained within an <item> element that contains the elements shown below. The <description> element will contain a detailed description of the error encountered. All other elements may or may not contain data.

```
<items>
  <item>
    <code>-1</code>
    <heading></heading>
    <description>Authorization key is incorrect.</description>
    <routine></routine>
    <icon></icon>
    <priority></priority>
  </item>
</items>
```

The <items> and <item> elements should always be checked for information regarding an error when the <status> element contains “fail”.

The <body> Element

Following the <head> element is the <body> element which contains the actual response data. The format of this data will vary and is specific to the API command being executed. Refer to the API command reference below for the specific format of each API command response.

Sample API Request Code

Each programming language will vary slightly from this example, however since the LMS is a Microsoft OS-based system, we will show a basic VBScript example that uses the Microsoft XML-HTTP Request object.

Formulate your XML request as appropriate for method that you will be invoking and ensure that the request method is "POST". Your XML request should be sent as a name/value pair with the name "xml". Refer to the example below.

Important Note! In most cases, your XML string will contain characters that require URL encoding when posting to the API scripts. Be sure to use an appropriate encoding method as provided by your chosen programming language. Refer to the examples below.

Example (ASP Classic VBScript):

```
dim obj
dim xmlrequest
dim xmlresponse

' formulate the XML request here

xmlrequest = "<serv_request>...</serv_request>"

set obj = Server.CreateObject("Microsoft.XMLHTTP")

obj.open "POST", "http://[domain_name]/_gateway/api/[filename].asp", false
obj.setRequestHeader "Content-Type", "application/x-www-form-urlencoded"
obj.send "xml=" & Server.URLEncode(xmlrequest)

xmlresponse = obj.responseText

set obj = nothing
```

Example (ASP.net C#):

```
string url = " http://[domain_name]/_gateway/api/[filename].asp";

// formulate the XML request here

string xmlrequest = "<serv_request>...</serv_request>";

NameValueCollection nvc = new NameValueCollection();
nvc.Add("xml", Server.UrlEncode(xmlrequest));

WebClient client = new WebClient();
byte[] byteresponse = client.UploadValues(url, nvc);

string xmlresponse = client.Encoding.GetString(byteresponse);
```

When the HTTP response in this sample is executed and completed, the variable “xmlresponse” will contain the XML response from the API.

The XML response will vary depending on the API method being invoked. Each response format is explained in detail later in this document.

API Method Reference

A complete list of available API methods follows.

XSD schemas are provided for all XML requests listed in this reference.

All dates and/or times in all XML requests should be provided in ISO date or ISO datetime format as specified in the XSD schema and normalized to GMT. Additionally, all dates and times that are included in XML responses will be in ISO datetime format and normalized to GMT.

NOTE: Some 'special characters' may need to be 'escaped' or 'URL encoded' in order to pass properly to the API interface. Whether this is the case or not depends on what you data you're passing. If you get 'fail' status, check your data values and URL encode any special characters (i.e. try using "%40" instead of "@").

In fact, using ASP's "Server.URLEncode" method is often the best approach by forming the XML string in ASP then use the above method to 'URL encode' that value. However, be sure to **ONLY** encode the XML string itself – don't encode the "xml=<serv_request>...</serv_request>" portion of your code!

You can find a URL encoding reference here:

http://www.w3schools.com/tags/ref_urlencode.asp

There's a handy URL Encoder here (we cannot vouch for its accuracy or uptime):

<http://meyerweb.com/eric/tools/dencoder/>

Method: getUser

URL: /_gateway/api/getUser.asp

XSD Schema: /_gateway/api/schema/getUser.xsd

Description

Use this method to retrieve user account data from the LMS.

The XML Request

Your XML request should contain the search criteria that will be used to find the desired user account(s). Each criterion shall be included as an element contained within the <body> element of your request.

Allowed criteria and their descriptions are below. You may include each criterion zero or more times. You may also specify a wildcard character within criteria listed as type *string* by inserting the % character.

- <id>
Type: integer
Returns the user account whose internal identifier matches the value provided.
- <username>
Type: string
Returns the user account(s) whose username(s) match the value provided.
- <firstName>
Type: string
Returns the user account(s) whose first name(s) match the value provided.
- <lastName>
Type: string
Returns the user account(s) whose last name(s) match the value provided.
- <email>
Type: string
Returns the user account(s) whose email(s) match the value provided.
- <employeeID>
Type: string
Returns the user account(s) whose employee ID(s) match the value provided.
- <search>
Type: string
Returns the user account(s) whose username(s), first name(s), last name(s) or email(s) match the value provided.

Example:

```
<serv_request>
<head>
  <securityContext>
    <account>www</account>
    <key>1234567890</key>
```

```

        </securityContext>
</head>
<body>
    <username>jsmith</username>
    <username>johnsmith</username>
    <email>jsmith@%</email>
    <email>johnsmith@%</email>
</body>
</serv_request>

```

The XML Response

The <body> element of the response will contain a single <users> element containing zero or more <user> elements; each containing data from a user account that matched your search criteria.

Requests sent with no criteria will return *all* user accounts. Otherwise, user accounts whose data matches at least one of each criterion will be returned.

Using the above example, all user accounts that match at least one of the <username> criteria *and* at least one of the <email> criteria will be included in the response.

Example:

```

<serv_response>
<head>
    <status>success</status>
</head>
<body>
    <users>
        <user id="1001" enabled="True" mustChangePassword="False">
            <username>jsmith</username>
            <firstName>John</firstName>
            <middleName></middleName>
            <lastName>Smith</lastName>
            <email>jsmith@mycompany.com</email>
            <company>My Company</company>
            <address>123 Main Street</address>
            <city>New York</city>
            <province>New York</province>
            <postalCode>12345</postalCode>
            <country>US</country>
            <phonePrimary>123-456-7890</phonePrimary>
            <phoneWork></phoneWork>
            <phoneFax></phoneFax>
            <phoneHome></phoneHome>
            <phoneMobile></phoneMobile>
            <phonePager></phonePager>
            <phoneOther></phoneOther>
            <employeeID>1234567-89</employeeID>
            <department></department>
            <division></division>
            <region></region>
            <jobTitle>President</jobTitle>
            <jobClass>Management</jobClass>
        </user>
    </users>
</body>
</serv_response>

```

```

        <hireDate></hireDate>
        <termDate></termDate>
        <language></language>
        <ssn>123456789</ssn>
        <gender>m</gender>
        <race></race>
        <dob>1960-06-01 00:00:00</dob>
        <created>2010-01-01 15:29:56</created>
        <expires>2015-12-31 00:00:00</expires>
        <identifier></identifier>
        <supervisor>
            <user id="275" enabled="True" mustChangePassword="False">
                ... [data truncated for space]
            </user>
        </supervisor>
        <field00></field00>
        <field01></field01>
        <field02></field02>
        <field03></field03>
        <field04></field04>
        <field05></field05>
        <field06></field06>
        <field07></field07>
        <field08></field08>
        <field09></field09>
    </user>
    <user id="9876" enabled="True" mustChangePassword="False">
        ... [data truncated for space]
    </user>
    <user id="9999" enabled="True" mustChangePassword="False">
        ... [data truncated for space]
    </user>
</users>
</body>
</serv_response>

```

The element names of the properties contained within the <user> element remain as shown above even if you have re-labeled the property on the User Account Data screen within the LMS.

Note that passwords are encrypted and write-only; therefore they are not included in the XML response.

Note, if the user account has a supervisor assigned; a <user> element containing data for the supervisor's user account will be returned in the user's <supervisor> element.

Note that because the supervisor element contains the user element, which can in turn contain a supervisor element with another user element, this node of the response can potentially be many levels deep. The number of levels does not have any imposed limit other than that which is created by the administrator(s) of the LMS instance.

Method: [getSSOToken](#)

URL: [/_gateway/api/getSSOToken.asp](#)

XSD Schema: [/_gateway/api/schema/getSSOToken.xsd](#)

Description

Use this method to generate a single-sign-on token. This token can then be used to seamlessly bypass the normal sign-in process.

The XML Request

Your XML request specifies which user account the token shall belong to.

You must use exactly one element within the <body> element of your request; either a single <id> element or single <username> element to identify a single user account. Wildcards are not allowed.

- <id>
Type: integer
Specifies the user account whose internal identifier matches the value provided.
- <username>
Type: string
Specifies the user account whose username exactly matches the value provided.

Example:

```
<serv_request>
<head>
  <securityContext>
    <account>www</account>
    <key>1234567890</key>
  </securityContext>
</head>
<body>
  <username>jsmith</username>
</body>
</serv_request>
```

The XML Response

The XML Response contains the single-sign-on token for the user specified in your request.

Redirect the user's browser to **http://[domain_name]/_gateway/sso/?token=X** to complete the single sign on process. Replace "[domain_name]" with the domain name of your LMS account and "X" with the token contained within the XML response.

The token is valid for 15 seconds once it has been generated.

Example:

```
<serv_response>
<head>
  <status>success</status>
</head>
<body>
  <token>HIAGE86MK6M8CME6FHA9L8A9E</token>
</body>
</serv_response>
```

Method: `getCourse`

URL: `/_gateway/api/getCourse.asp`

XSD Schema: `/_gateway/api/schema/getCourse.xsd`

Description

Use this method to retrieve course data from the LMS.

The XML Request

Your XML request should contain the search criteria that will be used to find the desired course(s). Each criterion shall be included as a element contained within the `<body>` element of your request.

Allowed criteria and their descriptions are below. You may include each criterion zero or more times. You may also specify a wildcard character within criteria listed as type *string* by inserting the `%` character.

- `<id>`
Type: integer
Returns the course whose internal identifier matches the value provided.
- `<name>`
Type: string
Returns the course(s) whose name(s) match the value provided.
- `<code>`
Type: string
Returns the user course(s) whose code(s) match the value provided.
- `<search>`
Type: string
Returns the course(s) whose name(s), code(s) or short description(s) match the value provided.

Example:

```
<serv_request>
<head>
  <securityContext>
    <account>www</account>
    <key>1234567890</key>
  </securityContext>
</head>
<body>
  <name>%Introduction%</name>
  <code>PM-%</code>
  <code>AS-%</code>
</body>
</serv_request>
```

The XML Response

The <body> element of the response will contain a single <courses> element containing zero or more <course> elements. Each <course> element contains data from a course that matched your search criteria.

Requests sent with no criteria will return *all* courses. Otherwise, courses whose data matches at least one of each criterion will be returned.

Using the above example, all courses that match the <name> criteria *and* at least one of the <code> criteria will be included in the response.

Example:

```
<serv_response>
<head>
  <status>success</status>
</head>
<body>
  <courses>
    <course id="299" isPublished="True" isClosed="False" isLocked="False">
      <name>Project Management Introduction</name>
      <code>PM-001</code>
      <cost>295</cost>
      <credits>3</credits>
      <shortDescription></shortDescription>
      <description></description>
    </course>
    <course id="101" isPublished="False" isClosed="False" isLocked="False">
      <name>Advanced Underwater Basket Weaving</name>
      <code>BW-400</code>
      <cost>495</cost>
      <credits>3</credits>
      <shortDescription></shortDescription>
      <description></description>
    </course>
  </courses>
</body>
</serv_response>
```

Method: getEnrollment

URL: [/_gateway/api/getEnrollment.asp](#)

XSD Schema: [/_gateway/api/schema/getEnrollment.xsd](#)

Description

Use this method to retrieve enrollment data from the LMS.

The XML Request

Your XML request should contain the search criteria that will be used to find the desired enrollment(s). Each criterion shall be included as a element contained within the <body> element of your request.

Allowed criteria and their descriptions are below. You may include each criterion zero or more times.

- <id>
Type: integer
Returns the enrollment whose internal identifier matches the value provided.
- <idCourse>
Type: integer
Returns the enrollment(s) that are for the course whose id corresponds to the value provided.
- <idUser>
Type: integer
Returns the enrollment(s) that belong to the user whose id corresponds to the value provided.

Example:

```
<serv_request>
<head>
  <securityContext>
    <account>www</account>
    <key>1234567890</key>
  </securityContext>
</head>
<body>
  <idCourse>456</idCourse>
  <idUser>5</idUser>
  <idUser>7</idUser>
</body>
</serv_request>
```

The XML Response

The <body> element of the response will contain a single <enrollments> element containing zero or more <enrollment> elements. Each <enrollment> element contains data from an enrollment that matched your search criteria.

Requests sent with no criteria will return *all* enrollments. Otherwise, enrollments whose data matches at least one of each criterion will be returned.

Using the above example, all enrollments that match the <idCourse> criteria *and* at least one of the <idUser> criteria will be included in the response. Therefore, the response would contain all enrollments for Course “456” belonging to User “5” and User “7”. No enrollments for any other courses will be included in the response nor would any enrollments for User “5” or “7” that were not for Course “456”.

Example:

```
<serv_response>
<head>
  <status>success</status>
</head>
<body>
  <enrollments>
    <enrollment id="299" idUser="5" idCourse="495" isSeries="True">
      <name>Project Management Introduction</name>
      <startDate>2010-02-24 15:00:00</startDate>
      <endDate></endDate>
      <accessInterval>1</accessInterval>
      <accessTimeframe>m</accessTimeframe>
      <dueInterval>2</dueInterval>
      <dueTimeframe>ww</dueTimeframe>
      <recurInterval>1</recurInterval>
      <recurTimeframe>yyyy</recurTimeframe>
    </enrollment>
    <enrollment id="475" idUser="7" idCourse="495" isSeries="False">
      <name>Project Management Introduction</name>
      <startDate>2010-01-15 18:00:00</startDate>
      <endDate></endDate>
      <accessInterval>1</accessInterval>
      <accessTimeframe>m</accessTimeframe>
      <dueInterval>2</dueInterval>
      <dueTimeframe>ww</dueTimeframe>
      <recurInterval></recurInterval>
      <recurTimeframe></recurTimeframe>
    </enrollment>
  </enrollments>
</body>
</serv_response>
```

Method: saveUser

URL: /_gateway/api/saveUser.asp

XSD Schema: /_gateway/api/schema/saveUser.xsd

Description

Use this method to save new or modified user account data to the LMS.

The XML Request

Your XML request contains the user account data that is to be saved or updated. The <body> element of the request shall include exactly one <user> element containing attributes and properties as detailed below. All user account data provided in your XML request must comply with the requirements that you have configured within your LMS account on the User Account Data screen.

Attributes:

- id - indicates the user ID of the account that should be updated by the request. If no user account in the LMS contains the ID specified, a new user account will be created with the data contained in the request (to specifically create a new user account, use "0"). This attribute is required.
- enabled (Boolean) – indicates the status of the user account. Enter "true" to indicate an enabled account and "false" to indicate a disabled account. Users cannot login to disabled accounts. This attribute is optional.
- mustChangePassword (Boolean) – true indicates that the next time the user logs into the system they will be required to change their password. Changes to the user's password using the API will not reset this value automatically – only the user changing their password through the LMS interface will reset it.

Note that not all properties contained within the <user> element of an XML response are valid for this XML request. All properties are type *string* unless otherwise noted.

Properties (Elements):

- <username>
- <password>
- <firstName>
- <middleName>
- <lastName>
- <email>
- <company>
- <address>
- <city>
- <province>
- <postalCode>

- <country>
- <phonePrimary>
- <phoneWork>
- <phoneFax>
- <phoneHome>
- <phoneMobile>
- <phonePager>
- <phoneOther>
- <employeeID>
- <department>
- <division>
- <region>
- <jobTitle>
- <jobClass>
- <hireDate> - If provided, this value must be an ISO date or ISO datetime formatted value in GMT. This property is informational only and does not affect user functionality within the LMS.
- <termDate> - If provided, this value must be an ISO date or ISO datetime formatted value in GMT. This property is informational only and does not affect user functionality within the LMS.
- <language>
- <ssn> - If provided, this value is restricted to 9 digits with no other characters or whitespace allowed.
- <gender> - If provided, this value is restricted to "m", "M", "f" or "F"
- <race>
- <dob> - If provided, this value must be an ISO date or ISO datetime formatted value in GMT.
- <expires> - If provided, this value must be an ISO date or ISO datetime formatted value in GMT. User accounts whose expiration dates have passed are not be able to login to the LMS.
- <identifier> - This field is not visible within the LMS interface. It is provided as an alternative to ID, employeeID and username to programmatically identify the user account. Note that this field is used by the LMS's Active Directory Synchronization as well as potentially by other LMS data synchronization features.
- <supervisor> - If provided, this value must be the username of a current user within Inquisiq.
- <field00>
- <field01>
- <field02>
- <field03>
- <field04>
- <field05>
- <field06>
- <field07>
- <field08>

- <field09>

When updating existing user accounts, properties are updated with the data contained within their associated elements. When no associated element is included in the XML request, the user account property will be unaffected.

The following example instructs the LMS to create a new user account with the account data as specified (note the ID attribute with the value "0"). All user properties not specified in this request will not be populated.

Example #1:

```
<serv_request>
  <head>
    <securityContext>
      <account>www</account>
      <key>1234567890</key>
    </securityContext>
  </head>
  <body>
    <user id="0" enabled="true" mustChangePassword="true" >
      <username>jsmith</username>
      <password>p@55w0rd</password>
      <firstName>John</firstName>
      <middleName></middleName>
      <lastName>Smith</lastName>
      <email>jsmith@icslearninggroup.com</email>
    </user>
  </body>
</serv_request>
```

The following example instructs the LMS to update the user account whose ID is "1001" with the data specified. User account properties not specified in the XML request will retain their current value. Note that because the "region" property is specified with no value, the current value contained in the user account for that property will be replaced with no value.

Example #2:

```
<serv_request>
  <head>
    <securityContext>
      <account>www</account>
      <key>1234567890</key>
    </securityContext>
  </head>
  <body>
    <user id="1001">
      <lastName>Johnson</lastName>
      <region></region>
    </user>
  </body>
</serv_request>
```

The XML Response

The XML response contains the user account data of the updated or saved account. A single <user> element will be contained within the <body> tag.

For new accounts, the ID attribute of the <user> element will be populated with the system-assigned ID of the account.

Example:

```
<serv_response>
<head>
  <status>success</status>
</head>
<body>
  <user id="1001" enabled="True" mustChangePassword="False">
    <username>jsmith</username>
    <firstName>John</firstName>
    <middleName></middleName>
    <lastName>Smith</lastName>
    <email>jsmith@mycompany.com</email>
    <company>My Company</company>
    <address>123 Main Street</address>
    <city>New York</city>
    <province>New York</province>
    <postalCode>12345</postalCode>
    <country>US</country>
    <phonePrimary>123-456-7890</phonePrimary>
    <phoneWork></phoneWork>
    <phoneFax></phoneFax>
    <phoneHome></phoneHome>
    <phoneMobile></phoneMobile>
    <phonePager></phonePager>
    <phoneOther></phoneOther>
    <employeeID>1234567-89</employeeID>
    <department></department>
    <division></division>
    <region></region>
    <jobTitle>President</jobTitle>
    <jobClass>Management</jobClass>
    <hireDate></hireDate>
    <termDate></termDate>
    <language></language>
    <ssn>123456789</ssn>
    <gender>m</gender>
    <race></race>
    <dob>1960-06-01 00:00:00</dob>
    <created>2010-01-01 15:29:56</created>
    <expires>2015-12-31 00:00:00</expires>
    <identifier></identifier>
    <field00></field00>
  ...

```

```
        <field09></field09>
    </user>
</body>
</serv_response>
```

Method: saveEnrollment

URL: /_gateway/api/saveEnrollment.asp

XSD Schema: /_gateway/api/schema/saveEnrollment.xsd

Description

Use this method to enroll a user or modify existing user enrollments within the LMS.

The XML Request

Your XML request contains the enrollment that is to be saved or updated. The <body> element of the request shall include exactly one <enrollment> element containing attributes and properties as detailed below.

Attributes:

- id - indicates the ID of the enrollment that should be updated by the request. If no enrollment in the LMS contains the ID specified, a new enrollment will be created with the data contained in the request (to specifically create a new enrollment, use "0"). This attribute is required.

Note that not all properties contained within the <enrollment> element of an XML *response* are valid for this XML request.

Properties (Elements):

- <idUser>
Type: integer
Specifies the ID of the user account that the enrollment is assigned to.
- <idCourse>
Type: integer
Specifies the ID of the course that the enrollment is for.
- <startDate>
Type: dateTime
Specifies the start date and time of the enrollment. This is when the enrollment will appear on the user's "My Account" page.
- <endDate>
Type: dateTime
Specifies the end date of a recurring enrollment. The enrollment will recur per the schedule defined by the recurInterval and recurTimeframe (described below) until the end data specified here.
- <dueInterval>
Type: integer
Specifies the due date interval (refer to the following element description for more details).
- <dueTimeframe>
Type: restricted value: ("yyyy", "m", "ww", "d")

Specifies the timeframe of the due date interval. A due interval of “1” with a due timeframe of “m” indicates that the due date of the enrollment shall be exactly 1 month after the start date.

- <recurInterval>

Type: integer

Specifies the recurrence interval (refer to the following element description for more details).

- <recurTimeframe>

Type: restricted value: (“yyy”, “m”, “ww”, “d”)

Specifies the timeframe of the recurrence interval. A recurrence interval of “2” with a recurrence timeframe of “yyy” indicates that the enrollment shall recur every 2 years from the start date and until the end date.

- <accessInterval>

Type: integer

Specifies the access interval (refer to the following element description for more details).

- <accessTimeframe>

Type: restricted value: (“yyy”, “m”, “ww”, “d”)

Specifies the timeframe of the access interval. An access interval of “6” with an access timeframe of “ww” indicates that the enrolled course shall be accessible for exactly 6 weeks after the start date.

When modifying an enrollment, the idUser and idCourse properties are ignored as they may not be changed for an existing enrollment.

The following example instructs the LMS to enroll User “7” into Course “495”. The enrollment will start on January 1, 2010 at 12:30 GMT and recur every year until January 1, 2020 at 12:30 GMT. Each occurrence will be due after 2 weeks and the user may access the course for 1 month (this allows for the user to review the completed course beyond the specified due date).

Example:

```
<serv_request>
  <head>
    <securityContext>
      <account>www</account>
      <key>1234567890</key>
    </securityContext>
  </head>
  <body>
    <enrollment id="0">
      <idUser>7</idUser>
      <idCourse>495</idCourse>
      <startDate>2010-01-01 12:30:00</startDate>
      <endDate>2020-01-01 12:30:00</endDate>
      <dueInterval>2</dueInterval>
      <dueTimeframe>ww</dueTimeframe>
      <recurInterval>1</recurInterval>
      <recurTimeframe>yyyy</recurTimeframe>
      <accessInterval>1</accessInterval>
      <accessTimeframe>m</accessTimeframe>
    </enrollment>
  </body>
</serv_request>
```

The XML Response

The XML response contains the data of the updated or saved enrollment. A single <enrollment> element will be contained within the <body> tag.

For new enrollments, the ID attribute of the <enrollment> element will be populated with the system-assigned ID of the account.

Example:

```
<serv_response>
  <head>
    <status>success</status>
  </head>
  <body>
    <enrollment id="475" idUser="7" idCourse="495" isSeries="True">
      <name>Project Management Introduction</name>
      <startDate>2010-01-01 12:30:00</startDate>
      <endDate>2020-01-01 12:30:00</endDate>
      <dueInterval>2</dueInterval>
      <dueTimeframe>ww</dueTimeframe>
      <recurInterval>1</recurInterval>
      <recurTimeframe>yyyy</recurTimeframe>
      <accessInterval>1</accessInterval>
    </enrollment>
  </body>
</serv_response>
```