Direct Entitlement API

Version 2

Adobe Systems, Inc
February 2013

This document contains an API specification for the digital magazine Direct Entitlement API, which integrates with the Fulfillment server. Third parties that wish to integrate with the Fulfillment server must implement this API set. The viewer and the Fulfillment server will call these APIs as is documented here.

This is the second version of the API. Publisher's can now submit multiple configurations, one for each version of the API, to Adobe. The viewer will choose to use the latest version of the API, that it supports. Publishers will need to leave in service old versions of the API as long as viewers remain in the field that do not support newer versions of the API.

With the exception of the authentication calls, this is a REST style API.

Each method below should set the HTTP response code appropriately and should return the result in an XML snippet in the body of the response HTTP packet. The result should be wrapped in the following canonical XML element.

```
<result httpResponseCode=”200” errorCode=”0”>
...
</result>
```

The httpResponseCode is duplicated in the XML response because it is more compatible with Flex clients, which have limited access to the response code in the HTTP packet.

The attribute errorCode is optional. It may be set by the server to give additional information about the internal nature of the error.

Note: All communication between between the viewer, the fulfillment server and third party integration servers must use encrypted (SSL) http protocol to protect user credentials and device IDs.

Changes

Description	Date	Version Affected
Add optional subscriberType / subscriberID attributes to entitlements response	January, 4, 2013	2
Change signInWithCredential from GET to POST method	May 1, 2012	2
Pass emailAddress and password in the body of the request	May 1, 2012	2
Pass productId, coverDate pairs to getEntitlements	May 1, 2012	2
Pass coverDate to verifiyEntitlement	May 1, 2012	2
Create new version of the API	May 1, 2012	2
Add SSL requirement	February 24, 2011	1
Add appId to signInWithCredentials	February 2, 2011	1
Add appId to renewAuthToken	February 2, 2011	1
Add appId and uuid to getEntitlements	February 2, 2011	1
Add appId to verifyEntitlement	February 2, 2011	1

Sign In With Credentials

Authenticate with third party using the users account name and password

Method: POST
Path: /SignInWithCredentials

Query Parameter	Required	Type	Description
appId	no	string	The ID of the viewer application
appVersion	no	string	Version of the viewer application
uuid	no	string	Device UUID
Return		**Type**	**Description**
authToken		xml	An authentication token used to identify the authenticated user in future calls to the third party system.

Request Body: Will contain the following xml document specifiying the user credentials

```
<credentials>
        <emailAddress>joeblank@smooth.com</emailAddress>
        <password>stupid</password>
</credentials>
```

The emailAddress element is the user account name. It does not have to be an emailAddress, just a string that identifies the user in the third party system.

Notes: appId, appVersion and uuid should be treated as optional by the entitlement server. New viewers will always send this information, but older viewers will not.

Success Results:

```
HTTP 200 OK
Body:
<result httpResponseCode=”200”>
        <authToken>VFgrV1lKd09pL2s2NnlIKzE5RWJKUD...jMGExOGNiYWM</authToken>
</result>
```

Error Results:

```
HTTP 401 Unauthorized
Body:
<result httpResponseCode=”401” errorCode=””/>
```

Renew Auth Token

If an authentication token is present, use it to obtain a new auth token. This eliminates the need for storing the user’s account id and password on the device.

Method: GET
Path: /RenewAuthToken

Parameter	**Required**	**Type**	**Description**
authToken	yes	string	AuthToken returned by signInWithCredentials
appId	no	string	The id of the viewer application
appVersion	no	string	Version of the viewer app.
uuid	no	string	Device UUID
Return		**Type**	**Description**
authToken		xml	An authentication token used to identify the authenticated user in future calls to the third party system.

Notes: appId, appVersion and uuid should be treated as optional by the entitlement server. New viewers will always send this information, but older viewers will not.

Success Results:

```
HTTP 200 OK
Body:
<result httpResponseCode=”200”>
        <authToken>VFgrV1lKd09pL2s2NnlIKzE5RWJKUD...jMGExOGNiYWM</authToken>
</result>
```

Error Results:

```
HTTP 401 Unauthorized
Body:
<result httpResponseCode=”401”/>
```

Get Entitlements

Get the list of issues that the user is entitled to.

Method: POST
Path: /entitlements

Parameter	Required	Type	Description
authToken	yes	string	The authToken returned by signInWithCredentials.
appId	no	string	The id of the viewer application
appVersion	no	string	Version of the viewer application.
uuid	no	string	Device UUID
Return		**Type**	**Description**
entitlement list		xml	The list of issues the user is entitled to.

Notes: appId, appVersion and uuid should be treated as optional by the entitlement server. New viewers will always send this information, but older viewers will not.

Request Body: The request should contain the following xml document. The document lists the productIds and coverDates of the folio for which the caller needs entitlement status.

```
<folios>
        <folio>
                <productId>com.bonnier.flying.10.01.2010</productId>
                <coverDate>2011-10-11T20:49:40Z</coverDate>
        </folio>
        <folio>
                <productId>com.bonnier.flying.11.01.2010</productId>
                <coverDate>2011-11-11T20:49:40Z</coverDate>
        </folio>
        <folio>
                <productId>com.bonnier.flying.thanksgiving.special</productId>
                <coverDate>2011-12-11T20:49:40Z</coverDate>
        </folio>
        <folio>
                <productId>com.bonnier.flying.12.01.2010</productId>
                <coverDate>2012-01-11T20:49:40Z</coverDate>
        </folio>
</folios>
```

Example Success:

```
HTTP 200 OK
Body:
<result httpResponseCode=”200”>
    <entitlements>
        <productId subscriberType=”print” subscriberId=”a1234”>com.bonnier.flying.10.01.2010</productId>
        <productId subscriberType=”print” subscriberId=”a1234”>com.bonnier.flying.11.01.2010</productId>
        <productId subscriberType=”web” subscriberId=”c”>com.bonnier.flying.thanksgiving.special</productId>
        <productId subscriberType=”web” subscriberId=”c90”>com.bonnier.flying.12.01.2010</productId>
    <entitlements>
</result>
```

Notes: The productId should uniquely identify an issue and it should be the same ID that was assigned in the bundler when the issue was uploaded to the fulfillment server. This same ID should also be used to register the issue with the Apple Store for single issue in app purchase.

Auth Token Invalid Error Results:

```
HTTP 401 Unauthorized
Body:
<result httpResponseCode=”401”/>
```

Verify Entitlement

Verify that the user is entitled to a given issue.

Method: GET
Path: /verifyEntitlement

Parameter	**Required**	**Type**	**Description**
authToken	yes	string	The authToken returned by signInWithCredentials.
productId	yes	string	The product ID of the issue
coverDate	no	string	The cover date of the folio
appId	no	string	The id of the viewer application
appVersion	no	string	Version of the viewer application.
uuid	no	string	Device UUID
Return		**Type**	**Description**
result		xml	The results

Notes: appId, appVersion and uuid should be treated as optional by the entitlement server. New viewers will always send this information, but older viewers will not.

Example Entitled Results:

```
HTTP 200 OK
Body:
<result httpResponseCode=”200”>
        <entitled>true</entitled>
</result>
```

Example Not Entitled Results:

```
HTTP 200 OK
Body:
<result httpResponseCode=”200”>
        <entitled>false</entitled>
</result>
```

Notes: The productId should uniquely identify an issue and it should be the same ID that was assigned in the bundler when the issue was uploaded to the fulfillment server. This same ID should also be used to register the issue with the Apple Store for single issue in app purchase.

Auth Token Invalid Error Results:

```
HTTP 401 Unauthorized
Body:
<result httpResponseCode=”401”/>
```