# VitalSource API Version 3

## VitalSource API Version 3

Last Updated: May 20, 2013

## Table of Contents



b.38
b.38

## General Notes

### Scope

This documentation refers to version 3 of the API only, hereafter referred to as *API v3*.

### Authorization to use the API

All requests require an API Key, set as an HTTP request header [ *X-VitalSource-API-Key: ABCDEFGHIJKLMNOP* ]

### Architectural Style

VitalSource API v3 is a REST-based API.  Plain HTTP response codes are favored over additional XML when responding to queries, if possible.

### Errors

Responses will be comprised of HTTP error codes, or an HTTP 200 response with an XML body containing further explanation when needed.

### HTTP Protocol

Secure Sockets Layer (SSL) is required when accessing the VitalSource API.

### Versioning

API v3 is the first version of the API to explicitly require a version number in the URL path.

### Hosts

API v3 has a different set of hosts than previous versions:
production - api.vitalsource.com
staging - api-staging.vitalsource.com
development - api-dev.vitalsource.com
{+}https://api.vitalsource.com+ is the endpoint for API v3. Clients should only use this domain name when referring to this service. Do not hard code any IP addresses into your client or network firewalls, as this will impact your access to our services as we scale our infrastructure to improve service performance or we fall back to our DR data center in times of emergency. If you need verification that you are talking to our services, your application can validate against our SSL certificate.

### Notable changes from previous API revisions

1. The *GUID* can no longer be used as an authentication token.  It will continue to serve its primary function of uniquely identifying an account.  A new concept called an A*ccess Token* has been created for authentication.  The access token grants the same privileges that the email and password provide.  It is important to note, however, that access tokens may expire at some point in the future.  If this happens, clients will need to provide valid account credentials[ |#id.3f0964e5a315]to obtain the new access token.
2. API Keys, passwords, and access tokens are passed as HTTP headers instead of URL parameters.
3. The API is versioned now and the version is part of the URL path structure.
4. Dates/times are returned in the HTTP date format (specified in rfc 822)
5. The hostnames for API v3 services have changed from services.vitalbook.com to api.vitalsource.com.

### Definitions

- Access Token - used by the Partner to gain access to the Protected Resources on behalf of the User, instead of using the User's VitalSource credentials. (use header *X-VitalSource-Access-Token*)
- User - an individual who has an account with VitalSource.
- Partner - a website or application that accesses VitalSource on behalf of the User.
- Protected Resource(s) - data controlled by VitalSource, which the Partner can access through authentication.
- API Key - value used by the Partner to identify itself to VitalSource. (use header *X-VitalSource-API-Key*)

### Further Reading

- HyperText Transfer Protocol (HTTP)
- eXtensible Markup Language (XML)
- Representational State Transfer (REST)

http://en.wikipedia.org/wiki/Representational_State_Transfer
[TOC]

# *Authentication*

## Verify Credentials (Bulk)

Definition: A *valid account credential* is either a) email/password, or b) reference. From these, a valid access token can be obtained which grants an API client the ability to act on behalf of a user account. All calls which operate on a specific user require a valid access token API v3.
Purpose: check (in bulk, if necessary) if credentials are valid. if they are, the access token, email, and guid are returned in the response.
**URL**:
https://api.vitalsource.com/v3/credentials.*xml*
**Formats**:

XML
**HTTP Method**:
POST

**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>
<credential email="emailgoeshere" password="passwordgoeshere"/>
...
<credential access-token="cdefghijklmnopqrstuvwxyzab" />
</credentials>
**URL Parameters**:
None
**Requires user authentication**:
false
**Headers**: *X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Response**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>
<credential guid="ABCDEFGHIJKLMNOP" email="abc@abc.com" access-token="abcdefghijklmnopqrstuvwxyz" />
...
<credential guid="CDEFGHIJKLMNOPQR" email="cde@cde.com" access-token="cdefghijklmnopqrstuvwxyzab" />
</credentials>
-
**Usage examples**:
**Scenario**: Given a single valid email and password
**Request**: https://api.vitalsource.com/v3/credentials.xml
*Headers for API Key are always the same, and as such are excluded from the rest of this document. Calls which require authentication should also include the *X-VitalSource-Access-Token* header.
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>
<credential email="bob@bob.com" password="secret"/>
</credentials>
**HTTP Code**: 200
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>
<credential guid="ABCDEFGHIJKLMNOP" email="bob@bob.com" access-token="cdefghijklmnopqrcdefghijklmnopqr" />
</credentials>
-
**Scenario**: Given a set of credentials which contains a valid credential (bob@bob.com/secret), and 2 invalid credentials (sally@sally.com/wrongpassword and token wxyzabcd1234efghwxyzabcd1234efgh).
**Request**: https://api.vitalsource.com/v3/credentials.xml
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>
<credential email="bob@bob.com" password="secret" />
<credential email="sally@sally.com" password="wrongpassword"/>
<credential access-token="wxyzabcd1234efghwxyzabcd1234efgh" />
</credentials>

**HTTP Code**: 200
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>
<credential guid="ABCDEFGHIJKLMNOP" email="bob@bob.com" access-token="cdefghijklmnopcdefghijklmnop" />
<error email="sally@sally.com" code="466" message="Email or password was not accepted"/>
<error access-token="wxyzabcd1234efghwxyzabcd1234efgh" message="Invalid access token"/>
</credentials>
**Scenario**: Given a valid reference field as a credential, i.e., reference = ab12344511
**Request**: https://api.vitalsource.com/v3/credentials.xml
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>
<credential reference="ab12344511"/>
</credentials>

**HTTP Code**: 200
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<credentials>

<credential guid="ABCDEFGHIJKLMNOP" email="bob@bob.com" access-token="cdefghijklmnopcdefghijklmnop" />
</credentials>
[TOC]

**Create an Authenticated Redirect**
Purpose: provides a temporary URL which automatically signs a user into VitalSource and redirects to a location
**URL**:
https://api.vitalsource.com/v3/redirects.*xml*
**Formats**:
XML
**HTTP Method**:
POST

**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<redirect>
<destination>https://brand-dev.vitalsource.com/books/123456789</destination>
<brand>brand-dev.vitalsource.com</brand>
</redirect>
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
*X-VitalSource-Access-Token*: abcefghijklmnopqrstuvwxyzabcdef
**URL Parameters**:
None
**Fields**:
destination - where to redirect
brand - specify a custom brand [optional]
**Requires user authentication**:
true
**HTTP Code**: 200
**Response Body**:

<?xml version="1.0" encoding="UTF-8"?>
<redirect auto-signin="http://online.vitalsource.com/redirects/2Q26C55VM7JHXFB6NVUQJ6MNTWWTATTA86ZBSDHYSCKDXPGRMP"
expires="Thu Oct 06 16:51:51 +0000 2011">
</redirect>

**Notes**
*require-license* is not an option in this service (available in previous version); use License service to check product license/expiry.

# Licensing

## Get License

**Definition**: A product *license* grants the user the right to access a product until the license expiration date.
**Purpose**: provides a listing of all products and their expirations or checks a particular product's licensing status
**URL**:
https://api.vitalsource.com/v3/licenses.*xml*
**Formats**:
XML
**HTTP Method**:
GET
**URL Parameters**:
*sku* - the product sku for which to determine licensing [optional]
license_type - ('online' or 'download' or 'all') what types of licenses are you interested in ... default is all [optional]
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
*X-VitalSource-Access-Token*: abcefghijklmnopqrstuvwxyzabcdef

**Request Body**:
None
**Requires user authentication**:
true
**Response**:
<?xml version="1.0" encoding="UTF-8"?>

<licenses>
<summary distributable_products="2" total_products="3"/>
<license type="online" code-use="code-api" sku="skugoeshere1" name="namegoeshere" expiration="expirygoeshere" publisher="publishergoeshere"/>
<license type="download" code-use="code-api" sku="skugoeshere1" name="namegoeshere" expiration="expirygoeshere" publisher="publishergoeshere"/>
<license type="online" code-use="code-api" sku="skugoeshere2" name="namegoeshere" expiration="never" publisher="publishergoeshere"/>
<license type="download" code-use="code-api" sku="skugoeshere2" name="namegoeshere" expiration="never" publisher="publishergoeshere"/>
<license type="online" code-use="promotional" publisher="Mosby" sku="9780323055321M" expiration="Tue, 30 Oct 2012 16:08:47 GMT" term="" name="Anatomy & Physiology" imprint="Mosby"/>
</licenses>
**Notes**:

- If the product does not have an expiration date value, the value of the field will be *never*.
- The products shown are filtered by the calling company's distribution rights.
- typical code uses (also known as code-type) are: *promotional*, *add/drop*, *p+e*, and *code-api (which is the default)*

_
**Usage examples**:
**Scenario**: Retrieve all licensed products for a user
**Request**: http://api.vitalsource.com/v3/licenses.xml
**Request Body**:
None
**HTTP Code**: 200
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<licenses>
<summary distributable_products="2" total_products="2"/>
<license type="online" code-use="" sku="ABC123" name="20,000 Leagues Under the Sea" expiration="Mon, 30 Dec 2010 15:30:33 GMT" publisher="Hayes Barton Press"/>
<license type="online" code-use="code-api" sku="DEF456" name="The Grapes of Wrath" expiration="never" publisher="Penguin Books"/>
</licenses>

**Scenario**: Check a user's license for a particular product that they have
**Request**: GET https://api.vitalsource.com/v3/licenses.xml?sku=ABC123
**Request Body**:
None
**HTTP Code**: 200
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<licenses>
<summary distributable_products="1" total_products="5"/>
<license type="download" code-use="code-api" sku="ABC123" name="20,000 Leagues Under the Sea" expiration="Mon, 30 Dec 2010 15:30:33 GMT" publisher="Hayes Barton Press" />
</licenses>

**Scenario**: Check a user's license for a particular product that they do not have
**Request**: GET https://api.vitalsource.com/v3/licenses.xml?sku=GHI678
**Request Body**:
None
**HTTP Code**: 200
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<licenses />
XML ERROR CODES:

| Code | Message |
|---|---|
| 902 | Product sku could not be found |

[TOC]

# Users

# Create User

**Purpose**: create a VitalSource user account
**URL**:
https://api.vitalsource.com/v3/users.*xml*
**Formats**:
XML
**HTTP Method**:
POST
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<user>
 <email>bob%40vitalbook.com</email>**
 <password>sho</password>**
 <reference>bbobbington1234</reference>**
 <first-name>Bob</first-name>
 <last-name>Bobbington</last-name>
 <question-id>1</question-id>
 <question-response>12345</question-response>
 <promote-option>1</promote-option>*
 <survey-option>0</survey-option>*
 <affiliate>Penn State University</affiliate>*
 <redemption-code>ABCDEFGHIJKLMNOP</redemption-code>*
 <profile-url>http://someurl.com/profile</profile-url>*
 <notify>false</notify>*
</user>
```

**Fields**:

- - Optional
  - - Depends on the combination (see Notes section)
    Email - the user's current email address [depends]
    Password - Used in conjunction with the email address to identify the user [depends]
    Reference - An identifier that your organization would like to be associated to this user. May be used in lieu of email/password for online integrations. [depends]
    Promote option - "Keep up to date with VitalSource news, special promotions, and software updates" [optional]
    Survey option - "Keep me in mind for surveys about how we can improve our software" [optional]
    Affiliate (School or Org.) - Free-form text field describing the affiliated organization [optional]
    Redemption Code - To attempt to register a user and a code at the same time, pass in a valid redemption code. [optional]
    Profile URL - If profile management should be handled elsewhere (and not managed by VitalSource web applications), include a profile link. [optional]
    Notify - By default an email notification/welcome message is sent to the user when the account is created. Suppress by setting to false. [optional]
    GUID - A globally unique identifier for a VitalSource user (max 32 chars). If the user already exists, this field is only returned if the credentials match.
    Access Token - token value to be used in lieu of the user's credentials when operating with the VitalSource API. May be reset on occasion.
    Question-ID - security question options to aid in manual account verification/recovery by Support

| ID | Question |
|---|---|
| 1 | What is your Student ID number? |
| 2 | What is your favorite color? |
| 3 | In what city were you born? |
| 4 | What is your pet's name? |
| 5 | What is your mother's maiden name? |

**Notes**
In API v3, *reference* is now a company-specific value, as determined by the API Key.
**Requires user authentication**:
false
**Response**:

```
<?xml version="1.0" encoding="UTF-8"?>
<user>
 <email>bob@vitalboook.com</email>
```

<first-name>Bob</first-name>
<last-name>Bobbington</last-name>
<guid>ABCDEFGH01234567</guid>
<access-token>ghijklmnopqrstuvwghijklmnopqrstuvw</access-token>
<library>
<item href="vbk:1-59339-248-6" title="2005 Britannica Concise Encyclopaedia" author="Encyclopaedia Britannica" />
</library>
</user>
Notes:
There are now 2 options for identifying a user upon creation, depending on the intention for use of the account in the future. The options are outlined in the table below. One of these options must be used or else *error 485* will be thrown.

| option | required fields | user client options | authentication options |
|---|---|---|---|
| Email and password | email, password | Online & Downloadable | access token via API Services<br>email/password in Online & Downloadable client login |
| *Reference* field only | reference | Online only via sign-in service | access token only via API Services |

XML ERROR CODES:

| Code | Message |
|---|---|
| 463 | Question is invalid |
| 464 | Account already exists |
| 465 | Data validation error |
| 468 | Redemption code is invalid |
| 469 | Code has been redeemed |
| 470 | Redemption failed |
| 482 | Malformed create user request |
| 484 | Malformed update user request |
| 485 | Either the *reference* field or *email*/*password* are required |
| 902 | Access token and user do not match |
| 903 | User reference already exists |

**Usage examples**:
**Scenario**: Create a user with a email/password
**Request**: POST https://api.vitalsource.com/v3/users.xml
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<user>

<email>bob@bobbington.com</email>
<password>sho</password>
<first-name>Bob</first-name>
<last-name>Bobbington</last-name>
<question-id>1</question-id>
<question-response>12345</question-response>
<promote-option>1</promote-option>*
<survey-option>0</survey-option>*
<affiliate>Penn State University</affiliate>*
<redemption-code>1234567890123...</redemption-code>*
<profile-url>http://someurl.com/profile</profile-url>*
<notify>false</notify>*
</user>
Response:
HTTP Code: 200

Body:
<?xml version="1.0" encoding="UTF-8"?>

<user>
 <email>bob@vitalboook.com</email>
 <first-name>Bob</first-name>
 <last-name>Bobbington</last-name>
 <guid>ABCDEFGH01234567</guid>
 <access-token>ghijklmnopqrstuvwghijklmnopqrstuvw</access-token>
 <library>
   <item href="vbk:1-59339-248-6" title="2005 Britannica Concise Encyclopaedia" author="Encyclopaedia Britannica" />
 </library>
</user>

**Scenario**: Create a user with a *reference* identifier and no email/password
**Request**: POST https://api.vitalsource.com/v3/users.xml
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<user>
<first-name>Bob</first-name>
<last-name>Bobbington</last-name>
<reference>bbobbington1234</reference>
<question-id>1</question-id>
<question-response>12345</question-response>
</user>
Response:
HTTP Code: 200
Body:
<?xml version="1.0" encoding="UTF-8"?>

<user>
 <first-name>Bob</first-name>
 <last-name>Bobbington</last-name>
 <guid>ABCDEFGH01234567</guid>
 <access-token>ghijklmnopqrstuvwghijklmnopqrstuvw</access-token>
 <library>
   <item href="vbk:1-59339-248-6" title="2005 Britannica Concise Encyclopaedia" author="Encyclopaedia Britannica" />
 </library>
</user>

**Scenario**: Create a user with data validation errors
**Request**: POST https://api.vitalsource.com/v3/users.xml
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<user>
<email>bob%40vitalbook.com!</email>
<password>sho&</password>
<first-name>Bob</first-name>
<last-name>Bobbington</last-name>
<reference>bbobbington1234</reference>
<question-id>1</question-id>
<question-response>12345</question-response>
</user>
**Response**:
**HTTP Code**: 200
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>465</error-code>
<error-text>Data validation error</error-text>
<error>
<field>email</field>
<message>contains invalid characters: ! # $ % ^ * < > ( ) + space</message>
</error>
<error>
<field>password</field>
<message>contains invalid characters: % & < > + ; space</message>
</error>
</error-response>

**Scenario**: Create a user without *email*/*password* or *reference* fields

**Request**: POST https://api.vitalsource.com/v3/users.xml
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<user>
<first-name>Bob</first-name>
<last-name>Bobbington</last-name>
<question-id>1</question-id>
<question-response>12345</question-response>
<promote-option>1</promote-option>
<survey-option>0</survey-option>
<affiliate>Penn State University</affiliate>
<notify>false</notify>
</user>
**Response**:
**HTTP Code**: 200
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>485</error-code>
<error-text>Either the *reference* field or *email*/*password* are required</error-text>
</error-response>

**Update User**
**Purpose**: update an existing VitalSource account

**URL**:
https://api.vitalsource.com/v3/users.*xml*
**Formats**:
XML
**HTTP Method**:
PUT
**Requires user authentication**:
true
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
*X-VitalSource-Access-Token*: abcefghijklmnopqrstuvwxyzabcdef
**Notes**:
This call has the same structure as the User Create, however, the *redemption-code* option is not available on update.

**Scenario**: Update a user record
**Request**: PUT https://api.vitalsource.com/v3/users.xml/email%40email.com
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<user>
<first-name>Bob</first-name>
<last-name>Bobbington</last-name>
<reference>bbobbington1234</reference>
<question-id>1</question-id>
<question-response>12345</question-response>
</user>
Response:
HTTP Code: 200
Body:
<?xml version="1.0" encoding="UTF-8"?>

<user>
<email>damon@vitalsource.com</email>
<first-name>Bob</first-name>
<last-name>Bobbington</last-name>
</user>

**Show User**
**Purpose**: Show an existing VitalSource account (or check for its existence)

**URL**:
https://api.vitalsource.com/v3/users.xml/<*identifier*>

**Formats**:
XML
**Requires user authentication**:
false
**HTTP Method**:
GET
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
*X-VitalSource-Access-Token*: abcefghijklmnopqrstuvwxyzabcdef (optional: Access Token can be used to identify the user, but is not required on this call)
**Notes**:
Identifier can be *email* or *GUID*

**Usage examples**:
**Scenario**: Show a user with an email address
**Request**: GET https://api.vitalsource.com/v3/users.xml/bob%40bob.com
**Request Body**: none
Response:
HTTP Code: 200
Body:

<?xml version="1.0" encoding="UTF-8"?><user> <email>bob@bob.com</email> <first-name>Bob</first-name> <last-name>Bobbington</last-name></user>

**Scenario**: Show a user with a GUID
**Request**: GET https://api.vitalsource.com/v3/users.xml/ABCDEFGHIJKLMNOP
**Request Body**: none
Response:
HTTP Code: 200
Body:

<?xml version="1.0" encoding="UTF-8"?><user> <email>bob@bob.com</email> <first-name>Bob</first-name> <last-name>Bobbington</last-name> <last-activation-date>Wed, 23 Aug 2010 22:50:51 GMT</last-activation-date></user>

**Scenario**: Show a user by email address which matches a provided valid access token (returns GUID)
**Request**: GET https://api.vitalsource.com/v3/users.xml/bob%40bob.com
**Request Body**: none
**Header**: X-VitalSource-Access-Token: abcdefghijklmnop01234567890
Response:
HTTP Code: 200
Body:

<?xml version="1.0" encoding="UTF-8"?><user> <email>damon@vitalbook.com</email> <first-name>Damon</first-name> <last-name>Clinkscales</last-name> <guid>B9MAJ3BGDMY79TU4</guid> <uber-license>false</uber-license> <last-activation-date>Wed, 23 Aug 2006 22:50:51 GMT</last-activation-date></user>

**Scenario**: Show a user by email address which does not match a provided valid access token
**Request**: GET https://api.vitalsource.com/v3/users.xml/bob%40bob.com
**Request Body**: none
**Header**: X-VitalSource-Access-Token: zzkdefghijklmnopqrtstuv12345
Response:
HTTP Code: 200
Body:

<?xml version="1.0" encoding="UTF-8"?><user> <email>damon@vitalbook.com</email> <first-name>Damon</first-name> <last-name>Clinkscales</last-name></user>

**Scenario**: Show a user by email address including their question id/response, promotion option, and survey option in the response
**Request**: GET https://api.vitalsource.com/v3/users.xml/bob%40bob.com?full=true
**Request Body**: none
**Header**: None
Response:
HTTP Code: 200
Body:

<?xml version="1.0" encoding="UTF-8"?><user> <email>damon@vitalbook.com</email> <first-name>Damon</first-name> <last-name>Clinkscales</last-name> <sso-only>false</sso-only> <require-deactivate>false</require-deactivate> <require-book-list-update>false</require-book-list-update> <timestamp>20120301043428</timestamp>

<promote-option>true</promote-option> <survey-option>false</survey-option>
<question-response>1234567</question-response> <question-id>1</question-id></user>

**Scenario**: Show a user with an email address that does not exist
**Request**: GET https://api.vitalsource.com/v3/users.xml/123%40123.com
**Request Body**: none
Response:
HTTP Code: 404
Body: none

**Reset Access Token**
**Purpose**: force a reset of a VitalSource account's access token

**URL**:
https://api.vitalsource.com/v3/users/reset_access.*xml*
**Formats**:
XML
**HTTP Method**:
POST
**Requires user authentication**:
true
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
*X-VitalSource-Access-Token*: abcefghijklmnopqrstuvwxyzabcdef
**Notes**:
This call is useful for verifying the API flow in the case where an access token has changed and you need to re-establish access to a user account. Please keep in mind that if this is reset, you'll need a valid credential to retrieve the new access token.

Response:
HTTP Code: 200
Body: none
[TOC]

# Redemptions

**Bulk Redeem**
**Purpose**: Given a list of users, products, and a guaranteed minimum number of years of access, smartly redeem content for those users who have fewer than the minimum years of access; skip the others. Currently, the only valid values for the license terms are: some positive # of days or "perpetual".

**URL**:
https://api.vitalsource.com/v3/redemptions/bulk.*xml*
**Formats**:
XML
**Requires user authentication**:
false (as a header, no, however each redemption must have a valid access token)
**HTTP Method**:
POST

**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Request Body**:

<?xml version="1.0" encoding="UTF-8"?>
<redemptions callback="https://yourserver.com/yourpath/alldone"> <redemption access-token="abcdefghijklmnopqrstuvwxyz123456" sku="0390222801" term="Perpetual" online-term="1825" tag="4018045" ensure-years="3" /> <redemption access-token="abcdefghijklmnopqrstuvwxyz654321" sku="0077376323" term="1825" ensure-years="3" />
<redemption access-token="abcdefghijklmnopqrstuvwxyz654321" sku="0074231723" term="55" code-type="add-drop" />
</redemptions>

**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<redemptions status="processing"/>
**Callback POST Body**: (upon task completion, the callback URL will be hit with the results for the bulk update, if a callback is specified)

<?xml version="1.0" encoding="UTF-8"?>

<redemptions> <redemption access-token="abcdefghijklmnopqrstuvwxyz123456" sku="0390222801" status="redeemed" code="XYZABCDEFBUZGHIKKABD"/> <redemption access-token="abcdefghijklmnopqrstuvwxyz654321" sku="0077376323" status="skipped" msg="/>
</redemptions>

**Fields**:

- **access-token** (required) - the user's current (and unexpired) access token. the token can be retrieved (in bulk, if necessary) from the Credentials call
- **sku** (required) - product identifier for the product for which we are granting access
- **term** (required) - the desired license length to grant the user for downloadable access (two options: # of days, like "1825", or "Perpetual" for access that never expires)
- **online-term** (optional) - the desired license length to grant the user for online access. If this field is not specified, term will be used in place of online-term. (two options: # of days, like "1825", or "Perpetual" for access that never expires)
- **tag** (optional) - arbitrary tag which will be attached to any redemption code which is created (for reporting purposes)
- **ensure-years** (optional) - refers to the minimum number of years you wish for this user to have for the product. defaults to 3 years. if the account holder does not already have this number of years available, a new redemption code will be created and redeemed which grants the user access for the requested term length. To force the redemption to happen regardless of the current state of the user's content, pass an ensure-years value of -1.
- **callback** (optional) - if this attribute is present, it registers your desire to receive the status of this bulk request posted back to you at this URL. If this is not specified, you will receive no details of the bulk redemption execution.
- **code-type** (optional) - comp, add-drop,p-plus-e are valid values. if *code-type* is missing or blank, the code type will be "code-api".
- **status** (in bulk redemption response body) - *processing*
- **status** (individual redemption in callback response body) - *skipped*, *redeemed*, or *error*
- **message** - explanatory message about the particular redemption request

**Error Codes**:

| Code | Message |
|---|---|
| 901 | Malformed XML request |

## Redeem for a User

**Purpose**: Redeem a code for a particular VitalSource user account

**URL:**
https://api.vitalsource.com/v3/redemptions.*xml*
**Formats:**
XML
**Requires user authentication:**
false
**HTTP Method:**
POST

**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Request Body:**

<?xml version="1.0" encoding="UTF-8"?>

<redemption>
<user>damon@vitalbook.com</user>
<code>FKKZT47FW4C4S7SNXR3Z</code>
</redemption>

**Response Body:**
<?xml version="1.0" encoding="UTF-8"?>
<library>
<item href="vbk:1-59339-248-6" title="2005 Britannica Concise Encyclopaedia" author="Encyclopaedia Britannica"></item>
</library>
**Fields**:
User: either the email address or the GUID on file for the user's account.
Code: A valid, unexpired VitalSource redemption code
Notes:

- **If the access token header is set, that user value will override the user value in the XML.**
- Authentication is not required to add content to a user's account, thus there is no password or access token requirement in the call.
- The <library> response is named that way for legacy reasons.  It contains references to the content items which were just activated as a result of the redemption.

XML RESPONSE CODES:

| Code | Message |
|---|---|
| 468 | Redemption code is invalid |
| 469 | Code has been redeemed |
| 470 | Redemption failed |
| 474 | User could not be found |
| 483 | Malformed redemption request |

SUCCESS EXAMPLE (user is identified by the XML element):
POST /v3/redemptions.xml  (Header: no access token header)
BODY:

<?xml version="1.0" encoding="UTF-8"?>
<redemption>
<user>damon@vitalbook.com</user>
<code>FKKZT47FW4C4S7SNXR3Z</code>
</redemption>

RESPONSE:

HTTP RESPONSE CODE: 200 (Ok)
<?xml version="1.0" encoding="UTF-8"?>
<library>
<item href="vbk:1-59339-248-6" title="2005 Britannica Concise Encyclopaedia" author="Encyclopaedia Britannica"></item>
</library>

SUCCESS EXAMPLE (user is identified by the access token header)
POST /v3/redemptions.xml  (Header: X-VitalSource-Access-Token: abcdefghijklmnopqrstuv)

BODY:
<?xml version="1.0" encoding="UTF-8"?>
<redemption>
<code>FKKZT47FW4C4S7SNXR3Z</code>
</redemption>

RESPONSE:
HTTP RESPONSE CODE: 200 (Ok)
<?xml version="1.0" encoding="UTF-8"?>
<library>
<item href="vbk:1-59339-248-6" title="2005 Britannica Concise Encyclopaedia" author="Encyclopaedia Britannica"></item>

</library>
3. FAILURE EXAMPLE:

POST /v3/redemptions.xml  (Header: no access token header)

<?xml version="1.0" encoding="UTF-8"?>
<redemption>
<user>damon@vitalbook.com</user>
<code>FKKZT47FW4C4S7SNXR3Z</code>
</redemption>
RESPONSE:
HTTP RESPONSE CODE: 200 (Ok)
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>474</error-code>
<error-text>User could not be found</error-text>
</error-response>
[TOC]

**Codes**

## Create Codes

**Purpose**: Generate one of more redemption codes

**URL**:
https://api.vitalsource.com/v3/codes.*xml*
**Formats**:
XML
**Requires user authentication**:
false
**HTTP Method**:
POST

**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Request Body**:

<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="perpetual" num-codes="2" online-license-type="numdays" online-num-days="365" />

**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>

<codes> <code>XYZABCDEFBUZGHDKKABD</code>
<code>BCDBZDEFCDEBUAARGHZE</code>
</codes>

**Fields**:

- **sku** (required) - product identifier referenced by the code
- **license-type** (required) - *perpetual*, *numdays*, *absdate*, *noaccess*, *default*
- **num-codes** (required) - the number of codes of this type being requested
- **num-days** (required for numdays) - the number of days this license should be valid for, after redemption
- **exp-year** (required for *absdate*) - expiration year for the absolute date
- **exp-month** (required for absdate) - expiration month for the absolute date
- **exp-day** (required for absdate) - expiration day for the absolute date
- **online-license-type** (optional) - *perpetual*, *numdays*, *absdate*, *noaccess*
- **online-num-days** (required for *numdays*) - the number of days the online license should be valid for, after redemption
- **online-exp-year** (required for *absdate*) - online expiration year for the absolute date
- **online-exp-month** (required for *absdate*) - online expiration month for the absolute date
- **online-exp-day** (required for *absdate*) - online expiration day for the absolute date
- **tag** (optional) - mark a batch of codes with a particular code for tracking purposes. colons and spaces are not permitted in tags. it is suggested that tags be 10 characters or fewer, but there is a 50 character hard limit.

- **code-type** (optional) - comp, add-drop,p-plus-e are valid values
- **price** (optional) - custom price for code, must be > 0.00

**Notes**:

- The *license-type* option refers to the downloadable license for the book.
- The *default* type option for downloadable is new in V3. Instead of specifying a specific license rule for the code, it relies on the default rule for the product. Omitting the *online-type* will use the product default online access setting.
- The *noaccess* license type option provides a way to restrict an access method (downloadable or online).
- If the caller does not have distribution rights to the particular title, they will receive an HTTP 403 (permission denied)
- Hard expiration dates cannot be in the past.

**More Examples**:
**Example Request 1**: Perpetual codes

```
<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="perpetual" num-codes="1" />
```

**Example Request 2**: Absolute Date codes

```
<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="absdate" exp-year="2008" exp-month="2" exp-day="29" num-codes="1" />
```

**Example Request 3**: X Days from Redemption codes

```
<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="numdays" num-days="180" num-codes="1" />
```

**Example Request 4**: Perpetual downloadable access with 1 year of online access

```
<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="perpetual" online-license-type="numdays" online-num-days="365" num-codes="1" />
```

**Example Request 5**: Perpetual downloadable access with no online access

```
<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="perpetual" online-license-type="noaccess" num-codes="1" />
```

**Example Request 6**: No downloadable access with 1 year of online access

```
<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="noaccess" online-license-type="numdays" online-num-days="365" num-codes="1" />
```

**Example Request 7**: Perpetual downloadable access and perpetual online access

```
<?xml version="1.0" encoding="UTF-8"?>
<codes sku="0471430749" license-type="perpetual" online-license-type="perpetual" num-codes="1" />
```

**Example Request 8**: 7 Days from Redemption code with a $5.00 price

```
<?xml version="1.0" encoding="UTF-8"?>
<codes price="5.00" sku="0471430749" license-type="numdays" num-days="7" num-codes="1" />
```

**Cancel Code**
**Purpose**: Cancel a code

**URL**:
https://api.vitalsource.com/v3/codes/ABCDEFGHIJKLMNOP.xml
**Formats**:
XML
**Requires user authentication**:
false
**HTTP Method**:
DELETE

**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Request Body**:

none

**Response Body**:
none

**Fields**:

- **reason** (optional parameter) - the reason for cancellation

**More Examples**:
**Example Request 1**: Cancel existing code, no reason provided

DELETE /v3/codes/CDEFGHIJKLMNOP.xml
RESULT: HTTP 200

**Example Request 2**: Cancel existing code, reason provided

DELETE /v3/codes/CDEFGHIJKLMNOP.xml?reason=dropped
RESULT: HTTP 200

**Example Request 3**: Cancel nonexistent code
DELETE /v3/codes/FOO.xml

Result: HTTP 404

**Show Code**
**Purpose**: Show the status of an existing VitalSource redemption code (or check for its existence)

**URL**:
https://api.vitalsource.com/v3/codes/*thecodegoeshere.xml*
**Formats**:
XML
**Requires user authentication**:
false
**HTTP Method**:
GET
**Headers**:

*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Notes**:

- Only codes which were created by the company of the API key's company will be returned as a result
- An intended-use of *promotional* is the same thing as a "comp copy"

**Usage examples**:
**Scenario**: Show info on a code which exists that you have access to
**Request**: GET https://api.vitalsource.com/v3/codes/ABCDEFGHIJKLMNOP.xml
**Request Body**: none
**Response**:
**HTTP Code**: 200
**Body**:

**Unredeemed**:<?xml version="1.0" encoding="UTF-8"?><code> <id>ABCDEFGHIJKLMNOP</id> <intended-use>add/drop</intended-use> <product-name>Anne of Green Gables</product-name> <product-sku>978012345678</product-sku> <product-eisbn>97887654321</product-eisbn> <creation-details>Code created by XYZ University on August 24, 2011</creation-details> <creation-user>XYZ API</creation-user> <creation-login>xyzAPI</creation-login> <redemption-details>Not redeemed</redemption-details> <download-license-details>Expires 540 days after registration</download-license-details> <online-license-details>Product Default</online-license-details></code> **Redeemed**:<?xml version="1.0" encoding="UTF-8"?><code> <id>CDABCGDGHIJKLMNOP</id> <intended-use>code-api</intended-use> <product-name>Anne of Green Gables</product-name> <product-sku>978012345678</product-sku> <product-eisbn>97887654321</product-eisbn> <creation-details>Code created by XYZ University on August 24, 2011</creation-details> <creation-user>XYZ API</creation-user> <creation-login>xyzAPI</creation-login> <redemption-details>Redeemed on August 24, 2011</redemption-details> <user-name>Bob Dewald</user-name> <user-email>bob@bob.com</user-email> <download-license-details>Expires on September 4, 2011</download-license-details> <online-license-details>Expires 365 days after registration</online-license-details> <tags> <tag>01234111234</tag> </tags></code> **Deactivated**:<?xml version="1.0" encoding="UTF-8"?><code> <id>ADDFEFGHIGP78A124</id> <intended-use>promotional</intended-use> <product-name>Anne of Green Gables</product-name> <product-sku>978012345678</product-sku> <product-eisbn>97887654321</product-eisbn> <creation-details>Code created by XYZ University on August 24, 2011</creation-details> <creation-user>XYZ API</creation-user> <creation-login>xyzAPI</creation-login> <deactivation-details>Deactivate on August 24, 2011</deactivation-details> <deactivation-user>Ted Brown</deactivation-user> <deactivation-login>ted.brown</deactivation-login> <download-license-details>Perpetual License</download-license-details> <online-license-details>Product Default</online-license-details></code>

**Scenario**: Show info on a code which exists that you do not have access to
**Request**: GET https://api.vitalsource.com/v3/codes/ABCDEFGHIJKLMNOP.xml
**Request Body**: none
Response:
HTTP Code: 404
Body: none

**Scenario**: Show info on a code which does not exist
**Request**: GET https://api.vitalsource.com/v3/codes/ABCDEFGHIJKLMNOP.xml
**Request Body**: none
Response:
HTTP Code: 404
Body: none

**Search Codes by Tag**
**Purpose**: Find all of the codes which are tagged with a specific tag value
**URL**:
https://api.vitalsource.com/v3/codes/*search.xml?tag=foo*
**Formats**:
XML
**Requires user authentication**:
false
**HTTP Method**:
GET
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Notes**:

- Only codes which were created by the company of the API key's company will be returned as a result
- Multiple codes may be returned

**Usage examples**:
**Scenario**: Show codes with tag 'foo' when one or more exists
**Request**: GET https://api.vitalsource.com/v3/codes/search.xml?tag=foo
**Request Body**: none
**Response**:
**HTTP Code**: 200
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<codes>
<code>BCDEFGHIJKLMNOPQ</code>
<code>CDEFGHIJKLMNOPQR</code>
</codes>
**Scenario**: Show codes with tag 'bar' when none exist
**Request**: GET https://api.vitalsource.com/v3/codes/search.xml?tag=foo
**Request Body**: none
**Response**:
**HTTP Code**: 200
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<codes>
</codes>
[TOC]

**Products**

## Available Inventory

**Purpose**: provides a listing of all products which are either owned or distributable by the company of the API client.
**URL**:
https://api.vitalsource.com/v3/products/available*.xml*
**Formats**:
XML
**HTTP Method**:
GET

**Request Body**:
None
**Requires user authentication**:
false
**Headers**:
*X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Response**:
<?xml version="1.0" encoding="UTF-8"?>
<products>
<product sku="skugoeshere1" name="namegoeshere" publisher="publishergoeshere1" role="owner" list-price="25.00" build-status="Live"/>
<product sku="skugoeshere2" name="namegoeshere2" publisher="publishergoeshere2" role="distributor" list-price="9.00" build-status="Live"/>
</products>
**Notes**:

- The result set is filtered by what the company of the API user is allowed to distribute.

[TOC]

**Create Custom Package**
**Purpose**: Creates a Package product with a collection of SKUs and assigns a unique SKU to that package.
**URL**:
https://api.vitalsource.com/v3/products/packages.xml
**Formats**:
XML
**HTTP Method**:
POST

**Request Body (example)**:
<?xml version="1.0" encoding="UTF-8"?>
<package>
<name>Texas A & M College of Pharmacy Review</name>

<description>A review library for Texas A & M College of Pharmacy</description>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>432</library-id>
 <license-rule>
   <type>numdays</type>
   <num-days>99</num-days>
 </license-rule>
 <online-license-rule>
   <type>perpetual</type>
 </online-license-rule>
 <products>
   <product sku="sku1" />
   <product sku="sku2" />
 </products>
</package>
**Requires user authentication**:
 false
**Headers**:
 *X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Response**:
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <sku>ELS99AGAKKJ2</sku>
 <name>Texas A & M College of Pharmacy Review</name>
</package>
**Notes**:

- Please use this service to create a "package" of two or more products. For only a single SKU, you can just use that SKU to create the code with the "Create Code Service".
- The *Library ID* the package will reference should be created in Connect and should contain all of the product SKUs passed in. This is important because the products will not show up in Bookshelf if they are not in the library.
- Package SKU (even if one is explicitly provided) will be prefixed with the first 3 characters of the company name of the API Key. This is to avoid collisions in the system. If no SKU is provided, a SKU for the new package will be generated.
- There are 5 potential types for license rule (and online license rule): no access (*noaccess*), absolute date (*absdate*), number of days from redemption (*numdays*), *perpetual*, and *default*.
- If any of the product SKUs do not match, the package will not be created. An error denoting which SKUs don't match will be returned.

**Usage examples**:
**Scenario**: Create a package of 2 products which exist
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>432</library-id>
 <license-rule>
   <type>noaccess</type>
 </license-rule>
 <online-license-rule>
   <type>absdate</type>
<exp-month>12</exp-month>
<exp-day>10</exp-day>
<exp-year>2014</exp-year>
 </online-license-rule>
 <products>
   <product sku="sku1" />
<product sku="sku2" />
 </products>
</package>
Response:
HTTP Code: 200
Body:

<?xml version="1.0" encoding="UTF-8"?>
<package>
 <sku>ELS99AGAKKJ2</sku>
 <name>Texas A & M College of Pharmacy Review</name>
</package>

**Scenario**: Create a package of 2 products where 1 does not exist
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>432</library-id>
 <license-rule>
   <type>perpetual</type>
 </license-rule>
 <online-license-rule>
   <type>default</type>
 </online-license-rule>
 <products>
   <product sku="sku1" />
<product sku="doesnotexist" />
 </products>
</package>
```

Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>993</error-code>
<error-text>Package contains a non-existent SKU</error-text>
<error>
<field>sku</field>
<message>'doesnotexist' is not a valid SKU</message>
</error>
</error-response>
```

**Scenario**: Create a package without required fields, for example, no license rules
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>432</library-id>
 <products>
   <product sku="sku1" />
<product sku="sku2" />
 </products>
</package>
```

Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>994</error-code>
<error-text>Insufficient requirements for package create</error-text>
</error-response>
```

**Scenario**: Create a package with an non-existent library ID or a library which is not associated with the company of the API user.
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>2</library-id>
 <license-rule>
   <type>numdays</type>
```

```
  <num-days>99</num-days>
 </license-rule>
 <online-license-rule>
  <type>perpetual</type>
 </online-license-rule>
 <products>
  <product sku="sku1" />
<product sku="sku2" />
 </products>
</package>
```
Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>995</error-code>
<error-text>Library ID is not acceptable</error-text>
</error-response>
```

**Scenario**: Create a package with SKU that does not begin with the first three characters of the Company of the API key.
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>8899AGAKKJ2</sku>
 <library-id>47</library-id>
 <license-rule>
  <type>numdays</type>
  <num-days>99</num-days>
 </license-rule>
 <online-license-rule>
  <type>perpetual</type>
 </online-license-rule>
 <products>
  <product sku="sku1" />
<product sku="sku2" />
 </products>
</package>
```
Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <sku>ELS8899AGAKKJ2</sku>*
 <name>Texas A & M College of Pharmacy Review</name>
</package>
```
*Note that "ELS" is tacked onto the front of the SKU which was passed in to ensure proper namespacing is maintained across the system.

**Scenario**: Create a package with an invalid license type.
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>2</library-id>
 <license-rule>
  <type>num-dayz</type>
```

```
  <num-days>99</num-days>
 </license-rule>
 <online-license-rule>
  <type>perpetual</type>
 </online-license-rule>
 <products>
  <product sku="sku1" />
<product sku="sku2" />
 </products>
</package>
```

Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>104</error-code>
<error-text>Invalid license type</error-text>
</error-response>
```

**Scenario**: Create a package with an invalid absolute date.
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>432</library-id>
 <license-rule>
  <type>absdate</type>
<exp-month>12</exp-month>
<exp-day>32</exp-day>
<exp-year>2014</exp-year>
 </license-rule>
 <online-license-rule>
  <type>perpetual</type>
 </online-license-rule>
 <products>
  <product sku="sku1" />
<product sku="sku2" />
 </products>
</package>
```

Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>105</error-code>
<error-text>Invalid expiration date</error-text>
</error-response>
```

**Scenario**: Create a package with an invalid number of days on a *numdays* license rule.
**Request**: POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:

```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>2</library-id>
 <license-rule>
  <type>numdays</type>
  <num-days>x</num-days>
 </license-rule>
```

```
 <online-license-rule>
   <type>perpetual</type>
 </online-license-rule>
 <products>
   <product sku="sku1" />
<product sku="sku2" />
 </products>
</package>
```
Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>106</error-code>
<error-text>Number of days cannot be zero</error-text>
</error-response>
```

**Scenario**:  Create a package without any products.
**Request**:  POST https://api.vitalsource.com/v3/products/packages.xml
**Request Body**:
```
<?xml version="1.0" encoding="UTF-8"?>
<package>
 <name>Texas A & M College of Pharmacy Review</name>
 <sku>ELS99AGAKKJ2</sku>
 <library-id>2</library-id>
 <license-rule>
   <type>numdays</type>
   <num-days>99</num-days>
 </license-rule>
 <online-license-rule>
   <type>perpetual</type>
 </online-license-rule>
</package>
```
Response:
HTTP Code: 200
Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<error-response>
<error-code>994</error-code>
<error-text>Insufficient requirements for package create</error-text>
</error-response>
```
[TOC]

**Companies**

**Show Company**
**Purpose**: Show the status of the API client's company

**URL**:
  https://api.vitalsource.com/v3/company*.xml*
**Formats**:
  XML
**Requires user authentication**:
  false
**HTTP Method**:
  GET
**Headers**:
  *X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Notes**:

- Only the API key's company will be returned as a result
- The current potential company permissions are: *add/drop*, *p+e*, *comp*
- Only the permissions which the company has will be present.

**Usage examples**:
**Scenario**: Show info on the API client's company
**Request**: GET https://api.vitalsource.com/v3/company.xml
**Request Body**: none
**Response**:
**HTTP Code**: 200
**Body**:

<?xml version="1.0" encoding="UTF-8"?><company> <name>XYZ Company</name> <permissions>
<permission type="add/drop" />
<permission type="comp" />
<permission type="p+e" />
</permissions></company>
[TOC]
**Cover Images**

**Definition** :
The '*original*' **Cover** is the image seen at the frontmost of a **VitalBook**, usually provided by the content-owner, publisher, or conversion partner.
A **Thumbnail** is an image scaled down from the original cover to a fixed width.
A **VBID** is the globally-unique number (*VitalBook ID)* assigned via our central registry service, which identifies a particular digital asset in the *Vital Book* format. The **VBID** is also used as the **SKU** of the singleton product for a given asset.

**Purpose** :
The covers service enables partners to embed cover images for a given *VitalBook* in their own application interface. Various thumbnail sizes are provided in addition to the arbitrarily-sized *original*. The thumbnails are automatically updated whenever the original cover changes.
**URL :**
"http://covers.vitalbook.com/vbid/#{VBID}/width/#{WIDTH}"
**URL Parameters :**
None
**Formats :**
image/jpeg
**HTTP Method :**
GET

**Request Body :**
None
**Requires user authentication :**
False
**Headers :**
None
**URL Path Components :**
VBID
Width
**Response :**
An image file will be returned.
**Usage Notes :**
Thumbnail images are pre-rendered from the 'original' cover's arbitrary size (which can be very large depending upon the publisher and/or vendor) to a range of eighteen (18) thumbnail widths.
Most users will want to select a specific thumbnail size which best fits their application design.
The Eighteen (18) available thumbnail widths (in pixels) :
**20, 25, 28, 30, 32, 35, 60, 120, 150, 200, 240, 320, 480, 640, 720, 768, 960, 1024**

**Authentication :**
No authentication is currently required for use of the Covers service.
**Generic Covers :**
In the case of a request for the image for a particular VBID which does not exist, a placeholder image will be returned, instead of a 404 error. This ensures that no 'broken image' placeholders are shown to the end user without the calling service needing to check and/or maintain any information related to covers.
A cover will not be available for an in-progress conversion until it receives final distribution approval via our backend system, and will always return the generic vitalbook cover image.
**URL for Thumbnail cover at some specific width :**
"http://covers.vitalbook.com/vbid/#{VBID}/width/#{WIDTH}"
**Thumbnail Scenario :**
As a partner working with the textbook 'Paediatrics and Child Health, 3rd Edition' by 'Mary Rudolf', you want to embed the cover image in the online course syllabus. You know the VBID assigned to it is 9781444394832, so you are able to plug that into the general pattern above to produce the URL for a 240-pixel wide thumbnail :
**URL :**
"http://covers.vitalbook.com/vbid/9781444394832/width/240"
**Image tag :**
<img src="http://covers.vitalbook.com/vbid/9781444394832/width/240" alt="Cover for Paediatrics and Child Health">Paediatrics and Child Health</img>

**Service Response :**
$ _curl -I http://covers.vitalbook.com/vbid/9781444394832/width/240_
HTTP/1.1 200 OK
Date: Wed, 01 Apr 2012 18:36:12 GMT
Server: Apache
X-Powered-By: Phusion Passenger (mod_rails/mod_rack) 3.0.0
File-Name: 240-328.jpg
Served-By: app07.vitalbook.com
Last-Modified: Tue, 14 Jun 2011 15:37:43 GMT
Content-Length: 34459
Status: 200
Cache-Control: max-age=2592000
Expires: Sun, 08 Apr 2012 18:36:12 GMT
MS-Author-Via: DAV
Content-Type: image/jpeg
**Notes :**
The "File-Name" header will contain the actual image file name, which is the width followed by the height. In the above example, the file name is 240-328, so the height is 328 pixels. This is potentially useful in edge cases which may require you to first request the header info in order to determine the exact image dimensions.
In another view, you may want to show multiple VitalBook icons, so you select one of the smaller thumbnail sizes, such as the 60-pixel width, and swap 240 for 60 as the width component of the URL path.
**URL :**
"http://covers.vitalbook.com/vbid/9781444394832/width/60"
**Image tag :**
<img src="http://covers.vitalbook.com/vbid/9781444394832/width/60" alt="Cover for Paediatrics and Child Health">Paediatrics and Child Health</img>
**'Zoom' Scenario :**
You might choose to embed the original sized cover for a trivial 'zoom' functionality, perhaps in response to a click, tap, or hover event.
**URL for Full cover at arbitrary original dimensions :**
"http://covers.vitalbook.com/vbid/#{VBID}"
*Please note :* no indications are made regarding the size of the 'original' cover, not in the headers, or by any other means. The client must accept it as an arbitrary size, or determine it from the jpeg file via internal methods. Most modern web browsers will happily accommodate images without pre-determined sizes.
[TOC]

---

# *Unassigned Codes*

## Generate Unassigned Codes

Create one or more *unassigned codes* which are a special kind of VitalSource redemption code which have no SKU associated with them.
**URL**:
https://api\[-dev,-staging\].vitalsource.com/v3/unassigned_codes.xml?n=20
**Formats**:
XML
**HTTP Method**:
POST
**Request Body**:
None
**URL Parameters**:

- *n* - the number of unassigned codes in your batch

**Requires user authentication**:
false
**Headers**: *X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<unassigned-codes> <code>XYZABCDEFBUZGHDKKABD</code> <code>BCDBZDEFCDEBUAARGHZE</code>
</unassigned-codes>

## Map Unassigned Codes

Map an *unassigned code* to a SKU. Accepts multiple mappings at one time. Redeemed codes cannot be remapped--an error will occur

**URL**:
https://api\[-dev,-staging\].vitalsource.com/v3/unassigned_codes/mappings.xml
**Formats**:
XML
**HTTP Method**:
POST
**Request Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mapping> <unassigned-code code="XYZABCDEFBUZGHDKKABD" sku="ABCD1234" remap="true" />
</mapping>
**URL Parameters**:
None
**Requires user authentication**:
false
**Headers**: *X-VitalSource-API-Key*: ABCDEFGHIJKLMNOP
**Fields**:

- *code* - the unassigned VitalSource redemption code
- *sku* - the product SKU which is to be associated with the unassigned code
- *status* - the result of the mapping operation (valid values are: *mapped*, *mapped to different sku*, *already redeemed*, *code not found*, *sku not found*, *error*). The *error* status is only used when an unexpected error occurs.
- *remap* - [optional] when used it will unmap a previously map code and remap it to the new SKU. code cannot have been redeemed.

**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="XYZABCDEFBUZGHDKKABD" status="mapped" />
</mappings>

**Scenario 1**: Map an unassigned code which has never been mapped
**Request**: POST https://api.vitalsource.com/v3/unassigned_codes/mappings.xml
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings> <unassigned-code code="XYZABCDEFBUZGHDKKABD" sku="ABCD1234" />
</mappings>
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="XYZABCDEFBUZGHDKKABD" status="mapped" />
</mappings>

**Scenario 2**: Map an unassigned (and unredeemed) code which was mapped previously to the same SKU (with *remap* false)
**Request**: POST /v3/unassigned_codes/mappings.xml
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings> <unassigned-code code="YYYABCDEFBUZGHDKKABD" sku="ABCD1234"/>
</mappings>
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="YYYABCDEFBUZGHDKKABD" status="mapped" />
</mappings>

**Scenario 3**: Map an unassigned (and unredeemed) code which was mapped before to a different product (with *remap* false)
**Request**: POST /v3/unassigned_codes/mappings.xml
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings> <unassigned-code code="ZZZABCDEFBUZGHDKKABD" sku="ABCD1234" />
</mappings>
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="ZZZABCDEFBUZGHDKKABD" status="mapped to different sku" />
</mappings>

**Scenario 4**: Map an unassigned (and unredeemed) code which was mapped (with *remap* true)
**Request**: POST /v3/unassigned_codes/mappings.xml
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings> <unassigned-code code="ZZZABCDEFBUZGHDKKABD" sku="ABCD1234" remap="true"/>
</mappings>

**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="ZZZABCDEFBUZGHDKKABD" status="mapped" />
</mappings>

**Scenario 5 [error]**: Map a code which has already been redeemed
**Request**: POST /v3/unassigned_codes/mappings.xml
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings> <unassigned-code code="BBBABCDEFBUZGHDKKABD" sku="ABCD1234" remap="true" />
</mappings>
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="BBBABCDEFBUZGHDKKABD" status="already redeemed" />
</mappings>

**Scenario 6 [error]**: Map some codes which do not exist
**Request**: POST /v3/unassigned_codes/mappings.xml
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings> <unassigned-code code="AAAABCDEFBUZGHDKKABD" sku="ABCD1234" remap="true" />
<unassigned-code code="CCCABCDEFBUZGHDKKABD" sku="ABCD1234" remap="true" />
</mappings>
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="AAAABCDEFBUZGHDKKABD" status="code not found" />
<unassigned-code code="CCCABCDEFBUZGHDKKABD" status="code not found" />
</mappings>

**Scenario 7 [error]**: Map a code to a product which does not exist
**Request**: POST /v3/unassigned_codes/mappings.xml
**Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings> <unassigned-code code="AAAABCDEFBUZGHDKKABD" sku="DNEDNEDNE" remap="true" />
</mappings>
**Response Body**:
<?xml version="1.0" encoding="UTF-8"?>
<mappings>
<unassigned-code code="AAAABCDEFBUZGHDKKABD" status="sku not found" />
</mappings>

## Error Codes

| Code | Message |
|---|---|
| 104 | Invalid license type |
| 105 | Invalid expiration date |
| 106 | Number of days cannot be zero |
| 463 | Question is invalid |
| 464 | Account already exists |
| 465 | Data validation error |
| 466 | Your email or password was not accepted |
| 467 | Bundle is invalid |
| 468 | Redemption code is invalid |
| 469 | Code has been redeemed |

| | |
|---|---|
| 470 | Redemption failed |
| 473 | Invalid or expired signin token |
| 474 | User could not be found |
| 482 | Malformed create user request |
| 483 | Malformed redemption request |
| 484 | Malformed update user request |
| 601 | Invalid access token reference |
| 602 | Invalid GUID reference |
| 603 | Invalid reference value |
| 650 | Malformed credentials request |
| 900 | Insufficient permission to perform this action |
| 901 | Malformed XML request |
| 902 | Product sku could not be found |
| 903 | Access token and user do not match |
| 904 | User reference already exists |
| 905 | Invalid code request |
| 906 | Insufficient requirements for user create |
| 907 | Insufficient requirements for user update |
| 908 | Code price must be positive |
| 990 | Unable to merge users |
| 991 | Cannot merge user with itself |
| 992 | API user company is misconfigured (has multiple companies) |
| 993 | Package contains a non-existent SKU |
| 994 | Insufficient requirements for package create |
| 995 | Library ID is not acceptable |
| 996 | Product SKU already exists |