MAY 11, 2017

ACQUISITIONS API QUICK START GUIDE

REVISION 1.3

INNOVATIVE INTERFACES

Overview

The purpose of this document is to assist the experienced software developer who wants to quickly come up to speed on how to use the Sierra acquisitions API to create a better and more streamlined ordering experience for Sierra library staff.

It's likely that most of your library customers today use Innovative's Quick Click or some other homegrown manual/semi-automated multi-step ordering process that enables them to export a file containing order and bib information from your system, transfer it using FTP or EDI and then load those records into Sierra.

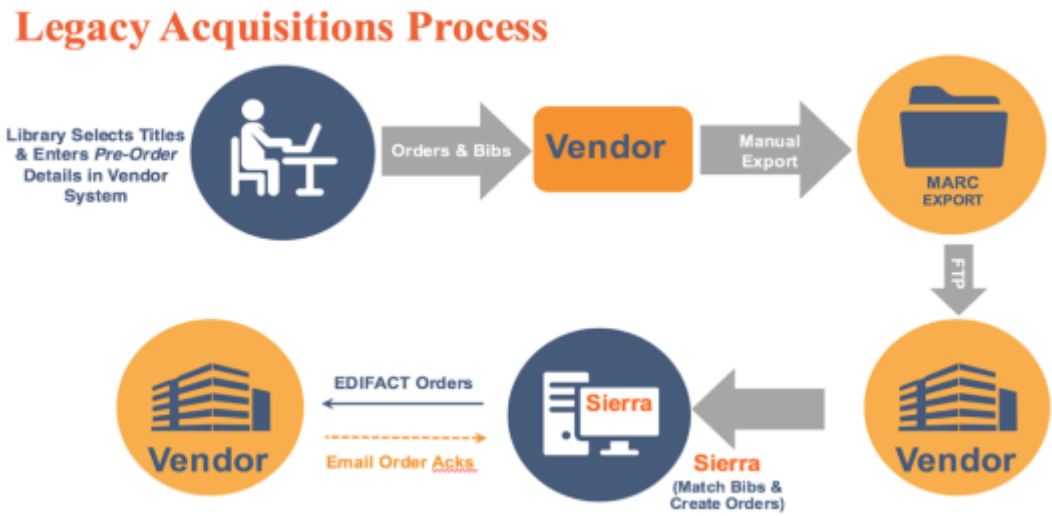


Figure 1: Legacy Acquisitions Process

The new acquisitions API allows users to submit orders directly from your ordering software to the library's Sierra system in real time, without requiring traditional FTP or EDI ordering and reducing the number of manual steps.

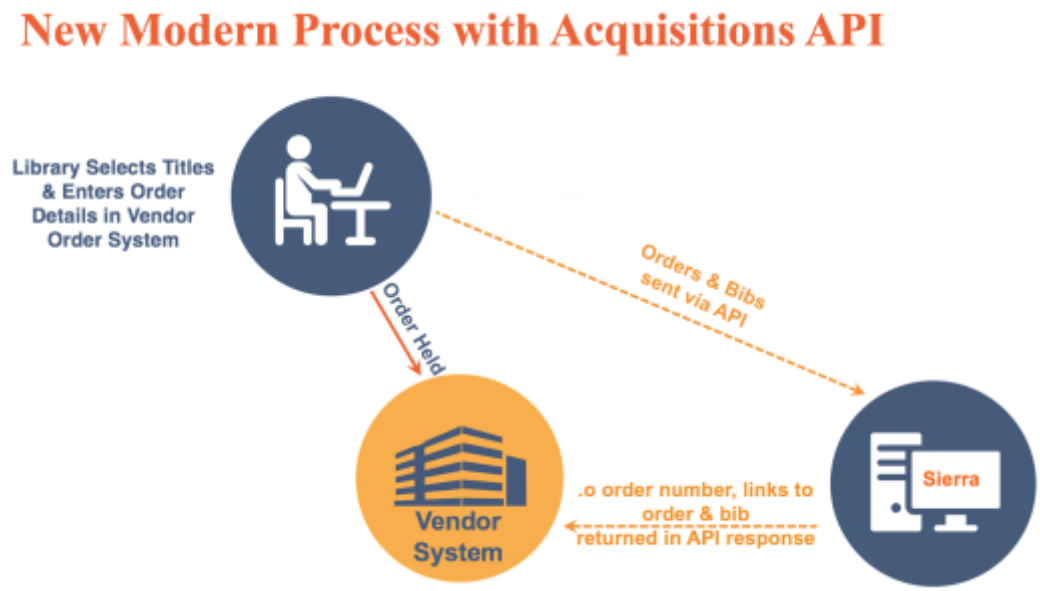


Figure 2: New Modern Acquisitions Process

Acquisitions API Endpoints

There are two endpoints that comprise the acquisitions API:

- **POST /v3/acquisitions/orders/validate** — During the set-up process, validates the required order parameters supplied to the vendor by the library.
- **POST /v3/acquisitions/orders** — Creates a new order in Sierra using order and bib information sent as body data and returns an Innovative .o order record number.

There is a third endpoint that performs the same function as **POST /v3/acquisitions/orders** but it has been deprecated and should not be used:

- ~~POST /v3/acquisitions/orders/classic~~ — **Do not use, deprecated.**

Set-up

You have two options for testing the acquisitions API. You can either test against Innovative's Sierra sandbox server that is loaded with mock data or you can test against the library's Sierra training server if they have one. Theoretically you could also test against the library's production server but this is not recommended. Depending on your situation, you should follow the Innovative Sandbox Set-up or Library Training Server Set-up procedure outlined below.

Innovative Sandbox Set-up

If you are using Innovative's sandbox server to develop your integration, follow these steps:

a. Obtain a sandbox API key from Innovative by signing up at developer.iii.com and then follow the Sierra REST API Interactive Documentation & Sandbox section of Getting Started on the developer portal. After obtaining your API key and secret, enter them below or elsewhere for future reference:

Parameter	Description	Value
API Key	API key received from Innovative	
API Secret	Secret you chose for the API key	

Table 1: Sandbox API Key and Secret

b. The following table has the information you need to access the acquisitions endpoints on the sandbox server as well as test data you can use for the required parameters:

Parameter	Description	Value
App Server Hostname	Fully Qualified DNS name of the Sierra app server (e.g., www.library.com)	https://sandbox.iii.com/
Base URL	Used to access the Sierra API endpoints on the Sierra app server. Construct a Base URL using the app server hostname concatenated with /iii/sierra-api/. For example, if the library's app server hostname is www.library.com, then	https://sandbox.iii.com/iii/sierra-api

	the Base URL would be https://www.library.com/iii/sierra-api/.	
Authentication URL	Creates an OAuth2 token.	https://sandbox.iii.com/iii/sierra-api/v3/token
Acquisitions URL	Create a new order.	https://sandbox.iii.com/iii/sierra-api/v3/acquisitions/orders
login	The Sierra username to associate with created orders. Login controls the accounting unit assigned to orders.	Any one of 1. acqstaff1 2. acqstaff2 3. acqstaff3
location	The shelving locations or branches for which the order is placed	Any or all of 1. 00 2. 00aca 3. 00ar
fund	The fund code against which the material is encumbered and paid	Any or all of 1. acfic 2. acnf 3. jcpb
vendor	The library-defined code for the vendor who handles the order	Any one of 1. vend1 2. vend2 3. vend3

Table 2: Sandbox Test Data

Library Training Server Set-up

If you are using the library's Sierra training server to develop your integration, follow these steps:

a. Obtain an API key and create a secret from the library system administrator.
   i. The library system administrator will need some information from you when they create your API key. Provide the following information to your library system administrator.
      1. Name — Name of the API key recipient within your organization
      2. Email address — Email address of the API key recipient within your organization

   ii. Ask the library system administrator to create an API key for you with minimally the **Acquisitions Write** (allows you to create order records in Sierra) permission. The administrator may also want to assign you the **Orders Read** permission (allows you to read order records) or other permissions depending on the application you're developing.

   NOTE: See the Accessing the Sierra REST API at a Library Site section of Getting Started on the developer portal for more details on API key creation for your library's server.

b. After obtaining your API key and secret, enter them below or elsewhere for future reference:

Parameter	Description	Value
API Key	API key received from the library	
API Secret	Secret you chose for the API key	

Table 3: Library Training Server API Key and Secret

c. In addition to the API key and secret, you will need to ask the library for the following information marked below with an asterisk (*). The other parameters below can be derived from those received from the library as described. Enter the library supplied values and then derive the other values and write them in the table below for easy reference:

Parameter*	Description	Value
App Server Hostname*	Fully Qualified DNS name of the Sierra app server (e.g., www.library.com)	
Base URL	Used to access the Sierra API endpoints on the Sierra app server. Construct a Base URL using the App Server Hostname obtained from your library by concatenating the Base URL with /iii/sierra-api/. For example, if the library's app server hostname is www.library.com, then the Base URL would be https://www.library.com/iii/sierra-api.	
Authentication URL	Creates an OAuth2 token. Create this URL by concatenating the Base URL with /v3/token. For example, http://www.library.com/iii/sierra-api/v3/token	
Acquisitions URL	Used to create a new order in Sierra. Create this URL by concatenating the Base URL with /v3/acquisitions/orders. For example, https://www.library.com/iii/sierra-api/v3/acquisitions/orders	
login*	The Sierra username to associate with created orders. Login controls the accounting unit assigned to orders.	
location*	The shelving locations or branches for which the order is placed	
fund*	The fund code against which the material is encumbered and paid;	
vendor*	The library-defined code for the vendor who handles the order. NOTE: You may want to ask the library to set up additional unique codes so that orders made through	

	the API can be easily distinguished from regular orders during the testing period.	

Table 4: Library Training Server Test Data ***Provided by the Innovative library**

NOTE: The Sierra login determines the accounting unit assigned to orders. And the accounting unit determines the fund codes that can be used to create an order and encumber the fund.

d. Lastly, the load profile on the library's server at $IIIDB/data/m2btab.oapi will need to be updated so that the incoming MARC data for the items being ordered is mapped to the correct fields within the order and bib records that represent the new order in the library's Sierra catalog. At around line 84 in this file, there's a comment, "add MARC tag section from site's m2btab.b", that marks the point where the library's MARC tags should be inserted from the site's m2btab.b file. If the library has a load profile trained person on staff, they can update the m2btab.oapi load profile themselves. Otherwise, Innovative offers a service through our professional services team to update this file. Please ask the library to contact an Innovative sales representative if the library would like to receive a quote for this service.

Test Basic API Connectivity

Test your ability to connect to the Sierra API server over the internet or from some other network connection by constructing a URL to access the "about" endpoint for the API on the server. For a library's server, type <Base URL for library>/about into your browser (for example, for Innovative's sandbox server, type https://sandbox.iii.com/iii/sierra-api/about). If you get a response similar to:

Response Received	Description
Sierra Rest API	Name of API
Version: 3.2.0	Sierra REST API Version
SierraVersion: 3.0	Sierra Version
FullVersion: 3.0_3.2.0	Sierra Version + Sierra REST API Version
Revision: 5a8d3671d83ea9afc67906f4a4b2e15a4ab5f777	Unique String Identifying this Revision
RevisionDate: Tue Jan 10 00:25:00 2017 +0000	Revision Date
Backwards Compatible: 2.0, 2.1, 3.0, 3.1	Sierra API Versions Accessible

You have successfully established connectivity with the Sierra API. If you do not receive a response similar to the one above, please log a ticket with Innovative.

Interactive Documentation

Once basic API connectivity is confirmed, the easiest way to experiment with the APIs is to use the interactive swagger documentation. It allows you to interact with the APIs and see how each endpoint behaves without

writing code. Each server has its own swagger documentation. Using your browser, access the interactive swagger docs:

- **On the Sandbox:** Sandbox Interactive Swagger Documentation

- **On Your Library Training Server** – Interactive swagger documentation is available on the library's Sierra training server at **<<Base URL from Table 4>>**/swagger/index.html. You should obtain the library's permission before using any of their servers for testing since real order records will be created. Remember to clean up and delete any order records that were created during testing.

Enter your API Key and Secret (from Table 1 for the Sandbox or from Table 3 for your library) in the two boxes at the top right of the swagger documentation home page and hit return. This authenticates you and allows you to make API calls.

Figure 3: Swagger Client Key / Client Secret fields

Validate Required Parameters

Now that you're authenticated, let's validate the required parameters to make sure they match the values configured on the server. In this example, we will use Innovative's sandbox server and corresponding parameter test data given above in Table 2. If you are using the library's training server, you will need to use values for the parameters that were supplied by the library and recorded in Table 4.

Once you receive a successful response ensuring that the values for login, location, fund, and vendor, are valid, you don't have to make any subsequent calls to this endpoint to create orders. It's only used once to validate the most critical order parameters.

In the Acquisitions section of the swagger documentation on the sandbox, find the **POST /v3/acquisitions/orders/validate** endpoint and perform the following steps:

1. Construct an HTTP body by copying/pasting or typing the HTTP body (below) into swagger for this endpoint:

	COPY/PASTE
HTTP Body	{ "login": "acqstaff1", "copies": 4, "allocation": [{ "location": "00", "fund": "acfic" }, { "location": "00aca",

	"fund": "acnf"
	},
	{
	"location": "00ar",
	"fund": "jcpb"
	}
	],
	"vendor": "vend1"
	}

2. Click “Try it now”.

Here are example success and error responses that you may receive:

Success Response

Success - HTTP Response Code 204

If successful, move on to “Copies Parameter Explained”

Error Response

Error – HTTP Response Code other than 204. For example,

```
{
 "code": 108,
 "specificCode": 0,
 "httpStatus": 400,
 "name": "Invalid parameter",
 "description": "Invalid parameter: fund code: 030eb not valid for login: acqstaff1"
}

Response code 400
```

If you received an error, before proceeding, make sure that the values for login, location, fund, and vendor match what’s configured within Sierra and that you receive a successful HTTP 204 response.

COPIES PARAMETER EXPLAINED

When there is only one location/fund/copies tuple within the allocation array for an order, you should not specify a global copies parameter (Z) outside of the allocation array. In this case, the only copies value should be specified along with the location and fund within the allocation array.

When there is more than one location and fund within an allocation array for an order, it is suggested that you specify copies for each location and fund within the allocation array and set the global copies parameter (Z) to be the sum of all of the copies parameters within the allocation array (e.g., Z = a + b + c). There is a complex set of rules that are followed when the global copies parameter (Z) is not equal to the sum of all the copies parameters listed within the allocation array. Therefore, it is simpler if you explicitly set the global copies parameter (Z) to be equal to the sum of all the copies parameters within the allocation array so that the allocation rules are not applied.

In the event that the global copies parameter (Z) is not equal to the sum of all the copies parameters listed within the allocation array (a, b, and c), the following table shows how the allocation rules are applied.

```
{
        "login":"acqstaff",
         "copies": Z,
        "allocation":[
        {
                "location":"00",
                "fund":"acfic"
                 "copies": a
        },
        {
                "location":"00aca",
                "fund":"acnf"
                 "copies": b
        },
        {
                "location":"00ar",
                "fund":"jcpb"
                 "copies": c
        }
        ],
        "vendor":"vend1"
}
```

Z, a, b, c are integers > 0

Scenarios	#Example Values	#Copies Placed in New Order Record
If Z > a + b + c	Z = 10, a = 4, b = 3, c = 1	a = 6, b = 3, c = 1
If Z = a + b + c	Z = 8, a = 4, b = 3, c = 1	a = 4, b = 3, c = 1
If Z < a + b + c	Z = 5, a = 4, b = 3, c = 1	a = 4, b = 1, c = 0
If a not provided	Z = 10, b = 3, c = 1	a = 1, b = 3, c = 6

If b not provided	Z = 10, a = 4, c = 1	a = 4, b = 1, c = 5
If c not provided	Z = 10, a = 4, b = 3	a = 4, b = 3, c = 3
If a and b not provided	Z = 10, c = 1	a = 1, b = 1, c = 8
If a and c not provided	Z = 10, b = 3	a = 1, b = 3, c = 6
If b and c not provided	Z = 10, a = 4	a = 4, b = 1, c = 5
If a, b and c not provided	Z = 10	a = 1, b = 1, c = 8
If Z not provided, then no order will be placed, regardless of what values are assigned to a, b, or c.	a = 4, b = 3, c = 1	No order will be placed

Figure 4: Copies Parameters

Place an Order

Since you have validated the required parameters, let's create an order.

Find the **POST /v3/acquisitions/orders** endpoint in the Acquisitions section of the swagger documentation.

Before placing an order, you must construct an HTTP body consisting of the order and bib details. The order details are listed first and then followed by the MARC record of the bib that is being ordered. You may choose to use JSON or XML for your data format by specifying the parameter content type in the HTTP header as marc-json, marc-in-json, or marc-xml. A minimum MARC record consists of a leader, directory, 008 field, and 245 field.

Here are two working examples, one simple and one more complex, for the Innovative sandbox. Perform the following steps:

1. Copy/paste the HTTP Body into swagger:

Simple Example	
API Endpoint	POST /v3/acquisitions/orders
Content-type	marc-in-json
HTTP Body	{ "order": { "login": "acqstaff1", "copies": 10, "allocation": [{ "location": "00", "fund": "acfic" }], "vendor": "vend1" }, "marcContentType": "application/marc-in-json",

```
"marc": {
        "leader": "00000nam a2200000 a 4500",
        "fields": [{
                "001": "8739"
        }, {
                "008": "690612s1969    caua    b    001 0 eng  cam   "
        }, {
                "100": {
                        "subfields": [{
                                "a": "Albrecht, Bob,"
                        }, {
                                "d": "1930-"
                        }],
                        "ind1": "1",
                        "ind2": " "
                }
        }, {
                "700": {
                        "subfields": [{
                                "a": "Lindberg, Eric."
                        }],
                        "ind1": "1",
                        "ind2": " "
                }
        }, {
                "700": {
                        "subfields": [{
                                "a": "Mara, Walter."
                        }],
                        "ind1": "1",
                        "ind2": " "
                }
        }, {
                "650": {
                        "subfields": [{
                                "a": "Mathematics"
                        }, {
                                "x": "Data processing."
                        }],
                        "ind1": " ",
                        "ind2": "0"
                }
        }, {
                "650": {
                        "subfields": [{
```

```
                                        "a": "Computers."
                                }],
                                "ind1": " ",
                                "ind2": "0"
                        }
                }, {
                        "650": {
                                "subfields": [{
                                        "a": "Programming languages (Electronic
computers)"
                                }],
                                "ind1": " ",
                                "ind2": "0"
                        }
                }, {
                        "010": {
                                "subfields": [{
                                        "a": "77004256"
                                }],
                                "ind1": " ",
                                "ind2": " "
                        }
                }, {
                        "504": {
                                "subfields": [{
                                        "a": "Bibliography: p. 202."
                                }],
                                "ind1": " ",
                                "ind2": " "
                        }
                }, {
                        "967": {
                                "subfields": [{
                                        "a": "ib10001207"
                                }],
                                "ind1": " ",
                                "ind2": " "
                        }
                }, {
                        "260": {
                                "subfields": [{
                                        "a": "Menlo Park, Calif.,"
                                }, {
                                        "b": "Addison-Wesley Pub. Co."
                                }, {
```

```
                    "c": "[1969]"
                }],
                "ind1": " ",
                "ind2": " "
            }
        }, {
            "300": {
                "subfields": [{
                    "a": "204 p."
                }, {
                    "b": "illus."
                }, {
                    "c": "28 cm."
                }],
                "ind1": " ",
                "ind2": " "
            }
        }, {
            "245": {
                "subfields": [{
                    "a": "Computer methods in mathematics"
                }, {
                    "c": "[by] Robert L. Albrecht, Eric Lindberg [and]
Walter Mara."
                }],
                "ind1": "1",
                "ind2": "0"
            }
        }, {
            "040": {
                "subfields": [{
                    "a": "DLC"
                }, {
                    "c": "DLC"
                }, {
                    "d": "CSJ"
                }],
                "ind1": " ",
                "ind2": " "
            }
        }, {
            "049": {
                "subfields": [{
                    "a": "CSJS"
                }],
```

```
                "ind1": " ",
                "ind2": " "
            }
        }, {
            "050": {
                "subfields": [{
                    "a": "QA76.5"
                }, {
                    "b": ".A368"
                }],
                "ind1": " ",
                "ind2": " "
            }
        }, {
            "910": {
                "subfields": [{
                    "a": "eng"
                }, {
                    "b": "901010"
                }, {
                    "c": "m"
                }, {
                    "d": "a"
                }, {
                    "e": "cau"
                }],
                "ind1": " ",
                "ind2": " "
            }
        }, {
            "910": {
                "subfields": [{
                    "a": "12APR78rev LB 8739"
                }],
                "ind1": " ",
                "ind2": " "
            }
        }]
    }
}
```


Figure 5: Simple Order

Example Response

Success - HTTP Response Code 200

```
{
 "controlNumber": "8739",
 "orderId": "https://sandbox.iii.com/iii/sierra-api/v3/orders/1760325",
 "bibId": "https://sandbox.iii.com/iii/sierra-api/v3/bibs/1000109",
 "legacyOrderId": ".o17603250"
}
```


Failure – 400 Bad request


```
{
"code": 115,
"specificCode": 0,
"httpStatus": 400,
"name": "Invalid JSON",
"description": "JSON object missing field or field has invalid data"
}
```


Complex Example	
API Endpoint	POST /v3/acquisitions/orders

HTTP Body	{
	"order": {
	"login": "acqstaff1",
	"copies":10,
	"allocation": [
	{
	"location":"00",
	"fund":"acfic",
	"copies": 4
	},
	{
	"location":"00aca",
	"fund":"acnf",
	"copies": 3
	},
	{
	"location":"00ar",
	"fund":"jcpb",
	"copies": 3
	}
	],
	
	"vendor": "vend1"
	},
	"marcContentType": "application/marc-in-json",
	"marc": {

```
"leader": "00000nam a2200000 a 4500",
"fields": [{
 "001": "8739"
}, {
 "008": "690612s1969    caua     b    001 0 eng  cam   "
}, {
 "100": {
  "subfields": [{
   "a": "Albrecht, Bob,"
  }, {
   "d": "1930-"
  }],
  "ind1": "1",
  "ind2": " "
 }
}, {
 "700": {
  "subfields": [{
   "a": "Lindberg, Eric."
  }],
  "ind1": "1",
  "ind2": " "
 }
}, {
 "700": {
  "subfields": [{
   "a": "Mara, Walter."
  }],
  "ind1": "1",
  "ind2": " "
 }
}, {
 "650": {
  "subfields": [{
   "a": "Mathematics"
  }, {
   "x": "Data processing."
  }],
  "ind1": " ",
  "ind2": "0"
 }
}, {
 "650": {
  "subfields": [{
   "a": "Computers."
```

```
    }],
    "ind1": " ",
    "ind2": "0"
   }
  }, {
   "650": {
    "subfields": [{
     "a": "Programming languages (Electronic computers)"
    }],
    "ind1": " ",
    "ind2": "0"
   }
  }, {
   "010": {
    "subfields": [{
     "a": "77004256"
    }],
    "ind1": " ",
    "ind2": " "
   }
  }, {
   "504": {
    "subfields": [{
     "a": "Bibliography: p. 202."
    }],
    "ind1": " ",
    "ind2": " "
   }
  }, {
   "967": {
    "subfields": [{
     "a": "ib10001207"
    }],
    "ind1": " ",
    "ind2": " "
   }
  }, {
   "260": {
    "subfields": [{
     "a": "Menlo Park, Calif.,"
    }, {
     "b": "Addison-Wesley Pub. Co."
    }, {
     "c": "[1969]"
    }],
```

```
    "ind1": " ",
    "ind2": " "
   }
  }, {
   "300": {
    "subfields": [{
     "a": "204 p."
    }, {
     "b": "illus."
    }, {
     "c": "28 cm."
    }],
    "ind1": " ",
    "ind2": " "
   }
  }, {
   "245": {
    "subfields": [{
     "a": "Computer methods in mathematics"
    }, {
     "c": "[by] Robert L. Albrecht, Eric Lindberg [and] Walter Mara."
    }],
    "ind1": "1",
    "ind2": "0"
   }
  }, {
   "040": {
    "subfields": [{
     "a": "DLC"
    }, {
     "c": "DLC"
    }, {
     "d": "CSJ"
    }],
    "ind1": " ",
    "ind2": " "
   }
  }, {
   "049": {
    "subfields": [{
     "a": "CSJS"
    }],
    "ind1": " ",
    "ind2": " "
   }
```

```
    }, {
      "050": {
        "subfields": [{
          "a": "QA76.5"
        }, {
          "b": ".A368"
        }],
        "ind1": " ",
        "ind2": " "
      }
    }, {
      "910": {
        "subfields": [{
          "a": "eng"
        }, {
          "b": "901010"
        }, {
          "c": "m"
        }, {
          "d": "a"
        }, {
          "e": "cau"
        }],
        "ind1": " ",
        "ind2": " "
      }
    }, {
      "910": {
        "subfields": [{
          "a": "12APR78rev LB 8739"
        }],
        "ind1": " ",
        "ind2": " "
      }
    }]
  }
}
```


Figure 6: Complex Order

Example Responses

Success - HTTP Response Code 200

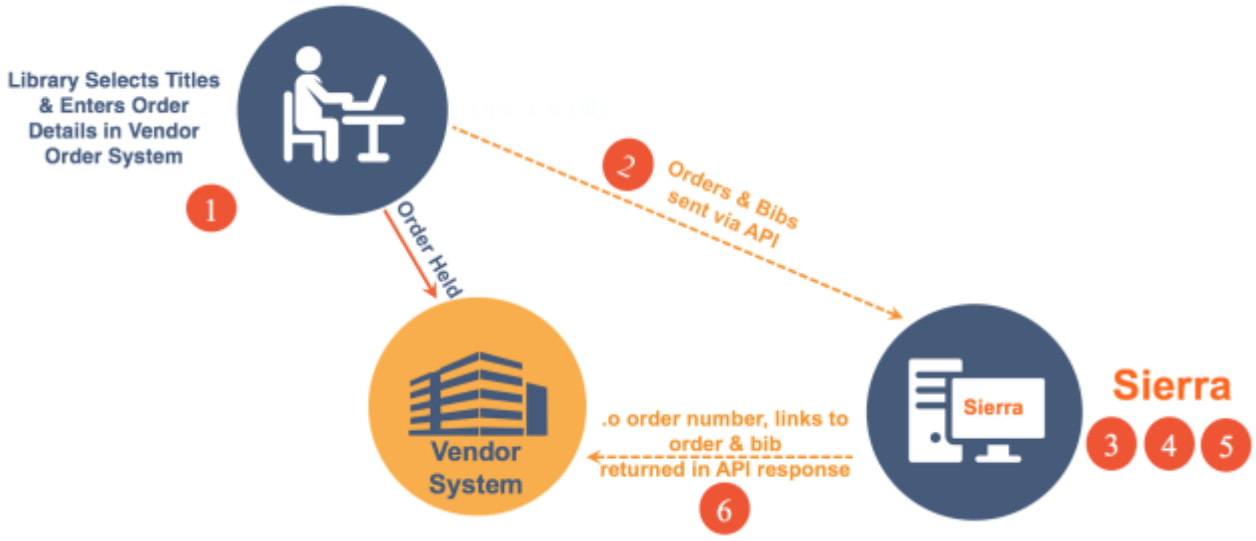
```
{
  "controlNumber": "8739",
  "orderId": "https://sandbox.iii.com/iii/sierra-api/v3/orders/1760326",
  "bibId": "https://sandbox.iii.com/iii/sierra-api/v3/bibs/1000109",
  "legacyOrderId": ".o17603262"
}
```


Failure – 400 Bad request


```
{
"code": 115,
"specificCode": 0,
"httpStatus": 400,
"name": "Invalid JSON",
"description": "JSON object missing field or field has invalid data"
}
```


How it Works

Using the example data in Figures 5 or 6, here are the steps the software performs to create the order:



1. Library selects titles and enters order details in vendor order system
2. The vendor packages up the order and bib data as shown in Figures 5 & 6 and sends it to the Sierra server via the API. The order is also held in the vendor order system.
3. Sierra parses the parameters data and validates that required parameters have been provided.
4. The login parameter is checked in Sierra to ensure the proper permissions are in place for that Sierra account and it determines the accounting unit associated with that login. That accounting unit is used for the purchase.
5. Sierra parses the order and bib data and creates Innovative bib and order records according to the rules outlined in the load profile on the Sierra server named m2btab.oapi. This load profile ships with every copy of Sierra and is used to customize how the order and bib data submitted through the API

gets mapped to fields in Innovative order and bib records and sets the match point for detecting duplicate bibs and specifies the action to take when a duplicate bib is detected.

6. After the bib and order records have been created, a response is sent to the Acquisitions API consumer that includes an HTTP status code, control number, orderID (link to the order record), bibID (link to the bib record), and legacyOrderID (the orderID prepended with .o and a check digit appended to it).

Building the Software

Authenticate with the API

Now that you are certain the values you have for the required parameters are valid, let's start writing some code. The first task is to write code to acquire an auth token by sending a valid API Key and Secret to the Sierra server. Review this tutorial on how to authenticate with the API and see example code written in Ruby.

After you have a valid auth token, you will be able to use it to make calls to the acquisitions API endpoint provided you were granted the proper permissions when your API key was created.

Create Orders with the Acquisitions API

Once you have an auth token, you can begin creating orders. Let's assume your ordering software handles the selection of the title and enables the user to specify the order details (e.g., number of copies, selector, etc):

Your software will package up the bib and order information into an XML or JSON body similar to what's shown in Figure 5 or Figure 6 (i.e., the order details followed by the title in JSON or XML format).

Next you will use a REST API library of your choice to make a call to the **POST /v3/acquisitions/orders** endpoint. After an order is created, Sierra returns the Innovative. o order number in its response – see figures 5 and 6. That number can then be used to invoice the customer. See example code written in Ruby.

Verify Orders and Bibs

After you've received a successful response to the **POST /v3/acquisitions/orders** endpoint, you should work with the library site administrator to verify that the order and bib records were created as you expected them to be. Although there could be a number of reasons why the order or bib record is not correct, most often it is because the mappings in the m2btab.oapi load profile need to be adjusted. If the library has a load profile trained person on staff, they can update the m2btab.oapi load profile themselves. Otherwise, Innovative offers a service through our professional services team to update this file. Please contact an Innovative sales representative if the library would like to receive a quote for this service.